

SETLabs Briefings

VOL 6 NO 4
2008

BUSINESS INNOVATION through TECHNOLOGY

EFFECTIVE BUSINESS ANALYSIS



Infosys®

POWERED BY INTELLECT
DRIVEN BY VALUES

Power your IT projects with EBA

SETLabs Briefings Advisory Board

Avejeet Palit

Principal Solutions Manager,
System Integration Practice

Gaurav Rastogi

Associate Vice President,
Head - Learning Services

George Eby Mathew

Senior Principal,
Infosys Australia

Kochikar V P PhD

Associate Vice President,
Education & Research Unit

Raj Joshi

Managing Director,
Infosys Consulting Inc.

Rajiv Narvekar PhD

Manager,
R&D Strategy
Software Engineering &
Technology Labs

Ranganath M

Vice President &
Chief Risk Officer

Srinivas Uppaluri

Vice President & Global Head,
Marketing

Subu Goparaju

Vice President & Head,
Software Engineering &
Technology Labs

As organizations are transitioning from linear lifecycles to more dynamic and disruptive ones the alignment of IT with business is becoming more crucial and questionable. More and more software projects are failing to deliver on their expectations. While some are being abandoned, some are challenged with innumerable performance constraints. There are reports abound on the failure of software projects, each with different estimates of the cost of failure. The annual cost of software project failures though can be comfortably pegged between \$50 and \$80 billion.

Budget overruns, higher maintenance costs, lesser ROI and dismal business value addition have plagued IT projects for quite long. And the moot question remains: *why are IT projects failing at an alarming rate?* Is there a systemic lacunae at development, integration and governance levels or aren't business requirements being properly captured by business analysts? Previous issues of SETLabs Briefings have drawn your attention to the systemic problems and addressed some nudging concerns. Some of you have suggested that 'business analysis' could be one of the primary reasons for such failures. We have taken cue from your feedback and put together an issue that addresses some important 'business analysis' concerns.

It is common knowledge that business analysts grapple with the challenges of understanding clients' business requirements. The sharpness of assimilating such requirements is lost in the mire of *islands of information* gathered from different stakeholders. There is thus an immediate need for the stakeholders to depart from their "I want this" mindset and embrace "this is my business objective, so let us discuss what my requirements could be" attitude. Only then can business analysts do justice to effectively capturing business requirements even while visualizing what the end solution should look like.

This issue contains a *mélange* of very interesting insights into model driven tools and technologies to capture business requirements. We also share ideas on how the benefits offered by functional storyboards, general semantics, behavioral slicing and OLAR model can be leveraged in effectively analyzing business requirements.

Huai-Ling Ch'ng, consulting editor for the issue fleshes out the core message of the theme by aptly closing *The Last Word*: "as long as business analysts understand the rationale for why they are doing what they are doing, they will continue to build many cathedrals in the years to come."

I hope you enjoy reading this issue as much as I have in putting it together for you.

Needless to mention, your feedback, as always adds fuel to our desire to provide you the best technology insights in every issue of SETLabs Briefings. So please do write in to me with your comments and suggestions.



Praveen B. Malla PhD
praveen_malla@infosys.com
Editor

Research Review: Building Blocks for Enterprise Business Architecture*By Eswar Ganesan and Ramesh Paturi*

In this paper the authors study all the established meta-models and frameworks of Enterprise Business Architecture and extract the key ingredients to build a meta-model of their own. This composite meta-model, they feel, can lead to a better definition of business architecture thus ensuring effective business analysis.

3

Opinion: Functional Storyboards: Requirement Elicitation tool for COTS Evaluation*By Nandini Toley*

Fitment evaluation of the COTS package is a must before it is adopted for an organization. The author opines that functional storyboarding can help in perfectly estimating and comparing the workflows in the existing system with the superior ones offered by the COTS package.

15

Viewpoint: Improving Requirements Accuracy through General Semantics*By Badhri Narayan C. S.*

The business analyst is swarmed with ambiguous communication while capturing requirements. There is a very little chance of requirements going amiss, if General Semantics is applied to analyze requirements, feels the author.

21

Framework: Business Analysis Using S2S Framework*By Anshuman Pramanick*

Improving ROI of strategic IT initiatives is an imperative. Bad business-IT alignment can play a spoilsport. The author suggests an S2S framework that helps validate IT requirements against business strategy and maintain a positive business-IT alignment.

29

Perspective: Integrated Approach to Requirements Engineering*By Ashish Chandra, Ravishankar N and Tushar Sharma*

The authors feel that poor elicitation of requirements can be detrimental to the progress of an IT implementation. They suggest an integrated approach that includes a standardized and structured set of procedures to Requirements Engineering that can prove more fruitful than the existing ones.

37

Practitioner's Perspective: Are you Plagued with Unnecessary Costs in your IT Projects?*By Manu Goel*

IT implementations are riddled with unforeseen costs and unexpected expenditures. How does one mitigate unwanted budgetary shocks? Objective linked approach for requirements development (OLAR) can come in handy to counter such problems, assures the author.

47

Insight: Strategy Value Traceability in Enterprise Requirements Elicitation*By Luke Housego*

Requirements keep changing all through the requirements elicitation phase. How does then one assess their value under dynamic conditions? Tie requirements to a value framework and look at them as enablers of strategy to ascertain their contributory value, asserts the author.

55

Practitioner's Solution: A Novel Approach to Software Requirements Analysis*By Anjaneyulu Pasala PhD and Ravi Prakash Gorthi PhD*

The authors offer behavioral slicing as a unique way for requirements gathering and analysis. Apart from finding gaps and errors in functional requirements, it also helps the business analyst in adopting a structured approach towards business analysis through validations.

63

Spotlight: Requirements Management in Multi-project Environments*By Manideepa Chatterjee, Sandeep Anand M and Tejal Prakash Nemane*

Identifying dependent and non-dependent requirements in a multi-project scenario is the key to a successful implementation of an IT program. The authors suggest that an early detection of overlaps can save cost and duplication and thus lead to effective requirements management.

71

The Last Word: Effective Business Analysts Build Cathedrals*By Huai-Ling Ching*

79

Index

81



Behavioral slicing of requirements is a path-breaking method to discover ambiguities and incompleteness early in the SDLC.

Ravi Gorthi Ph D

Principal Researcher and Head

Test Automation and Knowledge Engineering Labs

SETLabs, Infosys Technologies Limited



Objective linked approach for requirements development (OLAR) helps in mapping objectives to customer requirements thereby paring unnecessary IT costs that are otherwise incurred either due to incomplete requirements gathering or improper understanding of requirements.

Manu Goel

Senior Project Manager

Banking and Capital Markets Unit

Infosys Technologies Limited

Building Blocks for Enterprise Business Architecture

By Eswar Ganesan and Ramesh Paturi

A unified meta-model of elements can lead to effective business analysis

Enterprise Architecture (EA) is the blueprint of an organization's vision and provides a comprehensive view of the business strategy, processes, information components, applications and technology platforms used by it. According to TOGAF [1], there are four kinds of architecture that are commonly accepted as subsets of overall enterprise architecture - business, data, applications and technology. The focus of this paper is on business architecture. Business Architecture provides the much needed link to business strategy and the other major architectures - information (data), applications and security [2]. The scope of business architecture can vary in practical scenarios and can be deployed at business unit level or department level and when performed with an enterprise-wide scope, it qualifies to become an Enterprise Business Architecture. Enterprise Business Architecture (EBA) is a definition of what the enterprise must produce to satisfy its customers, compete in a market, deal with its suppliers, sustain operations and care for its employees [3]. EBA becomes

essential in the existing complex business scenario as it attempts to create a blueprint of why and how business is done while detailing the enterprise's vision, strategy, processes and strategy execution.

Various frameworks have been conceptualized by architects, industry players and research organizations since the beginning of EA practice. Each framework has a set of basic building blocks (commonly referred to as elements) defined. An architect would refer to multiple frameworks in order to architect the enterprise in accordance with the enterprise's requirement/constraint. This paper discusses extant frameworks and provides a comparative analysis of EBA elements used in each framework and later comes up with Composite EBA framework element list. Most of the EA today is based on Zachman Framework and we use the abstractions of Zachman framework for comparing the elements across various frameworks [4]. We also present a meta-model for EBA and identify some future possibilities.

COMPARISON OF EBA ELEMENTS ACROSS MULTIPLE FRAMEWORKS

Business Architecture defines the business strategy, governance, organization and key business processes [1]. An EBA defines the enterprise value streams and their relationships with all external entities, other enterprise value streams and events that trigger instantiation [3]. Ralph Whittle and Conrad Myrick in their book, 'Enterprise Business Architecture: The Formal Link between Strategy and Results' detail that almost every enterprise today lacks formal business architecture [3]. Such architecture and blueprints are critical in developing and maintaining complex business enterprises because one of the keys to successful strategic planning and engineering is an integrated architecture approach and it all begins with EBA and its component linkages. So, we can define EBA as the structure of components related to business and the manner these components interrelate among themselves and other architectures viz., data, application and technology, to create business value. In our study, we concentrate only on identifying an exhaustive list of business architecture elements from multiple EA frameworks and defining their relationships.

We find that Zachman framework is exhaustive in nature with multiple perspectives as well as abstractions. Many of the classic EA frameworks focus on software architecture and often neglect the first two rows of Zachman, that is, typically the business architecture [5]. However, we compare and contrast business architecture elements from eight other frameworks viz., TOGAF, FEAF, McDavid's Business Architecture description, Strategic Architectural model, Crompton Architectural Metamodel, Avancier Methodology, Japanese Government Enterprise Architecture framework

and ArchiMate EA Metamodel with Zachman framework, to come up with our unified meta model for defining business architecture.

APPROACH FOR COMPARISON

There are few approaches for comparison of EA frameworks that are available in literature. Goethals [5] differentiates between two classes of frameworks namely, Classic Enterprise Architecture Frameworks and Federated Enterprise Architecture Frameworks. Urbaczewski and Mrdalj compare EA frameworks across multiple views and abstractions of Zachman framework [6]. They also provide a comparison of frameworks on the basis of software development lifecycle phases. Tang et al., in order to analyze frameworks, have grouped fundamental elements into goals, inputs and outputs [7]. Based on support for these elements, Tang et al., have classified frameworks as Software Architecture Framework and Enterprise Architecture Framework. Sessions [8], having compared four EA frameworks including Zachman, TOGAF, FEAF and Gartner concludes that these methodologies can be seen as complementing each other and for many organizations, the best choice is all of these methodologies, blended together in a way that works well within an organization's constraints. Deborah Weiss explores multiple EA frameworks (including TOGAF & Zachman) and their approaches to develop the business context, analyze business vision and strategy, environmental trends and their implications on the enterprise and concludes that there is no one framework that can provide all answers [9]. Regardless of which EA framework the organization subscribes to, it needs to review the various frameworks and adopt the concepts to create a process for developing business context for its EA programs.

Abstraction Framework	DATA What	FUNCTION How	NETWORK Where	PEOPLE Who	TIME When	MOTIVATION Why
TOGAF	<ul style="list-style-type: none"> Business Services Business Data Model 	<ul style="list-style-type: none"> Business Processes Business Functions 	<ul style="list-style-type: none"> Locations 	<ul style="list-style-type: none"> Business Roles Users Organization Unit 	NA	<ul style="list-style-type: none"> Business Goals and Objectives
FEAF	<ul style="list-style-type: none"> List of Business Objects Semantic Model 	<ul style="list-style-type: none"> List of Business Processes Business Process Model 	<ul style="list-style-type: none"> List of Business Locations Business Logistics System 	NA	NA	NA
Business Concepts Architecture	<ul style="list-style-type: none"> Business Outcome Business Resource 	<ul style="list-style-type: none"> Business Behavior Business Function 	<ul style="list-style-type: none"> Business Location 	<ul style="list-style-type: none"> Business Role Player Business Commitment 	NA	<ul style="list-style-type: none"> Business Situation Business Purpose
SAM	<ul style="list-style-type: none"> Business Component 	<ul style="list-style-type: none"> Business Process Business Function 	NA	<ul style="list-style-type: none"> Organization 	<ul style="list-style-type: none"> Program/Project 	<ul style="list-style-type: none"> Objective or Goal
CAM	<ul style="list-style-type: none"> Product 	<ul style="list-style-type: none"> Business Process Function 	<ul style="list-style-type: none"> Location 	<ul style="list-style-type: none"> User Business Interested Party Supplier IT System 	NA	<ul style="list-style-type: none"> Goal Objective Critical Success Factor (CSF) Marketing Aim Critical Assumption Standard
AM	<ul style="list-style-type: none"> Scope: Actors, Inputs and Outputs 	<ul style="list-style-type: none"> Processes 	<ul style="list-style-type: none"> Location 	<ul style="list-style-type: none"> Organizations Sponsors and Stakeholders 	<ul style="list-style-type: none"> Plans 	<ul style="list-style-type: none"> Inputs: Goals, Requirements and Constraints
JEA	<ul style="list-style-type: none"> Transient Information Information Flow 	<ul style="list-style-type: none"> Business Processing Workflow Business Function 	<ul style="list-style-type: none"> Boundary Environment 	<ul style="list-style-type: none"> Resource Business Operation 	NA	<ul style="list-style-type: none"> Business Objective Business Policy
ArchiMate	<ul style="list-style-type: none"> Product Value Business Object Representation Meaning 	<ul style="list-style-type: none"> Business Process Business Function Business Interaction 	<ul style="list-style-type: none"> Business Interface 	<ul style="list-style-type: none"> Business Role Business Actor Organizational Service Contract Business Collaboration 	<ul style="list-style-type: none"> Business Events 	NA
Composite EBA Framework	<ul style="list-style-type: none"> Business Offering Business Information Business Resource 	<ul style="list-style-type: none"> Business Behavior Business Functions 	<ul style="list-style-type: none"> Business Location 	<ul style="list-style-type: none"> Business Role Player Business Commitment Business Organization Unit 	<ul style="list-style-type: none"> Business Events 	<ul style="list-style-type: none"> Business Motivation Business Situation

Table 1: EBA Elements' Comparison Matrix

Source: Infosys Research

The approach we have adopted for comparison of EA frameworks for EBA components is two fold: i) identify business architecture elements from direct sources of the framework or from previous research on these frameworks that are available as references, ii) populate these exhaustive list of elements into different abstractions (what, how, where, who, when and why) of Zachman framework

and compare them in order to derive the comprehensive list of elements for each abstraction. The intended result of the comparison exercise is to identify a unified list of elements and develop a meta-model where the relationships among these elements are established. In Table 1 abstraction frameworks alongwith their EBA elements identified are detailed in the comparison matrix.

The Open Group Architectural Framework (TOGAF): TOGAF's Architectural Development Methodology (ADM) prescribes certain business architecture building blocks or architectural models like – organization structure, business goals and objectives, business functions, business services, business processes, business roles, business data model and correlation of organizations and functions [1]. Even though TOGAF does not offer a meta-model of its ingredients, the abstracted business architecture elements or objects include *users* and *locations* [10]. There are nine EBA *objects* defined by or abstracted from TOGAF, excluding *correlation of organizations and functions* which is more to relate business functions to organizational units in the form of a matrix report.

architecture element includes list of business objects, list of business processes, business locations of *planner view*; semantic model, business process model and business logistics model of *owner view*. To sum up, there are six elements or 'list of things' that form the building blocks for business architecture as described in FEAF.

Standard for Business Architecture Description: McDavid in his classic IBM paper, 'A Standard for Business Architecture Description,' details that a set of generic *concepts* and their inter-relationships organize business information content in terms of requirements of the business, the boundary of the business and the business as a system for delivery of value [12]. He contends

Each Enterprise Architecture framework has a set of building blocks called elements that help model business in a structured way leading to effective business analysis

Federal Enterprise Architecture Framework (FEAF): FEAF uses a drill down process resulting in a four-level EA framework. Each level provides an understanding or frame of reference for the next as well for level IV which is the logical structure for classifying and organizing the descriptive representations of the Federal Enterprise. Zachman Framework and Spewak's Enterprise Architecture Planning (EAP) are the key elements in defining level IV and this level incorporates the five perspective rows and first three columns of Zachman [11]. The *planner* and *owner* rows focus on the business architecture definition and documentation. The business

that a set of standard business concepts can organize particular knowledge about any given enterprise. This organized business knowledge gives rise to requirements for enterprise business information systems. These requirements can be satisfied in two general ways, one by the traditional custom development approach and the other by matching patterns of requirements to patterns of existing assets. In his paper, McDavid provides a meta-model of *business concepts* that he calls *business concepts architecture*, a semantic framework relating common business concerns. There are nine concepts— business situation, business purpose and business outcome are the

drivers of the business; business role player and business commitment form business boundaries and business function, business behavior, business resource and business location form business delivery systems.

Strategic Architectural Model (SAM): Developed by Bob Jarvis, SAM is a more generic EA methodology and a specific version of SAM is the Microsoft Architecture Paradigm (MAP). SAM is based on a meta-model based approach and there are ten *structures* that form the ingredients of SAM [10]. The *structures* related to business architecture based on our conclusion include objective or goal, organization, business function, business process, business component and programme or project.

Avancier Methodology (AM): Developed by Graham Berrisford, Avancier Methodology for EA does not offer a definitive meta-model but guides in defining a meta-model [10]. AM details that an EA methodology involves a process and a product. The process involves three steps viz., scope architecture deliverables, define baseline architecture and define target architecture. The product is a model that describes an enterprise and this model defines a meta-model in itself. AM lists twelve areas of concerns within which the concerns related to business architecture include – *inputs*: goals, requirements and constraints, organizations, sponsors and stakeholders, locations; *scope*: actors, inputs and outputs, processes and plans.

Frameworks like SAM, CAM, AM and JEA define business architecture elements that can be modeled for conducting effective business analysis

Crompton Architectural Meta-model (CAM): Allistar Crompton developed CAM to capture the essence of his experience in a series of EA assignments. CAM is designated to be ready-to-go model and describes an extensive meta-model [10]. CAM is based on twenty eight ingredients known as *terms* that are most often needed to define practical EA assignments. The *terms* related to business architecture based on our conclusion include goal, objective, critical assumption, critical success factor, marketing aim, standard, user, business, interested party, supplier, IT systems, location, product, business process and function.

Japanese Government Enterprise Architecture (JEA): Hashimoto et al., in their paper ‘Case study on RM-ODP and Enterprise Architecture’, compare and contrast elements of EA between RM-ODP and Japanese Government Enterprise Architecture (JEA) [13]. In order to compare the interoperability between RM-ODP and JEA, the authors have derived the meta-model elements of JEA from JEA guideline book. There are four perspectives in JEA viz., business, data, application and technical. The authors list eleven business perspective *concepts* of JEA including business policy, business objective, business function, business operation, boundary,

environment, transient information, information flow, workflow, business processing and resource.

ArchiMate Enterprise Architecture Meta-model:

ArchiMate is an open and independent modeling language for enterprise architecture, supported by different tool vendors and consulting firms. ArchiMate provides instruments to support enterprise architects in describing, analyzing and visualizing the relationships among business domains in an unambiguous way. The business layer meta-model of ArchiMate shows the *concepts* and the predefined relationships that can be used to connect them. There are fifteen business architecture elements described in the

business architecture elements of Zachman framework across the multiple abstractions of Zachman viz., what (data), how (function), where (network), who (people), when (time) and why (motivation) [15]. Assessing the EBA elements from multiple abstractions suggested ways to analyze, define and finalize a unified list of elements is gathered in Table 1 on page 7.

- **What (Data) Abstraction:** Products/ services and information elements related to business is what multiple frameworks list out for this abstraction. Zachman calls it as ‘list of things important to the business’ – the understanding of and dealing with enterprise’s data. McDavid

A unified list of EBA elements is essential for categorizing elements and defining a meta-model to create a business architecture blueprint

ArchiMate business layer meta-model under three categories, viz., *structural concepts* comprising of business actors, business objects, business role, business collaboration and business interfaces; *behavioral concepts* comprising organizational service, business behavior or business interactions, business function, business process and business events; and *informational concepts* comprising representation, meaning, product, contract and value [14].

COMPOSITE EBA FRAMEWORK ELEMENTS

Having listed the comprehensive set of business architecture elements from the eight frameworks discussed above, we compare them with the

bundles business information as part of *business resources* [12]. Business resources include all those things that are required by a business to sustain its processes and create its outcomes. Business resources fall under five categories – physical things, energy, monetary value, information resources and various kinds of capabilities. It can be noted that three elements comprehensively (what we term as *attributes* here onwards) cover this abstraction – *business offering* or products and services offered by business; *business information* or information/data flow in the business and *business resources*

or things that are required to sustain business and create outcome. In order to differentiate business information and model it exhaustively, we have listed it as a separate attribute. So, business resources shall include all the things that are required to sustain business sans information.

- **How (Function) Abstraction:** *Business processes* and *business functions* are the elements that multiple frameworks list out for this abstraction. Zachman terms it as *Business Process Models* or the process of translating the mission of the enterprise into successively more

location of business, comprehensively covers this abstraction.

- **Who (People) Abstraction:** The comprehensive list of attributes here includes *business role player* – actors/users who perform business behavior; *business commitment* – binding of business with external and internal organization; and *business organization unit* – how the organization is structured and list of things related to it. Zachman framework includes list of organizations important to the business, roles and organization unit – describing who is involved in the business and an introduction of new

Elementary questions like what, how, where, who, when and why form the basis of multiple abstractions of the Zachman framework

detailed definitions of its operations. Two attributes comprehensively cover this abstraction, *business behavior* – business processes that are aligned to achieve business goals and *business functions* – the virtual and idealized organization within the business.

- **Where (Network) Abstraction:** *Where* abstraction is all about the business locations and Zachman calls it as ‘list of locations business operates’ – the geographical distribution of the enterprise’s activities. One attribute i.e., *business locations* – physical and logical

technology. The attribute additional to Zachman framework is *business commitment* that binds business entities.

- **When (Time) Abstraction:** Multiple frameworks do not detail much for *when* abstraction. ArchiMate terms its element as *business event*. SAM and AM term it as projects and *plans* respectively. Zachman calls it as ‘list of events significant to business’ describing the effect of time on business. One attribute viz., *business events* or things happening internally or externally, affects business behavior and comprehensively covers this abstraction.

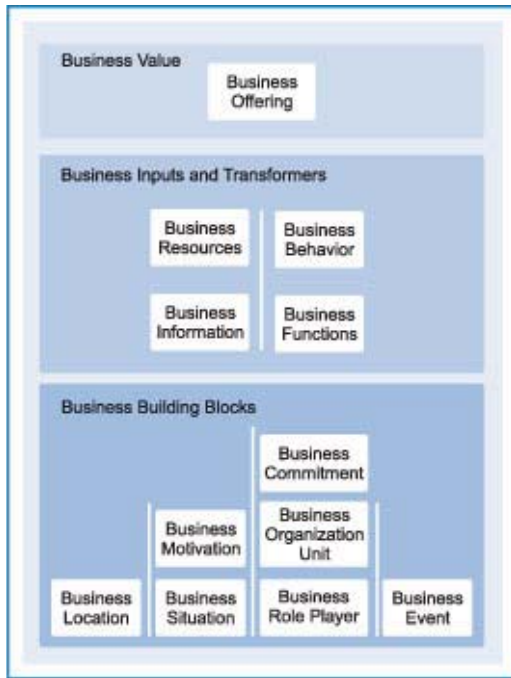


Figure 1: Composite EBA Framework Attribute Category
Source: Infosys Analysis

- Why (Motivation) Abstraction:** Business objectives and goals are the elements that are common across multiple frameworks. Zachman calls it as ‘list of business goals/strategy’ – detailing translation of goals and strategies into specific ends and means. There are two aspects to be covered in this abstraction, business objectives/goals that affect business internally and business situations that are outside business boundary and affect business externally. We chose to go for a more abstract attribute called *business motivation* – internal factors that motivate to establish business plans, along with *business situation* – external forces that act upon the business, taking into consideration both internal and external factors that affect business. We

include organizational values, culture and guiding principles as part of *business situation* though they are internal to the organization. *Business motivation* constitutes of *ends-means* concept and comprises of vision, goals and objectives as *ends* and mission, strategy and tactics as *means* comprehensively covering *why* abstraction [16].

To sum it up, the Composite EBA framework comprises of 12 attributes in three broad categories - *business building blocks* comprise of business location, business role player, business commitment, business organization unit, business events, business motivation and business situation; *business inputs and transformers* comprise of business information, business resource, business behavior, business functions; and *business value* comprises of business offering [Fig. 1]. Tables 2 and 3 provide a brief description of the *attribute categories* and *attributes* in a snapshot.

COMPOSITE EBA FRAMEWORK METAMODEL

Having finalized the attributes comprising EBA from multiple frameworks, we develop a meta-model by establishing relationships among the attributes. Of the eight frameworks compared, we find that the meta-model based *business concepts architecture* provided by McDavid [12] and ArchiMate Business Layer meta-model [14] are exhaustive in nature as relationships among elements are established. The CEBA meta-model is more closer to McDavid’s business concept architecture definition as compared to Archimate Business Layer meta-model since McDavid’s is more abstract in nature.

Let us understand in brief three of these attributes in the way they are structured and the

Attribute Category	Description
Business Building Blocks	This category includes attributes related to why the business exists, the business situations that are external to the organization, commitments that drive business, role players, the business organization structure, geographical locations and important events that drive business. All the attributes that are necessary to provide the basement for business are included in this category.
Business Inputs and Transformers	This includes attributes related to the resources that are necessary for the business, information that are required for the business, functions that are organized to carry on the business and the business processes that generate business products and services. All the attributes that provide input and transform input to create value to the customer are included in this category.
Business Value	This includes attributes like business offerings of the enterprise that satisfy the needs of the customer and provide value to the customers.

Table 2: Attribute Category

Source: Infosys Research

method that can be utilized to construct these attributes. This will help one understand how the relationships are established at this abstracted level.

Business situation is affected by three major factors – external, internal and current business situation. External factors can be political, economic, social, technological, legal or environmental (PESTLE) factors. Internal

factors include business policies and business standards of the organization. Internal strengths and weaknesses and external opportunities and threats form the current business situation. The methods that can be utilized to construct business situation models include PESTLE analysis, internal situation analysis and SWOT analysis.

Business motivation includes ingredients that define why the business exists – vision,

Attribute Category	Attribute	Description
Business Building Blocks	Business Motivation	Details why the business exists; factors that motivate to establish business plans and take advantage of the business situation
	Business Situation	Sources of requirements that are placed on the business externally; also includes internal aspects like values, culture and guiding principles of the organization that affects business decisions
	Business Commitment	Interaction that binds the organization and external/ internal entities
	Business Role Player	Actor who performs the business behavior
	Business Organization Unit	Details how organization is physically structured with the roles and responsibilities of role-players
	Business Location	Physical and logical geographical spread of the business
	Business Event	List of important happenings to the business; things happening internally or externally that affects business behavior
Business Inputs and Transformers	Business Resources	Things that are required to sustain business and create outcome - physical, energy, monetary value and capabilities
	Business Information	Information or data flow in the organization
	Business Behavior	Business processes that are aligned to achieve a business goal
	Business Functions	Virtual and idealized organization within the business
Business Value	Business Offering	Products and Services of the enterprise that creates value to the customer

Table 3: Attribute Description

Source: Infosys Research

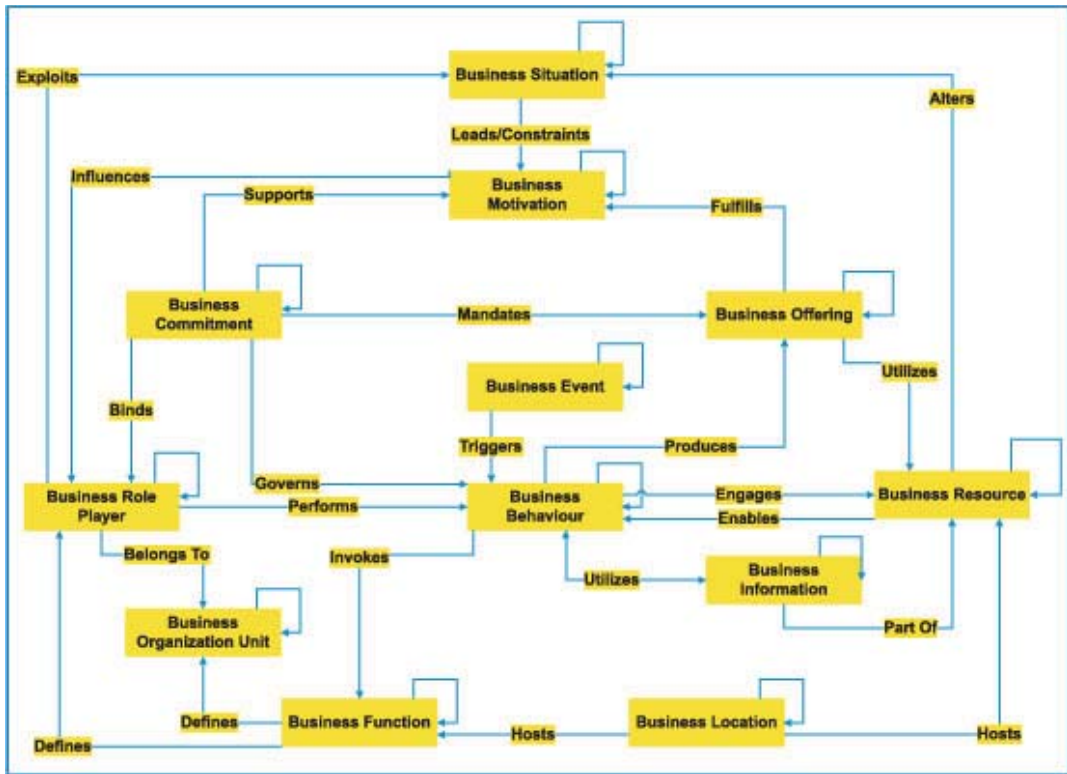


Figure 2: Composite EBA Meta Model

Source: Infosys Research

mission, goals, strategies, objectives and key performance indicators of the organization. The methods that can be utilized to construct business motivation include Business Motivation Model - a scheme or structure for developing, communicating and managing business plans in an organized manner utilizing the ends-means concept - from Business Rules Group, Balanced Scorecard for Goal Modeling and Porter's Five Forces Model and Value Chain Analysis for defining business strategy.

Business behavior includes the ingredients that define the business processes for the organization - the value stream, high level business processes and sub-processes, business workflow, activities, business participants (business organization unit, department, business

role player and systems) and the output entity. The methods that can be utilized to construct business behavior include value chain analysis, business context diagrams, value stream analysis, process modeling and analysis techniques.

Similarly for all the attributes, the underlying meta-model ingredients and the methods to construct them are utilized to establish the relationships in the CEBA Meta-model. The relationships established here are also based on our understanding from multiple frameworks that we have covered here and the reader/user is advised that these relationships can vary according to usage scenario in practical EA assignments and the meta-model can be tailored according to the context. Figure 2 depicts a network representation of our meta-model.


Establishing relationships between attributes help in developing a model that in turn helps in a structural view of a complex enterprise that is made up of multiple ingredients. Also, the attributes are recursive or hierarchical in nature and we can have a meta-model to define these attributes as discussed earlier in the paper.

CONCLUSION

Composite EBA framework developed in this paper is comprehensive with inputs from multiple frameworks and comprises of 12 attributes that detail the constituents of EBA. A meta-model based approach is advisable as an EBA can be defined more methodically and relationships can be established more effectively. The major contribution of this research is in defining business architecture in a structured manner, as the building blocks have been established now and can lead to effective business analysis and business architecture development. Also, the research has value in terms of contrasting elements across various *abstractions* of Zachman framework. As we find that the EBA attributes are highly abstract in nature and can be further decomposed hierarchically, the scope of work extends to defining deliverables and artifacts that need to be generated for each of these attributes of EBA.

REFERENCES

1. The Open Group Architecture Framework, v8.1.1, Enterprise Edition. Available at <http://www.opengroup.org/architecture/togaf8-doc/arch/>
2. Ken Orr, Business Architecture: Linking Business, Data and Technology, Cutter Consortium Enterprise Architecture Executive Report, Vol 10, No 2, 2007
3. Ralph Whittle and Conrad Myrick, Enterprise Business Architecture: The Formal Link between Strategy and Results, Auerbach Publications, CRC Press, USA, 2004
4. Ken Orr, Extending Zachman: Enterprise Architecture and Strategic IT Planning, Cutter Consortium Business-IT Strategies Executive Report, Vol 7, No 4, 2004
5. Frank Goethals, *An Overview of Enterprise Architecture Framework Deliverables* in Enterprise Architecture-An Introduction, ICFAI University Press, 2006. Also available at <http://www.econ.kuleuven.be/leerstoele/sap/downloads/Goethals%20Overview%20existing%20frameworks.pdf>
6. Lise Ubraczewski and Stevan Mrdalj, A Comparison of Enterprise Architecture Frameworks, Issues in Information Systems, Vol 7, No 2, 2006
7. Antony Tang et al., A Comparative Analysis of Architecture Frameworks, Available at <http://www.it.swin.edu.au/centres/TechnicalReports/2004/SUTIT-TR2004.01.pdf>
8. Roger Sessions, Comparison of the Top Four Enterprise Architecture Methodologies. Available at <http://www.objectwatch.com/whitepapers/4EAComparison.pdf>
9. Deborah Weiss, Enterprise Architecture Framework - Approaches to Business Context, Gartner, 2006. Available on www.gartner.com
10. G Berrisford, Modeling the Enterprise (Enterprise Architecture Metamodels). Available at http://grahamberrisford.bulldoghome.com/pages/grahamberrisford_bulldoghome.com/Docs/204b2%20Enterprise%20Architecture%20Meta%20Models.htm
11. Federal Enterprise Architecture

- Framework, v1.1, CIO Council, 1999
12. D W McDavid, A Standard for Business Architecture Description, IBM Systems Journal, Vol 38, No 1, 1999. Available at <http://www.research.ibm.com/journal/sj/381/mcdavid.html>
 13. Daisuke Hashimoto et al., Case Study on RM-ODP and Enterprise Architecture, Eleventh International IEEE EDOC Conference Workshop, 2007. Available at http://www.inf.ufes.br/~jpalmeida/wodpec2007/cameraready/WODPEC_Hashimoto.pdf
 14. Concepts for Architectural Description, ArchiMate Deliverable 2.2.1 v4.0. Available at <https://doc.telin.nl/dsweb/Get/Document-29421/>
 15. J Zachman, A Framework for Information Systems Architecture, IBM Systems Journal, Vol 26, No 3, 1987
 16. Organizing Business Plans, Business Rules Group. Available at http://www.businessrulesgroup.org/second_paper/BRG-BRMM.pdf. 
-

Functional Storyboards: Requirement Elicitation for COTS Evaluation

By Nandini Toley

While business requirements and use cases describe the plot, functional storyboards present the real script

The process involved in initial requirements elicitation and specification during the Commercial-Off-the-Shelf (COTS) package evaluation can be challenging. This paper seeks to highlight the importance of identifying critical business workflows prior to package selection. Since no product can ever be a perfect-fit to the organization's needs, customizations are inevitable. It becomes necessary to understand the degree of customization before selecting the product. Any customization to the vendor product comes at a tangible and significant cost. Thus, these are always the primary focus of the organization. This paper is based on the premise that functional storyboards developed before the COTS evaluation phase can not only provide a better *visual* understanding of the practical workflows in the existing system but may also help organizations appreciate the superior workflows that the COTS packages have to offer, thus minimizing workflow customizations.

COTS packages are pluggable components that provide end-to-end functionality for a given business process and their implementations have an edge over custom developed applications due

to faster development, reduced time-to-market, better quality and use of best-in-class business and technology frameworks.

The journey from legacy applications to custom developed technically-rich applications to COTS packages, needs to be understood first before we discuss it further. Legacy applications have been at the heart of every organization at some point in time. Changing business needs, significant advances in technology and increased focus on quality and productivity of service have forced organizations to revamp their core systems. This is where the *buy or build* decision gains importance.

The best business case for considering a COTS package is when an organization's fundamental need to re-engineer its existing business processes creates an opportunity to evaluate the best-in-breed COTS package options. This should be the most justified reason for an organization to consider purchasing a COTS package since it continuously evolves to keep abreast with the latest industry requirements. Even before COTS evaluation begins, organizations must prepare their business stakeholders to

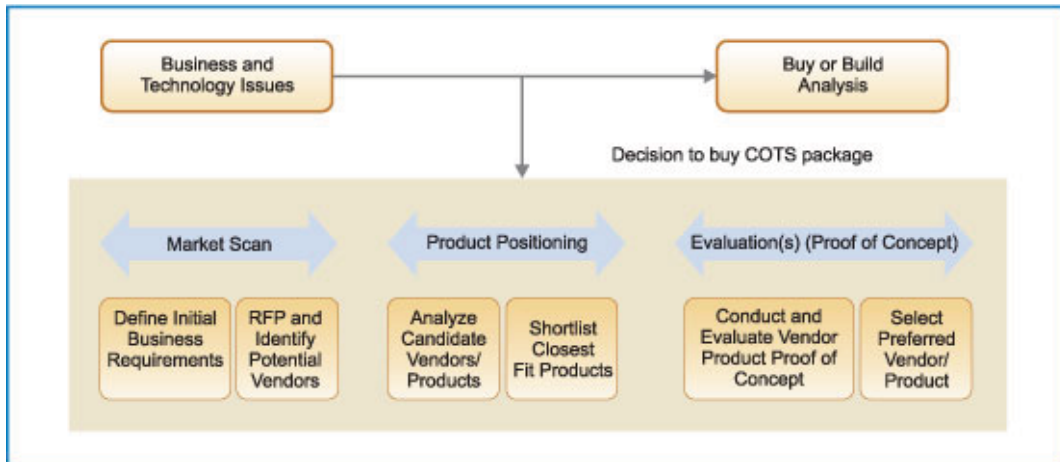


Figure 1: Vendor Product Evaluation and Selection Workflow **Source:** Infosys Research

appreciate the fact that they will be purchasing a business process as much as a COTS product.

The caveat here is that every organization has some critical business process workflows that are non-negotiable and cannot be compromised at the cost of a new COTS product. A *lessons learnt* published by SEI, Carnegie Mellon states that while purchasing a COTS application, a well-planned balancing act becomes essential to understand business processes that are non-negotiable [1]. The challenges around identifying critical non-negotiable workflow requirements are to be addressed for an effective COTS evaluation.

PITFALLS DURING COTS EVALUATION

A variety of COTS product evaluation frameworks exist in the software industry but they all share some common attributes. Figure 1 provides a generic description of these attributes. In theory, what may be lauded as a popular framework, often fails to deliver during practical product evaluations. Though the framework itself is considered to be a very robust one, many organizations adopt the COTS paradigm assuming that there could be absolutely no business process

change, be it for better or worse. Couple that with a weak requirements engineering phase during package evaluation and you have the perfect recipe for the failure of this framework.

Out of the many COTS implementations that we have been involved in, this paper is based on a case study related to participation in evaluation, selection and implementation of a COTS package for a client who intended to purchase enrollment, billing and accounts receivable functionality. In this case, the need to buy a COTS package arose as a result of the sunset of the technology platform that used to support the existing legacy system. The build versus buy decision was comparatively expedited since the replacement was needed to be put in place within a short span of time while the old platform support still offered support.

The existing system was a desktop application, used only by the internal business teams to perform day-to-day activities like enrolment processing, error resolution, invoice generation/tracking/ printing, payment reconciliation, reporting and balancing general ledger, member call tracking, etc. A variety of business processes

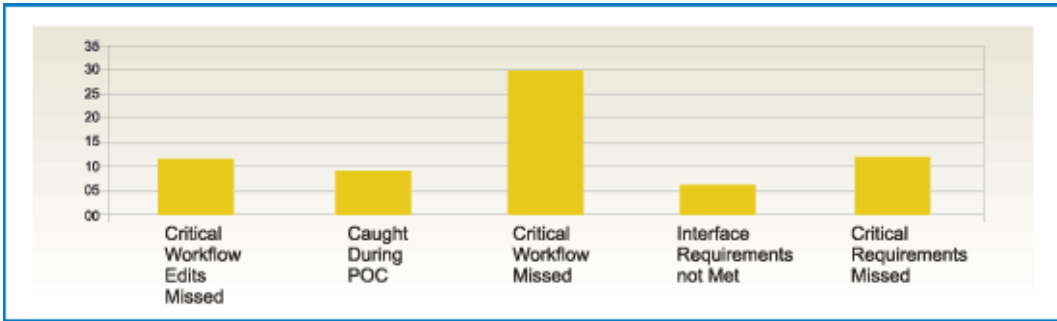


Figure 2: COTS Gaps/Customizations Analysis

Source: Infosys Research

required manual user intervention, for instance, enrollment verification, member maintenance, processing unassigned/unidentified payment records, moving misapplied payments across accounts, etc. However, many critical business processes like cash application, dunning delinquent members, generating enrolment/member maintenance edits, reconciliation with general ledger, etc. were automated. As is evident, user actions and automated system interfaces were equally driving the manner in which work flowed in the organization. Our worst enemy turned out to be the fact that these business workflows and requirements had no existing documentation, since they were business-owned applications.

The most interesting phase of our project began with elicitation, definition and specification of initial business requirements. Iterative dialogues were conducted with the business stakeholders to understand the business needs. High level business requirements were created on the basis of these discussions that were used as the blueprints for vendor evaluations. These requirements were primarily meant to evaluate the *ability* of a vendor product to perform all the business defined tasks and hence meet the business requirements.

Eventually, a couple of vendors were short-listed for lab demonstrations. The high

level requirements were detailed into use cases which were then developed into real-world business scenarios. Using these scenarios, the vendor products were tested in a hands-on environment by actual business users.

It is a fact that the proof of concept (POC) exercise was able to satisfy majority of the scenarios as desired, but in retrospect (after purchasing the product), it is clear that the evaluation failed to uncover many gaps that were primarily related to multiple scenarios, tied together through some critical business workflows. Some gaps needed major customizations in the vendor product that come at a substantial cost to the organization. Ironically, majority of the gaps required a change in the existing business process workflow, in some cases, even necessitating change of an automated process in the existing system to a manual one in the COTS world. Figure 2 shows a graph of the gaps uncovered (by category) after vendor product selection that required either customization in the vendor package or in the business workflows.

Let us analyze what might have led to the selection of a COTS misfit resulting in the dissatisfaction of the business stakeholders despite investing considerable time and effort in requirements elicitation. The deductions from

the case study stated in the previous section indicated that majority of the gaps were due to not testing the product against critical business workflows during the POC. So, is it that the initial business requirements along with use cases do not automatically cover all business workflows?

What are workflows? Workflows are a way of defining business processes. They define the steps followed in a process, specify which business user carries out each step and state the inputs and outputs of each step [2]. COTS package conformance to a critical business process workflow can be validated by ensuring that the following criteria are met:

- Qualitative criteria that ensure that the workflow does what it is meant to do
- Quantitative performance criteria that ensure that work is carried out at minimum cost in minimum time.

Functional requirements and use cases define what business requirements and rules are expected to be satisfied by a system and the sequence of interaction of each actor with the system. But in a system with little or no documentation and high user-interactivity, we must understand how work really 'flows' through the system and consider whether the 'success' of each work transaction depends on the end-user being led through a series of steps. We also need to identify the areas of impact if such a flow is not maintained.

PRE-SELECTION REQUIREMENTS ENGINEERING APPROACH

An organization's strategy should be to adopt the best practices of the COTS package, yet retain the critical business workflows that exist for certain key functionalities, specifically related to financial rules and mandated regulations.

Before we can begin business workflow requirements elicitation through functional storyboarding, sufficient representation must be requested from each business area that will be impacted by the COTS package. The success of this methodology highly depends on involving business users who work at the grass-root level as well as the business managers who understand the bigger picture.

For each business function for which we wish to create storyboards, the following steps can be used to gather and define workflow requirements:

- Decide on the important aspects of the business function that you are prototyping
- Identify the major steps involved in conducting the functionality
- Design the screen mock-ups required to complete each step. The screens should only capture elements that represent business critical information
- Build the individual frames after dialogue with business
- Assemble the frames in a sequence that represents the practical flow of work through the system.
- Capture user feedback for each individual frame as well as the complete storyboard.

One point to remember is that the storyboards (and the use cases) are not the model of the solution, but a mechanism to explore the interfaces and navigation that the solution must address, leading to concise statements of the requirements and a common understanding of the application.

Let us try to apply these steps to the refund processing business function of our case study. The functional requirements for the same

could be 'system must have the ability to identify groups eligible for refunds based on business defined criteria (for e.g. group termed with credit balance),' 'system must allow authorized users to approve refunds identified by the system' and 'system must be able to process refunds in different modes like EFT and check.'

A storyboard can help narrate a story from these requirements. Here is an example of the steps involved in the refund processing storyboard that extends the functional requirements mentioned above:

- System identifies groups that qualify for refunds based on the business defined criterion in the functional requirements
- The business users responsible for refund approvals are notified that the refund transactions are available for review
- Authorized business user(s) log(s) into the relevant screen to pull up all refund eligible groups ready for review
- Each reviewer must be able to filter available records using group ID, authorizer ID, refund reason code, groups in a particular state, etc.
- Each reviewer must be able to change the refund reason code or the status of the refund
- All changes made by reviewers are tracked by the system using user ID and date of change
- Approved refund requests are sent to Accounts Payable system to process the refunds
- Letters containing details of the approved refund transaction are automatically sent by the system to the members' address
- System automatically sends a file to the General Ledger containing details of the total refund dollars for the approved refund requests.

On close inspection one can notice that this storyboard helps us in better comprehending how the business functions map to the system functions in the existing application. Unless we have this knowledge, how can we understand whether the processes in each COTS package are superior or inferior to the existing system? The storyboard also helps us understand that the user may want to filter on the available records as well as change the status of the same. We also get insights into the steps in the workflow that are currently automated versus the ones that needed manual intervention. An added advantage of this methodology is that, while reviewing such detailed storyboards with the business team which also presents a UI mock-up, we may be able to uncover additional functional requirements. For instance, the business user may tell us that since we are already tracking status updates on every refund transaction, the business users would also like to receive e-mail status updates on the refunds approved by them.

If a particular event in a business workflow requires a default action to be taken automatically without manual intervention, this becomes an important workflow requirement. On a similar note, if the default action needs to be over-ridden manually by a user authorized with the correct privileges, that is also a significant workflow requirement. Such requirements are also highlighted through storyboards.

Once functional storyboards are developed for all the critical business functions, the workflows involved in the same must be categorized into automated and manual workflow categories. This is necessary to understand scenarios in which an automated workflow in the current system needs manual workarounds in the COTS package being evaluated.

While reviewing functional storyboards with the business, the project team should also try to document the business systems that are directly involved in each workflow as well as the business systems that get indirect feeds as a result of success or failure of the workflow. This will help in easy understanding of the business impact, if the workflow requirement validation against the COTS product confirms that there is a gap.

Once such extensive groundwork is done for the critical business process workflow requirements, the project team should be in a better position to move into the POC lab. The initial standalone business requirements, use cases as well as the functional storyboards should form a solid foundation for POC evaluation.


CONCLUSION

The COTS philosophy is rapidly gaining popularity with businesses around the world. With this paradigm shift in software development, there is a need to improvise the evaluation processes as well. As this paper reiterates time and again, the underlying principle for exploring COTS options must be an organization's inherent need to re-engineer their business processes. But at the same time, there are some very critical business processes that need to be treated with due respect while COTS package evaluation is being done.

The cost of changing critical business workflows is intangible to begin with and hence is often neglected. But in the final diagnosis, tailoring critical business workflows to fit the COTS package framework may mean spending millions in employing additional staff, training resources, uncovering cascaded impacts to other systems, fixing defects, quality assurance and

most importantly, readjusting customer service level agreements. Hence one can conclude that these costs definitely become tangible in the long run. In the COTS evaluation philosophy, functional storyboards could prove to be effective requirement elicitation tools that help us understand 'where we stand' with our existing business processes. After all, isn't that an essential foundation to find out 'where we eventually intend to be?'

REFERENCES

1. Business Processes, Lessons Learned, Software Engineering Institute Website. Available at <http://www.sei.cmu.edu/cbs/lessons/business-process/lessons.htm>
2. Business Process Workflows. Available at <http://www-staff.it.uts.edu.au/~igorh/cscw/busnets/workflow.htm>
3. Stephen E Cross, Towards Component-based Systems, 2000. Available at <http://sunset.usc.edu/Activities/feb6-9-01/Presentations/Steve%20Cross-Towards%20Component-based%20Systems.ppt>
5. Venkatesh Thyagarajan and Shyamala C Sivakumar, Critical 'Human' Success Factors for Purchasing Commercial Off the Shelf Software, The Workplace Review, October, 2005. Available at <http://www.stmarys.ca/academic/sobey/workplacereview/october2005/WPR3wpBenchmarks.pdf>
6. Craig Borysowich, Internet: Workflow and User Interfaces, 2007. Available at <http://blogs.ittoolbox.com/eai/implementation/archives/internet-workflow-user-interfaces-21410>. 

Improving Requirements Accuracy through the use of General Semantics

By Badhri Narayanan C.S.

Applying General Semantics during requirements elicitation can help increase the accuracy of requirements

The quality of Requirements Engineering is recognized as a critical factor in determining the success of software projects. Despite the numerous tools and techniques developed over the past two decades in the software industry to improve Requirements Engineering, poor requirements still figure among the top 3 reasons for project failures and impairments [1]. The primary reason for the intractability of Requirements Engineering is that unlike most other phases in the development of a software product, requirements elicitation relies more on human communication and perception, and less on technical processes. A major cause of poor requirements is the inability of users to understand and articulate their true business needs, leading to poor requirements elicitation.

Requirements Elicitation is the first step in the Requirements Engineering phase and involves discovering the requirements of a system from prospective users and other stakeholders. During requirements elicitation, business analysts rely on users to provide them

with the correct requirements. But in many cases, the users do not have enough clarity on their requirements. So even when a software is developed to match the requirements provided by the users, the users are often dissatisfied with the final product. This gap between the requirements stated by the user and the actual business needs of the user creates a major hurdle in collecting correct and accurate requirements.

It is in this context that the usage of the tools of General Semantics can improve the requirements elicitation process.

GENERAL SEMANTICS: AN INTRODUCTION

General Semantics can be described as *a systematic methodology for individuals to understand how they relate to the world around them, how they react to this world, how they react to their reactions, and how they adjust their behavior accordingly* [7].

The practice of General Semantics helps reduce the gap between our perceptions and reality. As a discipline, it was developed by Alfred

Korzybski during 1919 -1933. It covers a wide area of interest and a range of approaches and principles. But as requirements engineers, the aspect of General Semantics that is of particular interest to us is the *consciousness of abstracting*.

Consciousness of abstracting is a concept introduced by Korzybski in his seminal work 'Science and Sanity' [2]. It deals with the general human tendency to abstract every event and object, especially when stating it in the natural language. General Semantics contends that it is important to realize that an event or object is being abstracted in order to avoid false identification and ensure that perceptions match reality. In the next section, we will discuss why and how this concept can make a critical difference in improving the accuracy in eliciting requirements.

HOW ABSTRACTION HINDERS REQUIREMENTS ELICITATION

General Semantics reasons that every natural language statement defining an object or an event is at a certain level of abstraction and does not define the actual thing. The level of abstraction depends upon individual experiences. The formation of such subconscious abstractions makes us see the world through the words we use rather than see the world as it actually is. It is this tendency to abstract that creates the gap between the requirements that are stated by the users and the requirements that are needed by the users.

For example, when a person says, *I need an incandescent lamp installed in the porch*, the person might be referring to the need for a source of light in the porch. But the person has subconsciously associated a source of light with an incandescent lamp and therefore interchangeably used it. In this example the user has reduced the

level of abstraction from *a source of light to a specific source of light* which in turn causes a misrepresentation of the real requirement. This misrepresentation restricts the possible solution to *an incandescent lamp*, whereas the requirement can be satisfied in many other ways, some of which can be far more efficient (such as a fluorescent lamp or a solar lamp). If the requirement was implemented as stated, the person might have been dissatisfied because of the higher power consumption and increased heat generation of the incandescent lamp.

In the software industry, the case of misrepresented requirements is often similar to the example stated above. The customer is unable to state the actual requirement because of subconscious abstractions formed and instead states the abstraction as the requirement. By being conscious of the tendency to abstract and using the techniques of General Semantics, it is possible to extract the actual requirement from the stated abstraction.

ABSTRACTION LADDER AND ITS USE IN REQUIREMENTS ELICITATION

Any requirement provided by the user is usually an abstraction of the actual business requirement present in the user's mind. The actual business requirement could either be at a higher level of abstraction or at a lower level of abstraction than the stated requirement. It is important to avoid the general tendency to assume that higher levels of abstraction mean less accurate requirements and lower levels of abstraction mean more accurate requirements. The business need determines the level of abstraction and not vice versa. As requirements engineers, we need to be conscious of the possible abstraction of the actual requirement

Level of Abstraction	Type of Words used	Example
Level 5	Words representing abstract, intangible ideas	Wealth
Level 4	Words that have a "contains a" or "is a kind of" relationship with the object or event	Farm
Level 3	Identification of category to which the object or event belongs	Livestock
Level 2	Common nouns	Cow
Level 1	Specific Individually identifiable nouns	Bessie

Table 1: Abstraction Ladder

Source: 'Thought and Action' by S.I. Hayakawa

by the user and extract the actual requirement from the stated requirement.

An effective tool to remain conscious of the possible abstraction of the requirements is the Abstraction Ladder that was popularized by S.I. Hayakawa through his book 'Language in Thought and Action' [6]. Table 1 is a representation of the abstraction ladder. Hayakawa stated that an object can be described at different levels of abstraction. His original example involved a cow named Bessie living in a farm. The cow can be described by the farmer at various levels of abstraction as shown in Table 1. The lowest rung of the ladder is the most specific description and higher rungs increase the levels of abstraction. Table 1 also describes the types of words that are commonly used at different levels of abstraction. Depending on individual perceptions there could be many more levels of abstraction and consequently many more rungs in the abstraction ladder. The abstraction ladder is an effective way to arrange the different possible abstractions of a statement into different levels and understand the abstraction level of the word or statement.

The abstraction ladder can be used during the requirements elicitation process to categorize the different levels of abstraction possible in any requirement. This is done by splitting the stated requirement into phrases and then creating higher and lower levels of abstraction for each phrase. Each phrase is placed in an abstraction ladder and each rung in the abstraction ladder is populated with different levels of abstraction of the phrase.

The abstraction ladders created for each phrase are then combined to create an abstraction matrix. The requirements engineer can then verify the requirements in each cell of the abstraction matrix, with the user to decide on the level of abstraction that most accurately defines the business requirement. This process makes it easier for both the user and the requirements engineer to distinguish the actual business requirement from the initially stated one.

EXTRACTING ACCURATE BUSINESS REQUIREMENTS

Let us take the example of a business requirement that was provided by users during the development of an enterprise-wide help desk application and see how the abstraction ladder can be used to arrive at the actual business requirement. The requirement stated by the user was:

"An email must be sent to the assigned service provider as soon as a service ticket is raised by a user."

The steps to arrive at the actual business requirement are as follows:

Step 1 – Split the Stated Requirement into Phrases: The requirement is split into a number of phrases and each phrase is placed in a different abstraction ladder. The combination of

	Ladder 1	Ladder 2	Ladder 3	Ladder 4
Higher Level of Abstraction				
Requirement as Stated	An e-mail Must be sent	To the assigned service provider	As soon as	A service ticket is raised by a user
Lower Level of Abstraction				

Table 2: Abstraction Matrix

Source: Infosys Research

these abstraction ladders creates an abstraction matrix as shown in Table 2.

Step 2 – Create Different Levels of Abstraction for Each Phrase: By using points gleaned from General Semantics, each phrase in the requirement statement can be stated at different levels of abstraction. Here are some methods that can help us translate a requirement into different abstraction levels.

- Look for Hidden Inferences: When a requirement is stated, it is usually based on a number of inferences that the users have subconsciously arrived at. These inferences need to be brought out and verified. This can be done by asking for the reason behind each specification in the requirement.

For example, a requirement like - *Only employees in the Los Angeles office should have access to the control data of the application.* When asked for the reason behind the specification of *Los Angeles office* in the requirement, the user said that the data administrator’s office was in Los Angeles and hence the requirement. The user had made two inferences in stating this requirement:

- a) All employees who access the system from the Los Angeles office would be data administrators

- b) All data administrators work only from the LA office.

This subconscious inference (association of a role with a location) had resulted in the user providing the requirement based on the location, instead of providing a requirement based on the role.

In our example of the requirement for the helpdesk application stated at the beginning of this section, the user had made a subconscious inference that formal messages must be sent through emails. This was the reason that the requirement specified that an *email must* be sent. By becoming conscious of this inference, the requirement was translated to a higher level of abstraction i.e., *a message must be sent* [Ladder 1, Table 3].

- Acknowledge Non-binary Possibilities: Formal education mostly teaches us to deal in Aristotelian logic - things are *right or wrong, black or white*. General Semantics urges us to stop thinking in terms of binary possibilities alone and acknowledge the intermediate areas that are common in the real world. A thumb rule to identify assumptions of binary options in a requirement is to look for nouns that have been categorized in some way. Consider this requirement - *Only US citizens should have access to US specific data.*

	Ladder 1	Ladder 2	Ladder 3	Ladder 4	Ladder 5
Higher Level of Abstraction	A message must be			A service ticket is raised	
Requirement as Stated	An e-mail Must be sent	To the assigned service provider	As soon as	A service ticket is raised by a user	For a non-enterprise event
Lower Level of Abstraction		To the assigned service provider and the lead service provider of the ticketed group	Within 5 minutes		
	Incorrect inference removed	Accommodate non-binary option	Replace adjectives and adverbs by standard quantifiable term	Acknowledge multi-causality	Capture exceptions

Table 3: Refined Abstraction Matrix

Source: Infosys Research

Non-US citizens must be denied access. Here the noun *citizen* has been categorized as US and Non-US. The citizenship status is seen as a binary status - US citizen or Non-US citizen. Though this classification is technically valid, it makes us lose sight of the other stages such as *permanent resident, green card holder, dual citizenship owner, etc.*, that need to be handled in real world applications.

Similarly, in our example of the requirement for a helpdesk application, we can observe an assumption of binary category. The noun *service provider* has been qualified as *assigned*. Though technically a service provider is either assigned or not assigned to a ticket, we found that in actual business there were other service providers involved in resolving a ticket, though the ticket was not *assigned* to them. There was an optional secondary service provider who was required for some tickets and a *lead* service provider who guided the *assigned* service provider. After this point was brought forth, the users wanted the email to be sent to the *lead* service provider as well. So a new lower level of abstraction

was added to accommodate this non-binary situation [Ladder 2 in Table 3].

- Eliminate Adjectives and Adverbs: Adjectives and adverbs are always subject to the interpretation of each individual. For example, in the requirement - *The background color of the screen should be dark green*, the user is referring to a color that he/she perceives and identifies with dark green. This perception of color is subjective and varies from person to person. So eliminate adjectives in the requirement statement and instead use quantifiable terms using a standard scale. This way there will be no scope for miscommunication.

In our example of the helpdesk application, the adverb used was *soon*. By replacing the word *soon* with a quantifiable term (e.g., within 5 minutes), we were able to add a lower level of abstraction for that phrase [Ladder 3, Table 3].

- Look for Multicausality: Any event can have multiple causes for its occurrence. However, the human mind tends to pay attention only to the major or most

frequent cause. When analyzing a requirement, the requirements engineer must ensure that all possible causes for an event have been considered.

Consider this example for the following requirement: *When the location of an employee changes from one state to another, the permanent address of the employee must be updated.*

Here there is an assumption of monocausality. The user has assumed that a change in location is always the result of a permanent transfer of an employee. But the change in location might be requested during a temporary deputation or a prolonged business trip to a client's place, in which case the permanent address would not change.

the deceptive nature of *allness*. Absolute words such as *every, always, never, etc.*, in a requirement statement should be viewed with suspicion and verified with all the users. In most cases there will be exceptions that go against the stated requirement.

The catch in looking for absolutism is that absolutism is not always explicit. The lack of any exception statement by itself implies absolutism. In our example of the helpdesk application, the lack of exceptions in the requirements implies that an email must be sent for *every* service ticket. Users need to be urged to think of exception scenarios. In our case, we found an exception that the service provider need not be informed if the service ticket was created for issues classified

*Absolute statements from the users have to be suspected
and exceptional statements need to be encouraged to elicit
exact requirements*

In our example of the helpdesk application, the phrase *a service ticket is raised by a user* holds an assumption that only a user can raise a ticket. But on delving further we found that some mainframe systems were equipped to create tickets automatically on failure without user intervention. So a higher level of abstraction for this phrase was added by removing the clause *by a user* [Ladder 4 in Table 3].

- Suspect Absolutism, Seek Exceptions: Absolute definitions and blanket statements rarely work in the real world. General Semantics warns us of

as *Enterprise Events*. So a new clause was added in the requirement to include this exception [Ladder 5, Table 3].

Note that the number of meaningful levels of abstraction for a requirement will depend upon its complexity. Simple requirements tend to have lesser number of abstraction levels (2 to 3), while complex requirements can have many more levels of abstraction. In such cases, it becomes easier to discuss the different levels when they are labeled. For example, levels of abstraction those are lower than the level of abstraction of the stated requirement can be labeled as L1, L2 and higher levels of abstraction as H1, H2 etc.

Step 3 – Choose the Right level of Abstraction for Each Phrase in the Requirement: After the abstraction matrix has been populated with the different levels of abstraction, we need to find out which level of abstraction is most faithful to the business need. For this, we need to verify the validity of each phrase in the created abstraction matrix with the user, starting from the lowest level of abstraction in each ladder or column of the abstraction matrix. If the user is not able to provide a reason for the inclusion of a phrase at a certain level of abstraction, then it must not be part of the business requirement. The lowest level of abstraction at which the user is able to justify the requirement phrase will form a part of the final requirement. If a level of abstraction higher than this is chosen, it will result in the

requirement, while the higher level of abstraction missed out on the important requirement of sending the mail to the lead service provider of the ticketed group.

In Ladder 3, the lower level of abstraction that had a quantifiable term was chosen over the more subjective abstraction as the quantification was based on Service Level Agreements with the service providers.

In Ladder 4, the users realized that the requirement phrase at the higher level of abstraction was more suitable as there were automated systems that had the capability to create service tickets and messages needed to be sent out even for those tickets.

The phrase in Ladder 5 did not need any new level of abstraction and so it was included

*As the level of abstraction reaches the topmost rung,
the analyst can be sure to have gathered the final set of
business requirements*

loss of important details and moving to a lower level will impose unnecessary restrictions on the system. In our helpdesk application example, the levels at which the users were able to justify each of the phrases are highlighted in Table 3.

In Ladder 1 of the abstraction matrix, though the situation required an email to be sent to the users, there was no business case for it and all they really wanted from the service provider was to be informed of the ticket. So the higher level of abstraction i.e., *A message must be sent* was chosen.

In Ladder 2, the lower level of abstraction was chosen as it gave the exact business

in the requirement as stated.

By combining the phrases that the users are able to justify in each of the abstraction ladders, we can arrive at the exact business requirement. Thus, in our example of the help desk application, the final business requirement became – *A message must be sent to the assigned service provider and the lead service provider of the ticketed group within 5 minutes of a service ticket being raised for a non-Enterprise Event issue.*


As can be seen, the final requirement arrived at, is quite different from the initial requirement stated by the users. Arriving at the correct business requirement through the help of

the abstraction matrix can help provide a much more flexible solution. In the above example, since the requirement was restated to use 'Message' instead of 'e-mail' different channels of messaging such as paging, e-mail, desktop alerts, etc. could be considered to make the process more efficient. Also by getting to the right level of detail in the initial phase itself, scope creep in the latter stages can be significantly contained.

CONCLUSION

The practice of General Semantics through the use of the abstraction ladder and other techniques, during the requirements elicitation process can, to a large extent, improve the accuracy of requirements by helping the requirement engineer understand the real business needs of the user and arrive at the right level of submission. Improving the accuracy of requirements can help create more effective solutions to satisfy the requirements and at the same time reduce the scope creep in the later stages of the software lifecycle, leading to a reduction in both time and effort and increase in software quality.

REFERENCES

1. The Standish 2004 CHAOS Report. Available on http://www.standishgroup.com/quarterly_reports/index.php
2. Alfred Korzybski, *Science and Sanity: An Introduction to Non-Aristotelian Systems and General Semantics*, 1933
3. Alfred Korzybski, *The Role of Language in the Perceptual Processes*, The Ronald Press Company, New York 1951. Available at <http://learn-gs.org/library/ak-role.htm>
4. European Society for General Semantics. Available on <http://esgs.free.fr/uk>
5. Steve Stockdale, *Korzybski's Structural Differential and Hayakawa's Abstraction Ladder*, April 2000. Available at <http://learn-gs.org/learningctr/abs-mod.htm>
6. S I Hayakawa and Alan Hayakawa, *Language in Thought and Action*, Wadsworth Publishing, 1989
7. The Institute of General Semantics. Available at <http://www.time-binding.org/>. 

Business Analysis Using S2S Framework

By Anshuman Pramanick

S2S Framework enables IT requirements to be modeled for validating against business strategy

The concept of strategic business analysis has matured over the past few years and is now emerging as a special stream within business analysis. It is always critical for a successful business that organizational IT is aligned to it and provides support to the organization's business strategy. As an application is a part of the strategic and operational ecosystem, a business analyst plays a vital role in this alignment of strategic view with the applications and support processes. The role of the business analyst though remains the same and so do the critical success factors. However, the newly developed strategic aspect broadens the role of a business analyst. As more and more business analysts get aligned to the strategic aspect of business, it is being considered to be the most value adding task than value creation and value capture.

As the purpose of business is to earn profit, a business analyst should focus on increasing the top-line and decreasing the bottom-line so that the profit margin increases. Eventually all IT projects are sponsored by business, keeping some strategic goals in mind. So the purpose of business analysis

is to identify critical drivers of success and discern viability of business ideas.

Though analysis is an integral part of a business transformation or a project, traditionally in most of the IT projects, business analysis is considered more to be a part of requirements management. Thus the role of a business analyst demands more than mere requirements management. The importance of analysis is evident from the fact that CIOs and IT executives consistently rank alignment of IT with business strategy as a top priority. Effective strategic alignment positively influences IT effectiveness and leads to superior business performance. The objective of this paper is to discuss ways to align strategy with system so that business process management, requirement engineering and management become more and more effective.

ALIGNING BUSINESS ANALYSIS: S2S FRAMEWORK

The Strategy, Context and Process (SCP) is a requirement engineering model based on the three elements – strategy, context and process – whose

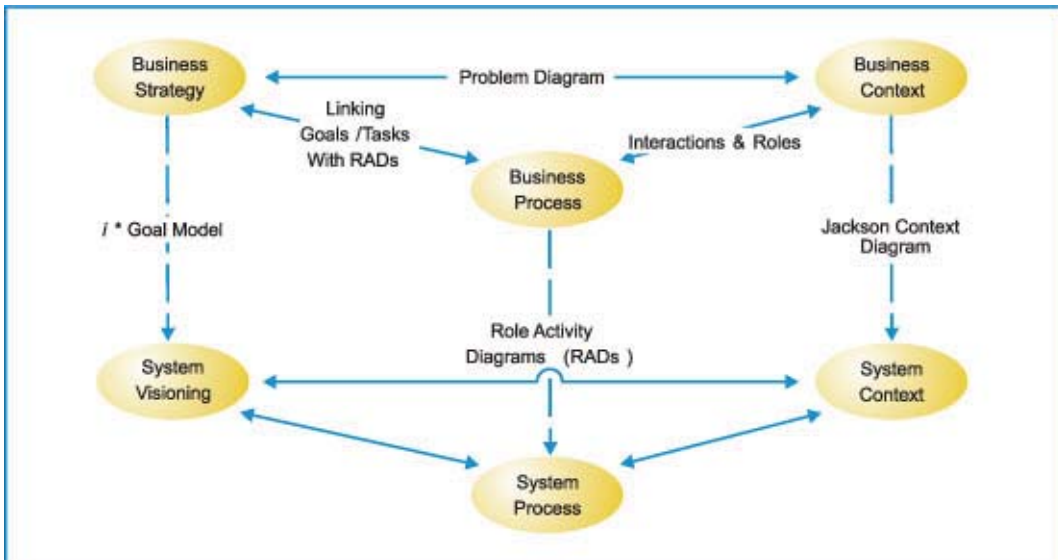


Figure 1: The S2S Framework

Source: www.dec.bournemouth.ac.uk

purpose is to enable verification and validation of requirements with underlying business strategy and supporting business processes [1]. *Strategy* refers to how an organization intends to use IT to compete in its market or industry. *Context* refers to the business and organizational environment in which an organization operates. It includes both internal and external organizational structures. *Process* refers to business operations, their support systems and other organizational resources, roles, entities and the interactions among these.

For each of the elements a requirements analysis technique is used – *i** goal modeling for strategy, Jackson context diagrams (part of Jackson problem diagrams) for context, and Role-Activity Diagrams (RADs) for process.

A simplified view of the model provides a framework that integrates and connects Strategy to System (S2S) [Fig. 1]. An *i** goal model is integrated with Jackson context diagrams using a problem diagram framework,

by treating goals/tasks as the requirements part of a problem diagram. Jackson context diagrams are integrated with RADs by maintaining equivalence between the roles in RADs and the domains of interest in context diagrams that imply equivalence between shared phenomena and interactions. Activities and state descriptions in RADs are cross-referenced with task, soft goal and hard goal entities in *i** [2]. Also, as the output of a process is to be the achievement of a goal, processes are linked to goals in the goal model. In this way, RAD is connected to a goal model at multiple points of reference. Thus, the three nodes shown in Figure 1 are inter-linked between each other and form an alignment in S2S framework.

Also, in order to validate lower-level requirements against higher-level strategic objectives, a means to top-down refinement and bottom-up traceability is critical. This is accomplished using the Goal model (Fig. 2), as goals can be refined from high-level strategic concerns to low-level technical ones. These

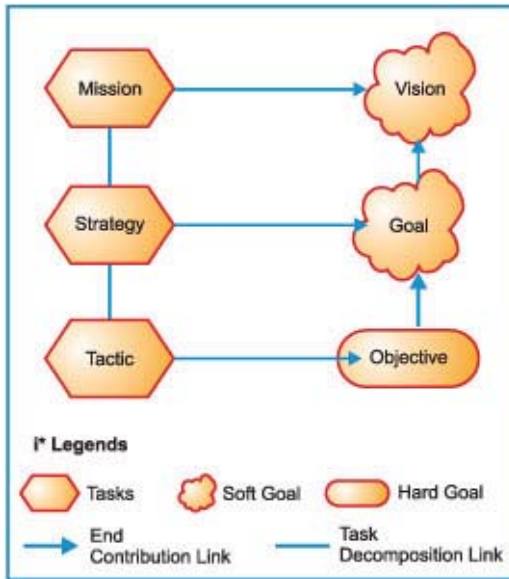


Figure 2: i* Goal Model

Source: www.e-sjis.org

requirements can be classified as goals and sub-goals respectively using reasoning approaches to refine goals into sub goals. As goals get refined, they refer to context at lower levels of abstraction. Jackson context diagrams accommodate context decomposition using projection. Similarly, RADs are capable of representing processes at multiple levels of abstraction.

Therefore, this framework mitigates three fundamentals of a business – *why* to be done (Goal model), *what* to be done (Context diagram) and *how* to be done (Process model). Context diagram can also be replaced with business models.

REQUIREMENT ENGINEERING MODEL USING THE S2S FRAMEWORK

S2S framework, a simplified SCP is presented in the previous section. Here we further discuss on how this S2S framework can be used for requirement analysis.

Strategy – The Goal Model: Discovering what an organization’s goals are and where they fit into the overall structure of a goal model is a well-known problem in requirements engineering. Goal here represents the objectives derived from business mission and strategy and what is to be achieved through co-operation of system/application and actors within the same business environment. Attempting to model an organization’s business strategy where soft goals are the norm and goal refinements are layer deep (sub-goals), poses additional challenge for a requirements engineering goal modeler.

There are several requirements engineering goal modeling notations. For convenience, i* model is used, as it is a widely recognized notation in the requirements research community [3]. The SCP strategy theme deals only with goals and tasks, whereas, modeling entities of soft goals, hard goals, tasks and their relationships are derived from the i* notation [Fig. 2]. Typically i* model consists of two components – Strategic *dependency* model and strategic *rationale* model. Strategic *dependency* model describes *dependency* on various actors in organizational context to achieve goals/tasks. Strategic *rationale* model attributes reasons for each *dependency* and explains how an actor reaches a goal. The same can be a useful input for use-case development.

Context – Jackson Context Diagrams: Jackson Context Diagrams [4] are means to scope the context of a problem, i.e., locating the problem and the reason why it concerns the real world, and answering the context that is relevant, useful and necessary to describe the problem.

Problem context may also be decomposed into smaller, sub-problem contexts at lower levels of abstraction and Jackson proposes heuristics for this. However, these heuristics are useful only once the problem scope and context have been

determined and the problem is recognized; a typical ground level problem involving a machine and an operator can be considered to be an example of such a situation. Problems discussed here are related to business strategies and are not typically ground level problems. Thus, they imply a need for an alternative approach to identify the problem context from where one can begin the breakdown into sub-problems. Weill and Vitale's business modeling framework might be applied to serve the purpose.

Weill and Vitale define a business model as "a description of the roles and relationships among an organization's consumers, customers, allies and suppliers that identifies the major flows of product, information and money, and the major benefits to the participants" [5]. Based on their definition, Weill and Vitale developed a graphical modeling notation whose entities include the organization of interest, suppliers, allies, customers, consumers and the relationship among these entities. The connects in relationships describe flow of money, product and/or information. A number of key business requirements are associated with each graphical model of business context so that the model might be successful.

The Weill and Vitale business model framework fits well into a problem diagram, as it provides the basis for separation of concerns between business model context and requirements of business strategy. The contextual part of the business model, that Weill and Vitale model represents graphically, can be modeled as a Jackson context diagram. Physical domains of interest represent business model participants, i.e., organization of interest, suppliers, allies, customers and consumers. The relationships among the participants are indicated as interfaces where flow of money, products and information are described as shared phenomena. While Weill

and Vitale simply list the strategic requirements associated with each business model, K. Cox et al., provide structure to these by representing them in a goal model in the requirement part of the problem diagram.

The context model describes a simple model of a wholesale banking setup [Fig. 3]. In wholesale banking, there are arrangements between the bank and the importer for providing line of credit, financing the production cost, protecting against non-payments/damages, etc. The wholesale bank, importer, exporter and logistics are described in the context diagram. The shared phenomena are indicated and labeled with single letters or a letter and number in the case of an interface. The interfaces A1, A2, B, B1, B2, and C in the context diagram are explained in the box below the problem diagram. The requirement part of the problem diagram describes the strategy of the wholesale business in the form of goals specified at each participant's level.

In Figure 3 the contexts and the goals are linked with dotted lines. Thus, the participants of the context diagram in turn become actors of the goal model and in this way, a problem diagram framework is used to integrate *strategy* represented as a goal model and *context* represented as a context diagram, according to the business modeling framework suggested by Weill and Vitale [5].

There are three steps to develop Weill and Vitale's business model problem diagram:

- Identify the business model participants (actors) – the organizations of interest, importer, exporter and bank in this model.
- Identify the relationships among the participants (actors) in this flow. The relationships between participants

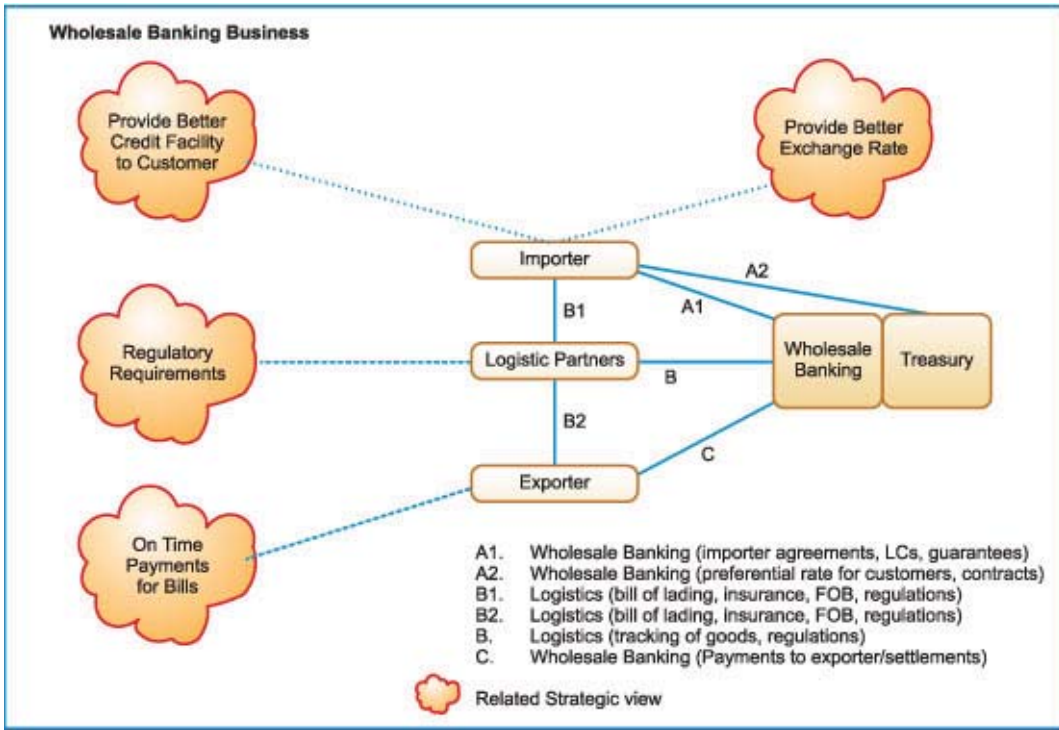


Figure 3: Business Strategy and Context as a Problem Diagram *Source: www.bnel.com*

represent interfaces between organizations of interest.

- Identify the strategic requirements of the business model and represent these as a goal model [1].

Identifying the Weill and Vitale business model and representing it as a problem diagram with an integrated goal model is a critical step in requirements analysis using S2S. The reason for it being that the top-level problem diagram defines both the scope of the business problem to be solved and the critical strategic, business objectives that are to be met. Without this knowledge the lower-level requirements of the system is unclear, and so eventually the strategic alignment will not be effective.

Refining Strategy and Context

A problem diagram at the Weill and Vitale business model level, however, is very distant from system-level requirements and is likely to be too abstract to begin designing and implementing a solution consisting only of hardware, software, data, network resources and human resource. The concept of a progression of problems is particularly useful when it comes to refining requirements from a high-level problem diagram till the machine level [6]. Bottom-up traceability of requirements in the goal model, in conjunction with verifying requirements within the context at appropriate levels of refinement, is essential to validate overall strategic alignment of requirements. Figure 4 overleaf illustrates a progression of problem diagram.

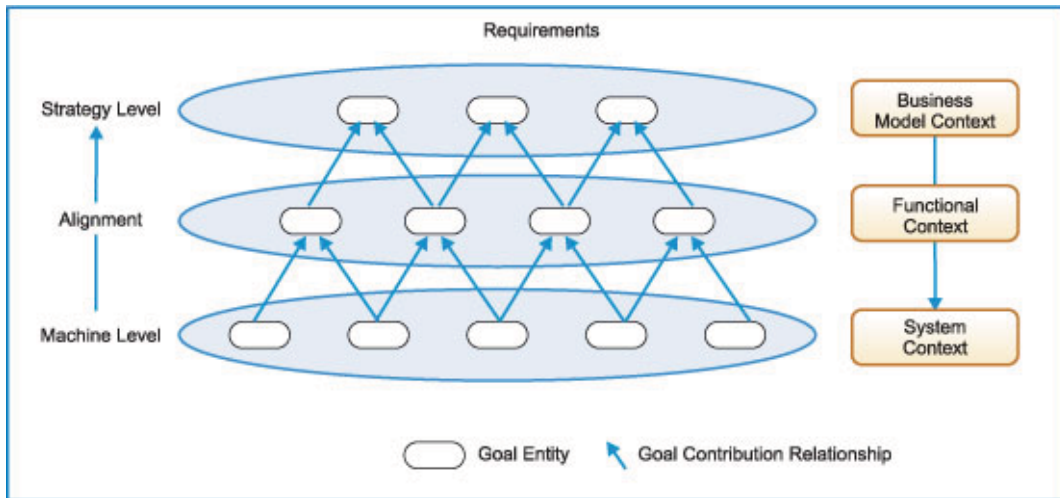


Figure 4: Progression of Problems

Source: www.jacksonworkbench.co.uk

In Figure 4 the uppermost layer represents the requirements of business strategy, in the form of a goal model, associated with the Weill and Vitale business model. Through an analysis of business model context, it is possible to decompose the domain context into a more refined functional context. Here goals are identified as requirements in problem diagrams. In Figure 4 a single, large goal model, used to model the strategy theme of the S2S framework is partitioned at multiple levels of refinement by the requirement elements in the upper layer of the figure. A goal model representation is particularly useful when performing a progression of problem diagram analysis, as the goal model furnishes a means to link one level of refinement of strategy with adjacent levels.

Linking Process Models to Goal Models and Context Diagrams

A process consists of tasks, activities and roles in order to achieve a desired output or goal [7]. Many companies use process models to describe their activities and systems. The three commonplace

tools are Class Diagram, Data Flow Diagram (DFD) and Role Activity Diagram (RAD). There is a close relationship among process models, context diagrams and goals, though moving from process models to context diagrams is a recommended approach [8]. In addition, Ould recommends performing business process analysis according to the organizational goals, providing a link between business processes described by RADs and the goals appearing in an organizational goal model [9].

This adopted framework, integrates the strategic business level problem to the system level problem in a progression.

IMPLEMENTING S2S FRAMEWORK IN SDLC

S2S framework is useful for aligning Strategy, Context and Processes from business level to system level. Software development lifecycle analysis identifies what a business needs and designs accordingly to realize the needs. But on the other hand, technology changes happen more quickly than business needs. Companies restructure accordingly, but their business

	Process	Data	Deliverable
Initiate	Business Requirement	Business Requirement	Terms of Reference
Analyze	Business Functions	Entities	Business Specifications
Design	Program Specification	Database Design	Technical Specifications
Build	Programs	Database	Tested system
Implement	Application		Stabilize and Review

Table 1: SDLC Classic Waterfall Model

Source: Infosys Research

purpose rarely changes. An effective business analysis bridges the gap here and gives a logical view of process and data needs but is not bound by physical implementation [10].

As shown in Table 1, typically in a software development and implementation setup, alignment is very important in the initial stages. Business requirements formed into use-cases and system requirements using S2S framework can always be traced back to the origin of the requirement. IT specialists and business users often complain about mismatch in their level of understanding. There is enough reason for the mismatch as technical people are concerned with making best use of technology and business users are concerned with achieving their business goals. Business analysis should neatly bridge the gap. The terms of reference produced during the initial phase should unambiguously include, the problem to be solved and the objective to be achieved. This SCP alignment through S2S framework is a complete top-down approach where ad-hoc requirements can be avoided as well as clear objectives of each requirements can be maintained. Consequently, the design derived out of the business analysis through S2S framework will be more effective.


CONCLUSION

S2S, a requirements engineering framework, is based on connecting and integrating the themes of strategy, context and process. The objective of this framework is to enable modeling organizational IT requirements for validating requirements against business strategy. S2S framework is aimed at proposing an approach to integrate business strategy model with context of business processes. It also aims at aligning scope of strategic-level context and IT requirements and demonstrating means of validating system requirements against business strategy and supporting processes via traceable links.

A practical implementation approach mitigating the SDLC requirements incorporating S2S framework is also proposed here. One key business imperative for implementing S2S framework will be to improve return on investment in strategic IT initiatives. Thus, the next logical step is to apply the framework in strategic IT initiatives and produce a business case specifying the benefits.

REFERENCES

1. Steven J Bleistein, Karl Cox, June M Verner, and Keith T Phalp, B-SCP: A Requirements Analysis Framework for Validating Strategic Alignment of Organizational IT based on Strategy, Context and Process, Information and Software Technology, Vol 48, No 9, 2006. Also available at http://nicta.com.au/__data/assets/pdf_file/0020/9047/Bleistein-et-al-B-SCPa.pdf
2. Karl Cox, Keith T Phalp, Steven J Bleistein, and June M Verner, Deriving Requirements from Process Models via the Problem Frames Approach, Information and Software Technology, Vol 47, No 5, 2005. Also available at <http://dec.bournemouth>.

- ac.uk/ESERG/kphalp/ist05.pdf
3. Claudio U Ciborra, De profundis? Deconstructing the Concept of Strategic Alignment, *Scandinavian Journal of Information Systems*, Vol 9, No 1, 1997. Also available at http://www.e-sjis.org/journal/volumes/volume09/articles/no1/99_Ciborra_p67-82.pdf
 4. Michael Jackson, *Software Requirements and Specifications: A Lexicon of Practice, Principles, and Prejudices*. Addison-Wesley Professional, New York, 1995
 5. P Weill and M Vitale, *Place to Space: Moving to eBusiness Models*. Harvard Business School Publishing Corporation, Boston, 2001
 6. Michael Jackson, *Problem Frames: Analyzing and Structuring Software Development Problem*, Addison-Wesley Professional, New York, 2001
 7. Nuno Melão and Michael Pidd, A Conceptual Framework for Understanding Business Processes and Business Process Modelling, *Information Systems Journal*, Vol 10, No 2, 2000
 8. Carol Britton and Jill Doake, *Software System Development: A Gentle Introduction*, McGraw-Hill, Berkshire, 2006
 9. Martyn A Ould, *Business Processes: Modelling and Analysis for Re-engineering and Improvement*. Wiley, New York, 1995
 10. Brian Cooney, *Separating Analysis from Design*, IRM Training White Paper, 2006. Available at http://www.irm.com.au/Separating_Analysis_from_Design.PDF. 
-

Integrated Approach to Requirements Engineering

By Ashish Chandra, Ravishankar N and Tushar Sharma

Structured approach to requirements engineering can overcome key organizational challenges and enhance efficiency and effectiveness of business analysis

“The hardest single part of building a Software System is deciding precisely what to build [1].” Known by many names such as ‘Product Definition’, ‘Scope Elaboration’ or ‘Conceptualization’, the activity of understanding what the customer wants and clearly documenting requirements sits right at the top of a product or application development lifecycle priority list. With such a positioning, if this important activity is not carried out correctly and in a timely manner, the resulting solution is bound to be plagued with defects, running into cost overruns, customer dissatisfaction and loss of credibility of the IT team.

According to a Standish Group research report, out of every 100 projects started in 2006 only 35 projects succeeded. 46 projects were challenged and 19 of them failed [2]. Poor requirements elicitation is one of the major concerns for such failures and challenges. An integrated approach to requirements

engineering with a structured set of procedures and process components will help address these challenges.

UNDERSTANDING REQUIREMENTS ENGINEERING

Requirements engineering as a discipline involves an integration of technical skills, functional knowledge, processes and a significant amount of human involvement. It is a crucial step where all the objectives are gathered, prioritized and the team sets to achieve the objectives through the development lifecycle. However, as in most software projects, inadequate time is spent on developing and managing requirements effectively. Statistical evidences such as the ones listed in this paper show that inadequate requirements negatively impact a project.

According to a survey by IAG Consulting that studied more than 100 organizations,

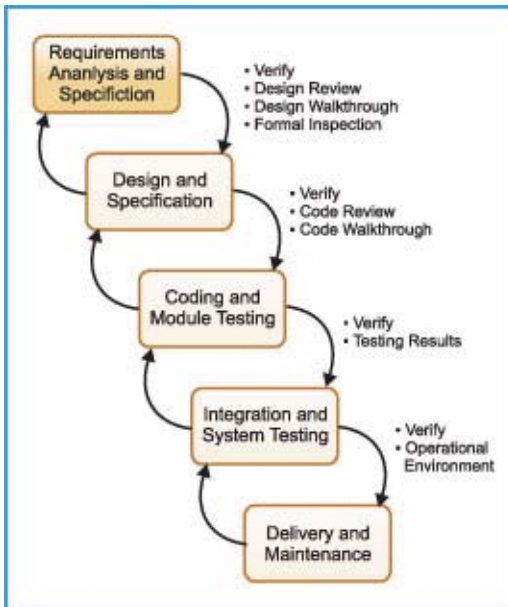


Figure 1: A Typical Software Development Lifecycle
Source: Infosys Research

companies with poor requirements spend on an average \$2.24 million more per project on strategic projects than those that employ requirements best practices [3].

Requirements analysis and specification is the first step in a typical Software Development Life Cycle (SDLC). As illustrated in Figure 1, this step sets the foundation for all the activities in the subsequent stages of SDLC.

Requirements engineering as a part of the software development life cycle is responsible for:

- Finding out what the requirements are
- Understanding the dependencies and relationships of the various requirements
- Prioritizing the requirements
- Managing the changes to the requirements
- Making sure that the requirements are met.

Often, the ownerships of such critical steps are not clearly established. This leads to lack of communication among the key stakeholders which further leads to misunderstood or unclear requirements [3]. Captured requirements are often vague and ambiguous, making them difficult to be tested. Also, too many requirements from too few individuals, lead to loss of an organization-wide perspective and a lack of business focus. On the other hand, overstating requirements may lead to unwanted features being delivered.

CHALLENGES IN BUSINESS ANALYSIS AND THEIR IMPACT

Challenges faced by an organization during developing and managing requirements are multi-dimensional. An absence of communication between the key stakeholders of the project, a lack of an organization-wide vocabulary to conduct effective interviews or workshops lead to inconsistencies in the capture and analysis of requirements. This is also compounded by a lack of knowledge of probing techniques and a depth of understanding required to separate opinions from hard facts.

Completeness and correctness of requirements are often not verified and the requirements are not reviewed by the right set of roles even if they are reviewed. Figure 2 depicts the categories of challenges in requirements engineering.

Such challenges have a snowballing effect on the project. There can be cost overruns or in other words, the cost of fixing a requirements related defect in production can be 110 times more than the cost of correcting it during requirements definition [4]. Requirements errors can contribute up to 85% of the total project



Figure 2: Requirements Engineering Challenges

Source: Infosys Research

Organizational Impact	Project Impact	Human Impact
<ul style="list-style-type: none"> Lack of predictability Cautious spending for future projects Lack of credibility for the IT organization Impact on Business Strategy due to delayed project 	<ul style="list-style-type: none"> Schedule Slippages Costly rework and ineffective utilization of budget Inability to predict project performance Poor quality of end deliverables 	<ul style="list-style-type: none"> Low Customer Satisfaction Low Team Morale Loss of credibility of IT group Higher attrition rates

Table 1: Impact of Ineffective Business Analysis

Source: Infosys Research

rework cost [5]. Schedule slippages can occur leading to a delayed project and an unsatisfied customer. IT as well as business team morale can go down because of unstable requirements. The impact has been summarized in Table 1.

STAGES OF EXCELLENCE

A study of the steps used by an IT organization to gather and analyze requirements will provide insights into the maturity of the methods and procedures related to requirements engineering. Typically an IT organization will pass through multiple stages before achieving a proven, practical and robust set of processes and practices to achieve excellence in gathering, analyzing and managing requirements. These stages of excellence proposed in Table 2 overleaf, can be applied to leverage process models and industry best practices.

In Table 2, it can be seen that a governance structure and organization-wide processes and guidelines are fundamental to the success of Requirements Engineering. However mere existence of processes or guidelines will

Excellence Parameter	Stage 1	Stage 2	Stage 3	Stage 4
Governance and Stakeholder Involvement	A structure for Requirements elicitation and management does not exist. The relevant stakeholders are not involved at the appropriate stages of the life cycle.	IT-Business interface is ad hoc. Some business users are involved in requirements gathering stage.	All business stakeholders are involved during requirements gathering and change management. The necessary project/program level guidelines and Governance exist.	IT Organization level Governance structure exists for requirements engineering. All business and IT stakeholders are involved upfront in the requirements stage and continuously in the subsequent stages of the SDLC.
Requirements Elicitation and Review	Business Requirements are gathered and documented in an ad-hoc manner. Time availability is the primary parameter to select people for gathering requirements. Elicitation mechanism is based on individual preferences.	Functional and non-functional requirements are gathered using multiple elicitation techniques and documented in multiple places. Individuals with some business domain exposure are involved in requirements elicitation. The requirements are not formally reviewed by stakeholders.	Requirements are formally documented and reviewed by some stakeholders. Completeness of requirements is not confirmed. Individuals trained in Business Analysis gather and analyze requirements.	Requirements are gathered using standard techniques, documented using standard templates. Final requirements are reviewed and signed off by all stakeholders.
Analysis and Modeling	Captured requirements are not analyzed to determine design/ architecture elements and development sequence.	Requirements are analyzed. Models are not developed.	Proof of Concept is developed in an unplanned manner whenever found necessary.	Requirements are analyzed and need for prototypes and models is identified upfront in consultation with the stakeholders. Models, prototypes and proofs of concepts are developed as per an agreed plan.
Traceability and Change Management	Requirements are maintained in isolation and not linked to the subsequent stages of the development lifecycle	Initial requirements are uniquely identified and mapped to test cases. Changes critical requirements are analyzed.	Initial requirements and changes are mapped to test cases. All changes are analyzed and the impact is communicated to stakeholders. Changes to design components, program elements or test cases are not traced back to requirements.	Initial requirements and changes are traced on a regular basis to all the components of the complete development life cycle. Based on the changes, effort, schedule and other estimates are revised and agreed upon by stakeholders.
Measurement	No measures related requirements are defined. Some measures are gathered in an adhoc manner.	Individual programs / projects define specific measures related to requirements. These are not continuously monitored. No measures related requirements are defined. Some measures are gathered in an ad hoc manner.	Measures related to scope stability; impact analysis time, change turn around time and other such aspects are defined at the IT organization level and are tracked at program / project level. No measures related requirements are defined. Some measures are gathered in an ad hoc manner.	Metrics are analyzed at the IT organization level and actions are initiated as necessary to improve the requirements specification and change management methods. The metrics themselves are revisited / refined periodically to align with business needs. Measures related to scope stability; impact analysis time, change turn around time and other such aspects are defined at the IT organization level and are tracked at program / project level. No measures related requirements are defined. Some measures are gathered in an ad hoc manner.

Table 2: Proposed Stages of Excellence

Source: Infosys Research

not result in effectiveness of Requirements Engineering. The success of the processes and guidelines depend on how well they are practiced ensuring stakeholder involvement,

thorough analysis and clear documentation of requirements at every project. In Table 2 the parameters related to project level practice of Requirements Engineering activities are

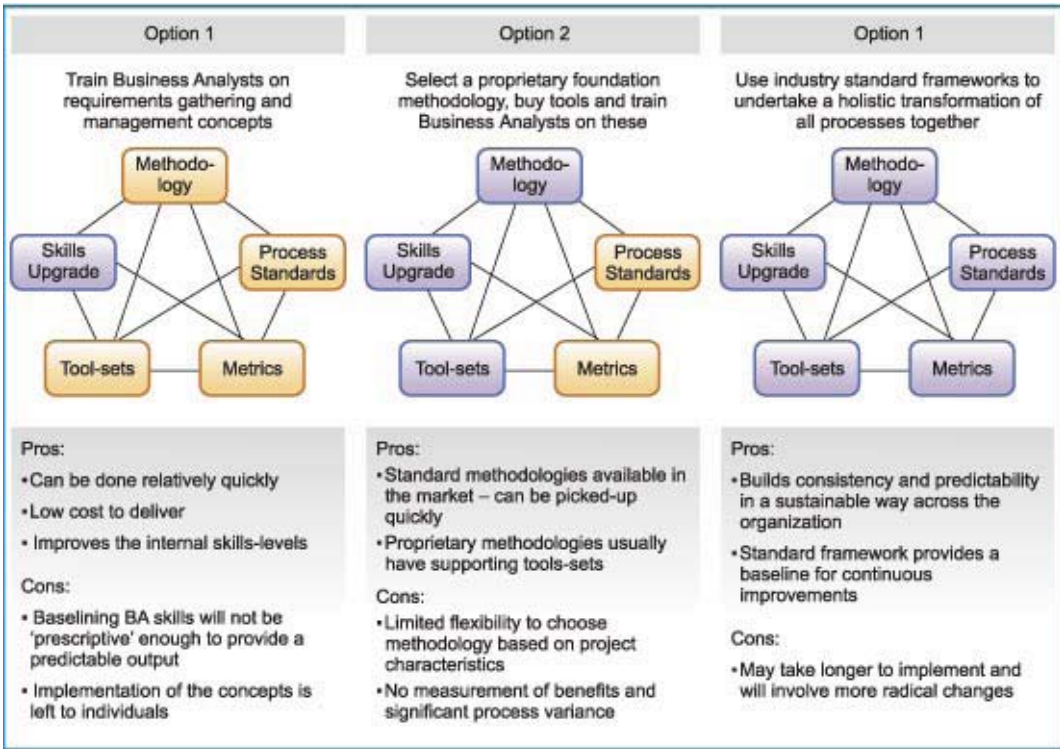


Figure 3: Options to Progress through the Stages of Excellence Source: Infosys Research

also described through the proposed stages of excellence. Finally, to track the maturity of the Requirements Engineering practices, the table suggests taking the parameter of measurement through the stages of excellence. In other words, when the Requirements Engineering practices mature, the measurement and analysis activities related to them also mature and the project teams get clear quantitative indicators on how well requirements are gathered, documented and managed. Eventually the IT organization develops the capability to predict the performance of the projects with respect to requirements.

IT or the business leadership can consider many options to progress through

the stages of excellence. Depending on the business priorities, an organization may adopt a combination of the options as shown in Figure 3. For example, if 'visibility to the current requirements practices and realizing quick benefits' are short term objectives, the organization can choose to address the metrics dimension under option 1 or 2 along with the basic parameters such as skills and tools.

INTEGRATED REQUIREMENTS ENGINEERING FRAMEWORK

To achieve efficient requirements elicitation, documentation and management, IT and business have to collaborate and adopt a structured approach involving stakeholder

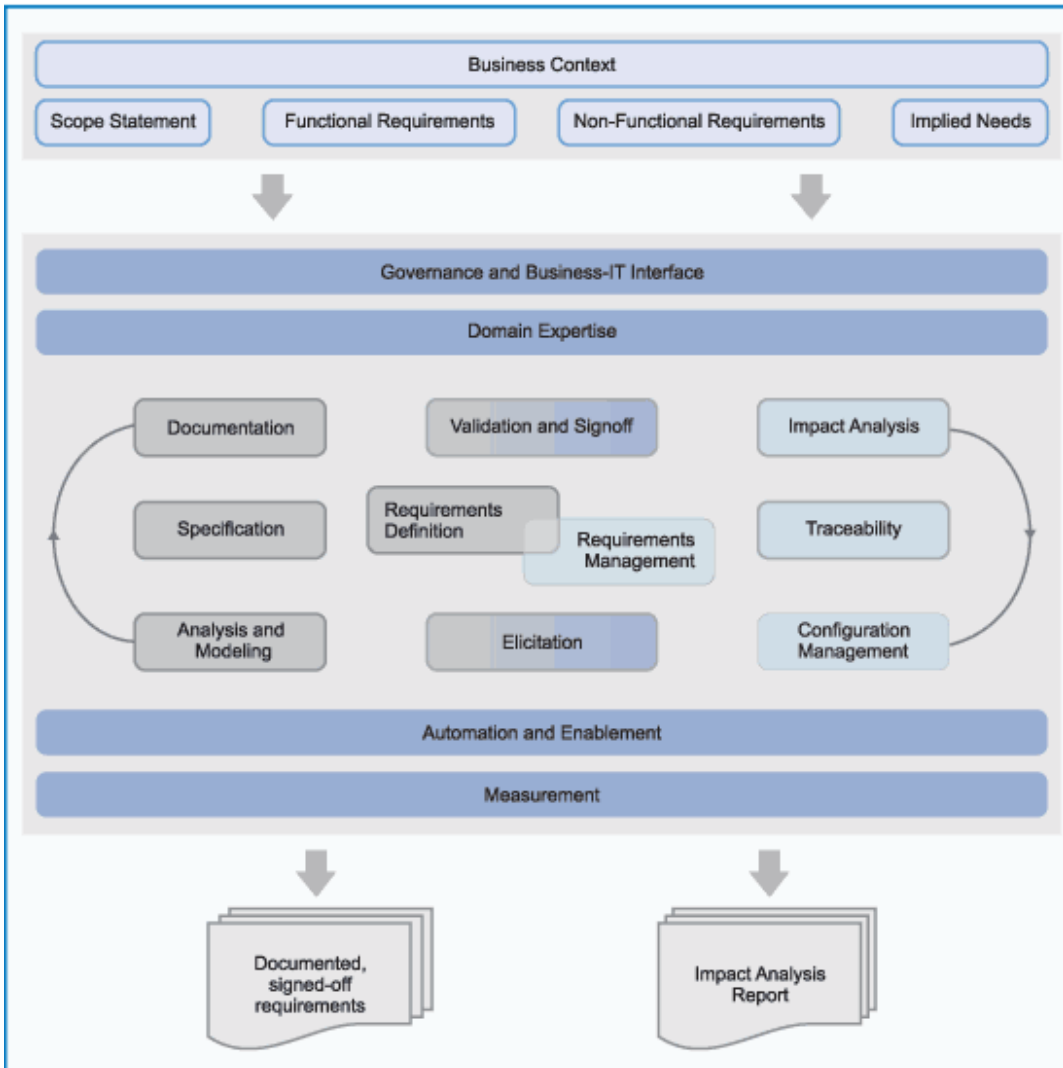


Figure 4: Integrated Requirements Engineering Framework *Source: Infosys Research*

education, business analyst skills enhancement, SME reviews, documentation and necessary automation. This structure can be provided by a framework that leverages industry best practices in requirements engineering.

The integrated framework described in Figure 4, synergizes the relevant aspects of requirements elicitation, documentation, change management and communication to

help IT organizations achieve excellence in Requirement Engineering.

The framework uses standard process improvement models such as CMMI® and industry best practices [6]. It places business context at the top in order to pass vital inputs on overall scope, and functional and non-functional requirements on the subsequent steps of Requirements Engineering. The framework

then takes the business needs as inputs and uses Requirements Definition and Requirement Management at the core, supported by several key components as described in Figure 4.

Requirements Definition: This is one of the initial steps in the SDLC and it sets the foundation for the subsequent stages of the SDLC. This step is characterized by the following:

- **Requirements Elicitation:** Here the IT team interacts with business stakeholders to gather their needs. Typically techniques such as brainstorming, discovery workshops and storyboarding can be used to elicit business stakeholder requirements.
- **Analysis and Modeling:** Here the gathered requirements are analyzed and prioritized. To demonstrate understanding of the requirements by IT, use case models, prototypes and proofs of concept can be used at this stage.
- **Specification and Documentation:** These steps help in documenting business user needs into more detailed specifications that can be linked forward to subsequent technical components of the proposed application. At this stage after the necessary reviews by the stakeholders concerned, the requirements are baselined and communicated to all.

Requirements Management: This step takes prominence after the requirements are baselined. This step involves the following:

- **Impact Analysis:** The proposed change to

one or more requirements is documented and the impact of the change is analyzed to make decisions on priority, re-estimation and other related preparations to carry out the change. After the impact analysis it may also be decided to defer or reject the proposed change.

- **Traceability:** This aspect is also closely related to Requirements Definition. This entails mapping the initial set of requirements and the subsequent changes to the components of all the subsequent stages of the SDLC. It also entails tracing the related requirements amongst themselves. For example, a change to a requirement may impact another requirement. Requirement Traceability can be implemented in many ways using a sophisticated database to uniquely identify the requirements and map them in a one-to-one, one-to-many, many-to-one or many-to-many relationships with technical components such as design elements, software code and test cases. Alternatively, a simple spreadsheet can be used to trace requirements to the lifecycle stages.
- **Configuration Management:** This is an important activity to make sure that the latest versions of multiple requirements documents, use case models, prototypes and other components of requirements engineering and their change history are accessible to the relevant stakeholders. This activity also enables setting up the necessary access permissions, back-up mechanisms and procedures to verify the integrity of the baselined requirements engineering components.

- **Validation and Signoff:** This activity ensures that the requirements are reviewed and confirmed by the stakeholders.

To ensure that these core activities are carried out effectively, the framework proposes the following support structures and enablers.

Governance and Business IT Interface:

This sets the tone and pace of Requirements Engineering. Governance typically defines the necessary organization structure, processes, guidelines, measurements and monitoring mechanism to ensure that the IT and business teams interface effectively to define the requirements accurately in a timely manner, eliminating delays and rework.

Domain Expertise: This is another key aspect specified in the framework to ensure that experience and knowledge specific to the business domain are available to articulate and elaborate the needs of the business. Domain expertise is also required to bring to the table facts on trends in a given domain, in order to specify provisions for potential enhancements in the proposed application.

Automation and Enablement: The framework recognizes that the requirements definition and management activities could be accelerated through necessary automation. Tools such as Caliber® from Borland or Influx from Infosys could be used to document the requirements and trace them through the SDLC. Along with the need for automation, the framework also suggests enabling and training the IT and business participants in requirements elicitation and requirements change management.

Measurements: “If you cannot measure it, you cannot improve it [7].” Taking cognizance of this fact the framework specifies *measurement* as one of the key elements to achieve excellence in Requirements Engineering. Examples of metrics that can be used to determine the Requirements Engineering capability of an IT organization are Requirements Stability Index, Change Acceptance Percentage and Impact Analysis Agility.

As is evident from the discussion so far, certain activities such as validation and signoff, elicitation and traceability overlap between the *definition* and *management* aspects.

BENEFITS OF THE FRAMEWORK

If implemented effectively, the above framework offers several benefits to the business and IT organization in their journey towards excellence in Requirements Engineering. The benefits are:

- Governance structure with the associated measurements helps in bringing the stakeholders together to increase their involvement to provide their inputs during requirements elicitation, analysis, change impact analysis and implementation.
- Continuous stakeholder involvement and validation of the captured requirements with the stakeholders improve understanding of stakeholder context and increase the probability of capturing all their stated and implied needs in the context of the proposed IT solution. Eventually the components of the framework help create a complete set of unambiguous prioritized requirements.

- It improves management of changes to requirements to minimize surprises during acceptance. Impact analysis, requirements traceability, configuration management practices described in the framework work in conjunction to effectively manage changes to the requirements throughout the lifecycle of the project.
- Eventually it creates a 'win - win' situation for customers, stakeholders and the project team.


CONCLUSION

Effective Business Requirements Analysis is key to laying the foundation for a quality software application. Lack of a structured approach towards gathering and analyzing requirements, and managing changes in the initial stages leads to confusion in the design and development stages and results in cost, quality and schedule issues. Addressing issues such as lack of domain expertise, business analysis capability, inadequate processes/standards or lack of tool in an isolated manner will not eliminate the root causes for ill defined requirements and inadequate analysis. An integrated framework such as the one suggested in this paper will help an IT organization determine their current strengths in requirements engineering and understand the areas for improvement in all the people, process and tools issues related to requirements engineering. A structured approach to requirements engineering using the above framework will also enable the IT organizations navigate through the Stages of Excellence in Requirements Engineering.

REFERENCES

1. Fred Brooks, *The Mythical Man-Month: Essays on Software Engineering*, 20th Anniversary Edition. Reading, Addison-Wesley, 1995
2. David Rubinstein, Standish Group Report: *There's Less Development Chaos Today*, 2007. Available at www.sdtimes.com/content/article.aspx?ArticleID=30247
3. Business Analysis Benchmark Report, IAG Consulting, 2008. Available at <http://www.iag.biz/about-iag/news--events/business-analysis-benchmark.html>
4. Robert B Grady, *An Economic Release Decision Model: Insights into Software Project Management*, Proceedings of the Applications of Software Measurement Conference, Software Quality Engineering, 1999
5. Dean Leffingwell, *Calculating the Return on Investment from More Effective Requirements Management*. American Programmer, Vol 10, No 4, 1997
6. Capability Maturity Model Integration, Software Engineering Institute, Carnegie Mellon University. Available at <http://www.sei.cmu.edu/cmm/index.html>
7. Lord Kelvin, *Electrical Units of Measurement in Popular Lectures*, Vol 1, 1883.
8. Bashar Nuseibeh and Steve Easterbrook, *Requirements Engineering: A Roadmap*, Proceedings of the First Westminster Conference on Professional Awareness in Software Engineering, Royal Society, London, 1996. Available at <http://www.doc.ic.ac.uk/~ban/pubs/sotar.re.pdf>
9. Amer Al-Rawas and Steve Easterbrook, *Communication Problems in Requirements Engineering: A Field*

Study, Proceedings of the First
Westminster Conference on Professional
Awareness in Software Engineering,

Royal Society, London, 1996. Available
at [http://www.cs.toronto.edu/~sme/
papers/1996/NASA-IVV-96-002.pdf](http://www.cs.toronto.edu/~sme/papers/1996/NASA-IVV-96-002.pdf). 

Are you Plagued with Unnecessary Costs in your IT Projects?

By Manu Goel

OLAR can be an effective approach to snip unplanned costs common to all IT projects

Do you know how unforeseen costs in IT projects pose concealed threats during implementation? Consider this - an IT project was budgeted (and executed) at \$ 1 Million but gave birth to another project worth \$ 200K due to new requirements being discovered during the execution of the project. Will this be termed as a successful project or a failure?

This new project or *projectling* (sic) i.e., a fledging project of 200K will require re-budgeting and re-prioritizing various IT projects at the organization level. In other words, besides a 20% increase in the obvious cost, the cost of re-planning and opportunity cost due to re-prioritizing of certain projects will also get added to the total cost-to-organization. Similarly, there will be an overall IT schedule slippage to the organization due to the new project (that is actually a part of original project) and the re-prioritization of some other projects. Also, the defects discovered in the new project will be added to the total defects found in the original project thereby making the 'actual' quality also suffer; and of course, quality too has a cost associated with it. Thus, though

the project cost was under control, the overall cost to the organization increased in the form of new project + opportunity cost + cost of quality. Such *projectlings* have become so common that a lot of organizations now treat them as an inherent output from an IT project. It is at this stage of discussion that an Objective Linked Approach for Requirements Development or OLAR is being proposed to avoid creation of projectlings.

WHAT IS OLAR?

OLAR is an objective linked approach that runs in parallel with the requirements development process and helps verify each requirement with its objective. It also helps discover missed requirements for objectives. OLAR helps in mapping objectives to customer requirements and identifies gaps to enhance comprehensibility of requirements.

OLAR aims to change the way we approach requirements phase or business analysis for IT projects and development projects, by focusing on the objectives of business functions and sub-functions. This approach aims at doing it right

the first time and cutting down the unnecessary IT costs that an organization incurs due to incorrect and incomplete requirements gathering or improper understanding of requirements. It provides greater clarity to all the involved parties i.e., sponsors, business users, analysts, designers and developers and reduces the defect injection rate considerably.

OLAR is not about gathering requirements from the users or business stakeholders; rather it is all about the users and other stakeholders becoming a part of the business analysis process.

While discussing the ten step approach of OLAR, the paper will touch upon a conventional sample project i.e., ESU course registration system [1].

OLAR: A TEN STEP APPROACH

OLAR promotes introducing objectivity from the day the IT project is conceptualized. To start with, the OLAR team will look into the objectives of business functions. They will look at the overall mission and vision of the business at large, move to the business function in particular and try to find out sub-objectives and sub-sub objectives and see how they lead to the final overall business objective. To avoid any ambiguity in discussion within the paper, from here on sub-objectives or any further breakdown of an objective will be referred to as 'Objective'.

- **What and Why:** This step identifies the smallest of objectives. The focus is on identifying what the project wants to achieve and not on how it is to be achieved. Or in other words the focus should be on, what affects the business function rather than who is affected by the business function. Note that we are looking at preparing objective linking diagrams as an end result of the OLAR process, so one can

use either activity diagrams format or use-case diagram format to link the objectives. In Figure 1 one branch of objective linked diagram is shown. The prime objective (labeled S1) has been transformed into a slightly solution-linked objective (create an automated system) and then it is converted into several other objectives, each of which leads to a tree. Figure 1 shows the tree for only one of the second-level objectives. The rest of the second level objectives and their further synthesis is shown in Figure 2. However, Figure 2 is shown in a state of partial completion as only those objectives are shown that are needed to explain the OLAR approach. One can keep referring to Figure 1 and Figure 2 throughout in order to understand OLAR in a better way. In Figure 1, the way to link objectives is also shown. However, this linking is generally complete in step 3.

- **Objective Supporters:** This step identifies the data that is required to support an objective. This data, combined with required validations to be performed on the data, is noted against the corresponding objectives. However, the need for these data points should arise from the objectives themselves in order to make the data points eligible to be part of the system. For example first name, last name, address of customer, etc., are all data points but they arise from multiple objectives to be able to effectively communicate with customer, provide security, etc. If one knows such data points, the question to pose is *what objectives does this data point fulfill?* An example is shown in Figure 2. This query will help discovering several relationships

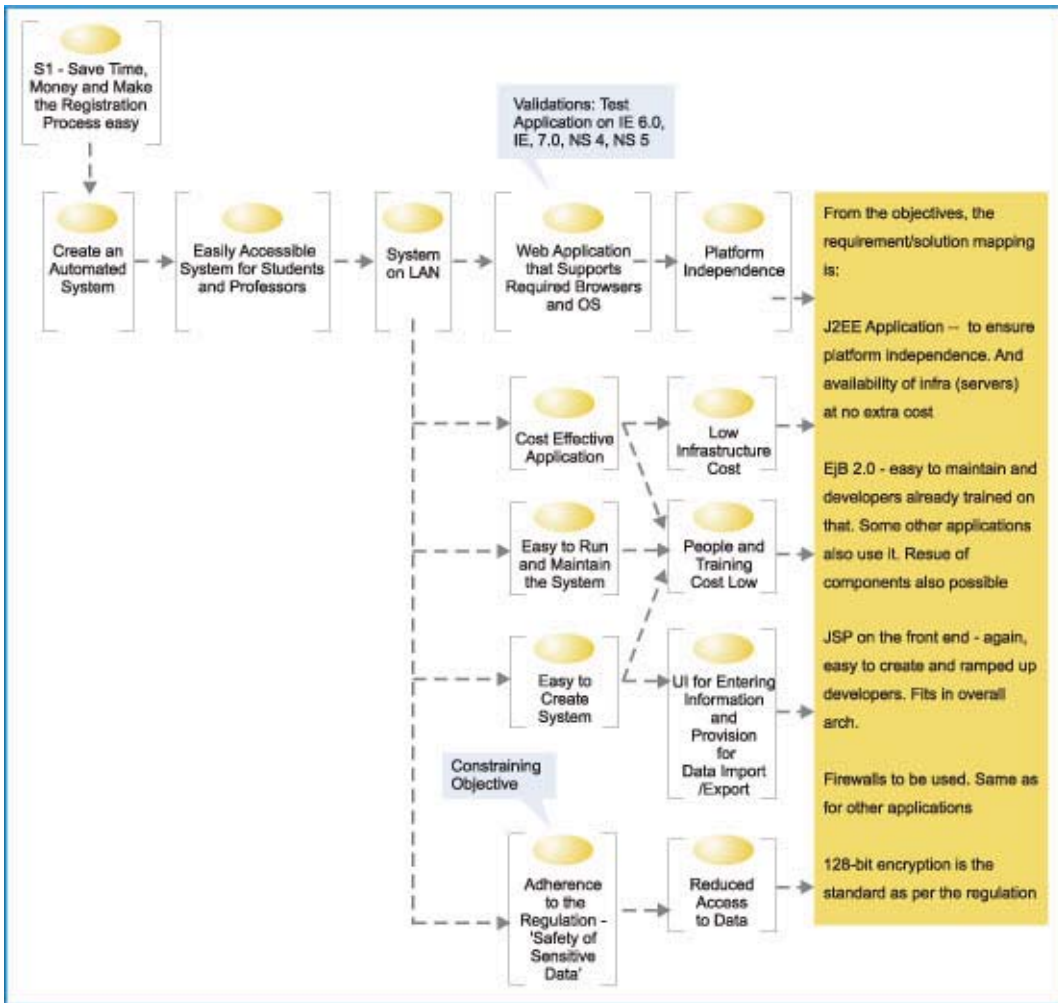


Figure 1: One Branch of an Objective-Linked Diagram

Source: Infosys Research

(inter-function and intra-function) that would otherwise go unnoticed. The said process will lead us directly to the next step.

- Comrades:** This step identifies the relationship between these smallest of objectives. The relationships should be such so as to lead to achievement of the overall business objective of the function. Note that we will not ask the users how

they want the things to work or how they carry out their work. Instead we will talk about what is expected from a particular business function. This step can be initiated and taken a step further in the first and second step too. From this particular step, our objective linking diagram will start taking some meaningful shape. Figure 1 shows complete Objective-linked diagram for the objective 'System on Lan.' Figure 2 illustrates some other OLAR concepts and

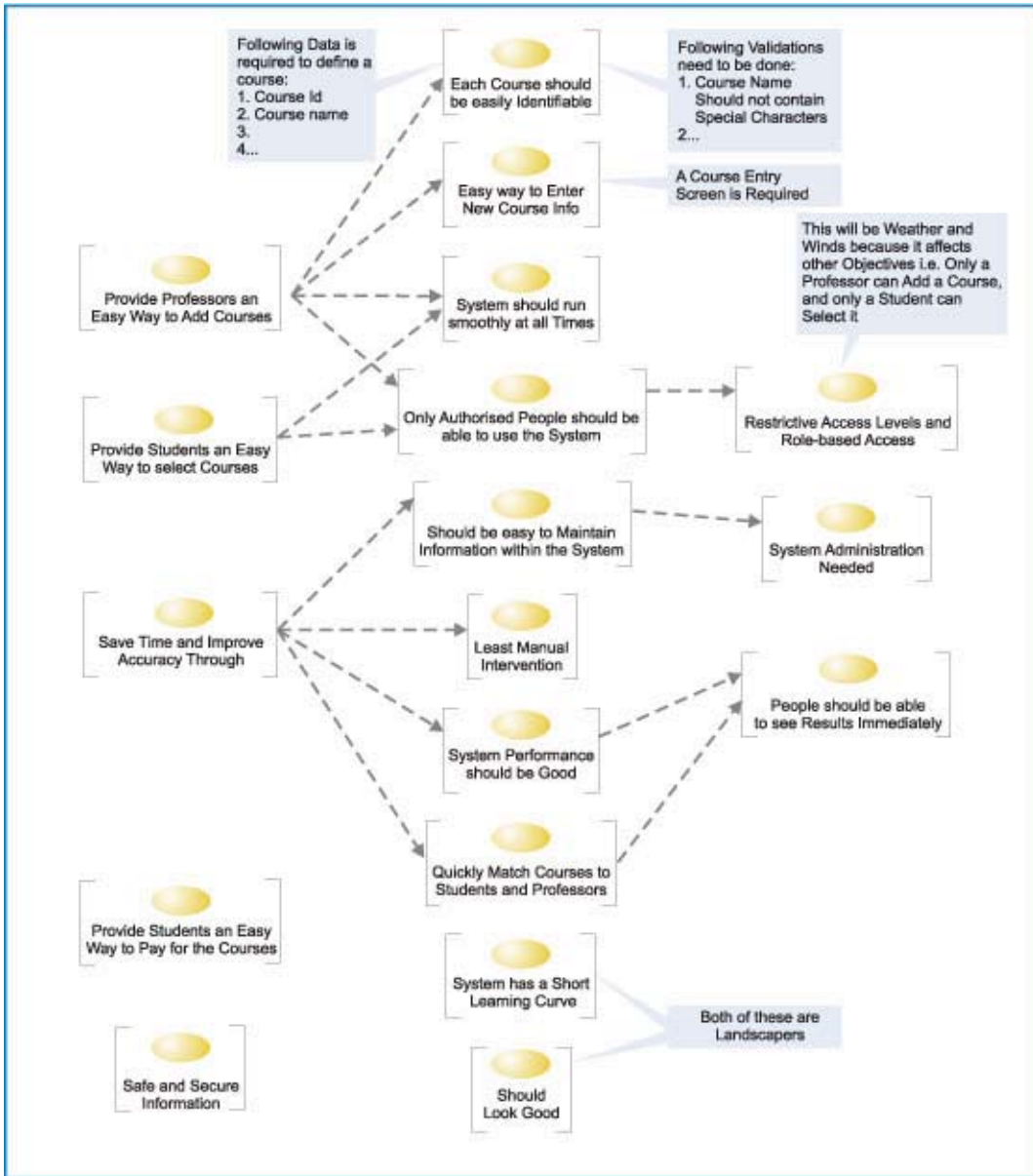


Figure 2: Incomplete Objective-Linked Diagram for the Project Source: Infosys Research

the Objective-linked diagram is in a partial state of completion.

- Weather and Winds:** This step defines the effect of other business functions on the business function in question. It involves

assessing how other business functions plan to synchronize their objectives with the objectives of the business function in question. Again, we will gather the relevant objectives of other business functions as the hidden objectives of

our business function. See Figure 2 for objective called “Restrictive access levels and role-based access.”

- **Coercers:** This step will include constraining objectives i.e., the constraints under which the objectives of the business functions need to be achieved, for e.g., regulatory constraints, people constraints, etc. See Figure 1 for adherence to the regulation - safety of sensitive data.
- **Landscapers:** This step includes objectives that act as differentiators (or rather add-on benefits) against competition – see the objectives placed in the lowest end of Figure 2. These might go unnoticed in the previous steps as these are not core to the overall objective. Hence, the need for this step. In fact, these are mostly nice-to-haves from the business function point of view, but sometimes these objectives become must-haves, especially when they are instrumental in customer acquisition. But in such cases one would expect them to be discovered in the fourth step Weather and Winds.
- **Re-comrades:** This step includes redefining the relationships so as to include all the objectives that have been identified by the OLAR team. This step will lead to validation and correction of objective-relationships that were defined in step 3 and also establishment of new/additional relationships that are updated in the objective-linked diagram. This is more like rechecking the objectives and their relationships before we call them final.
- **Estimator:** This step estimates the

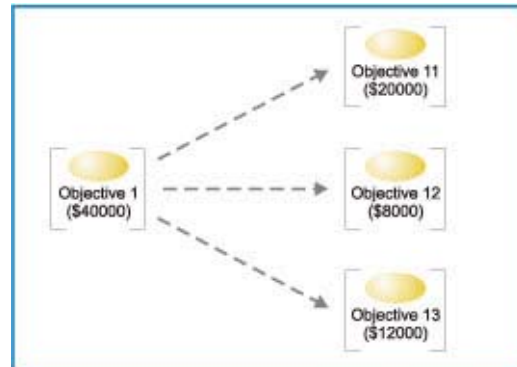


Figure 3: Estimator Example

Source: Infosys Research

approximate cost of fulfilling each objective. Here, it is important to understand that since we have a tree-like structure, the cost of an objective will be summation of costs of the branching objectives [Fig. 3]. Note that OLAR advocates use only FP estimation technique for such estimation. The estimated numbers – either FP count or effort or cost – can be depicted in parentheses suffixing the objective name. Again, the collaboration diagram is updated to depict this information. It is worth mentioning here, that OLAR method also helps in easy modularization on the basis of objectives. Objectives can also be chosen at a particular degeneration level as the level of modularization. The level of modularization can be decided upon the OLAR team as per their comfort.

- **Budgeter:** This step includes prioritizing the objectives. It involves assigning a rank to the objectives. Here we will rank the objectives serially as 1, 2, 3, 4 and so on. This step will provide us multiple views (on the basis of ranks) of the system under construction. This means that one can also

Entry Criteria	Process Steps	Exit Criteria
Training on OLAR Ideology and Methodology	Step 1: What and Why (Identification of Objectives to Lowest Level)	Business Analysis or Requirements Document
	Step 2: Objective Supporters (Data + Validations that Support Objectives)	
	Step 3: Comrades (Identification of Relationships between Objectives)	
Domain Training on the Business Functions	Step 4: Weather And Winds (Cross-function Influence on Objectives)	Acceptance of OLAR Artifacts by Stakeholders
	Step 5: Coercers (Constraining Objectives)	
	Step 6: Landscapers (Objectives that are Like Add-on Benefits)	
Clarity on Objectives of Business Analysis Exercise	Step 7: Re-comrades (Re-define Relationships between Objectives)	
	Step 8: Estimator (Estimating the Cost of Each Objective)	
	Step 9: Budgeter (Prioritization of Objectives)	
	Step 10: Concluder (Prepare the Requirements /BA Document)	

Table 1: The OLAR Process

Source: Infosys Research

determine the cost of each view and finally zero in on the view that fulfils requirements and is within the budget too.

- **Concluder:** This step prepares a requirements or business analysis document by explaining the in-scope objective-linked diagrams in detail and the out-of-scope ones in brief. So, each collaboration diagram followed by its explanation will form the requirements document.

Table 1 gives a quick view of the OLAR process.

OLAR METHODOLOGY

OLAR works by preparing questions based on objectives. Starting at a very high level and based on the understanding of the business, the approach drills down further into lower level objectives through questioning each question and so on so forth. It also questions what support is needed to realize the objective and finds the way to ascertain that the support is flawless (data validations). The business analyst continues with

these questions for several iterations, refining answers and gaining a better understanding with every iteration. Iteration is advisable to the extent of not frustrating the stakeholders.

Use UML tool or any other tool (e.g., Visio or Rational Rose) that helps in presenting the objectives in the way one presents objects in Object Oriented Programming and Systems (OOPS).

Objective linking diagrams should be used as a means to understand the requirements better. All requirements should link to one of the objectives and hence drawing a traceability matrix should be the right way to go. If there is a missing link then either an objective has been missed or a requirement has not been captured. It is also possible that a requirement or an objective might be redundant because it has already been covered as part of another requirement/objective.

ENTRY CRITERIA FOR OLAR

Like any new approach or concept it is essential to train the OLAR team so that the team understands and appreciates the ideology behind OLAR and is able to adopt the OLAR methodology successfully. This is the most important entry criteria for an OLAR exercise. The other important

thing to remember before the OLAR exercise starts is that the entire OLAR team needs to gather sufficient domain knowledge about the business functions for which the business analysis is being conducted. While talking about domain knowledge it will not be out of context to say that many a times one is astonished at the fact that the business users do not understand why they do what they do. And consequently they do not understand why others do what they do. So, there is no clarity with respect to business objectives. And since OLAR is all about objectives, domain training is a must for better comprehensibility of objectives.

And of course, as in any assignment, the business analyst ought to understand the objectives behind the business analysis exercise that she is required to take up.

EXIT CRITERIA FOR OLAR

The final result expected from the OLAR exercise is the requirements or business analysis document that details the relationships between objectives; documents data required for satisfying each objective and lists the checks needed on data to ensure that each objective is fulfilled. The exit criterion for OLAR involves successful completion of the OLAR process i.e., the final result delivered as specified in the previous section and acceptance by all the stakeholders involved.

CONCLUSION

With OLAR, the IT team as such is able to understand the requirements more easily as human mind is able to better appreciate things it can easily connect to a purpose. Whenever we read anything, there are a number of assumptions that we make sub-consciously in order to interpret and understand what we have read. These assumptions are based on our understanding of the domain of the topic and


logical patterns that our mind is able to discern. So, if the objectives are clear, logical patterns will be formed and hence the assumptions would be correct. This would in turn act as a defect prevention strategy as the requirements related assumptions are correct and hence the requirements related defects are bound to reduce. Since the cause of correcting defects increases with lifecycle stages, for instance, from requirements to maintenance, this cost rises 100 times, so this will go a long way in reducing the cost-to-project and in turn cost-to-organization.

With OLAR, business users and other stakeholders will enjoy their work better since they will understand the objectives of the various activities they perform and the roles and responsibilities that are assigned to them. Moreover they will also be able to appreciate their own importance in the entire context. And since this is done through a discovery mode the results are much better. Again, this leads to increase in productivity levels and hence brings down cost to organization.

Since the overall understanding of business and objectives is much better and complete using OLAR, estimating the cost of the project is easier and more accurate. The possibilities of inflated budget allocations or spawning out of new projects that get discovered during the execution of the original IT project are also much lesser. Moreover, since OLAR has an inherent feature of prioritizing objectives, it is easier to fine tune the budget as per the priority of the objectives. Thus, the need for ad hoc re-prioritization or unplanned increase in the opportunity cost is much lesser.

A Standish Group Chaos Report stated that more than 66% of the software projects delivered were challenged or were a failure due to the lack of a robust requirements process [8]. OLAR can turn this around and overcome the gaps in the requirements process.

REFERENCES

1. Terry Quatrani, *Visual Modeling with Rational Rose 2002 and UML*, Addison-Wesley Object Technology Series, 2002
2. Barry Boehm and Victor R Basili, *Software Defect Reduction Top 10 List*, IEEE Computer, IEEE Computer Society, Vol 34, No 1, January 2001. Available at <http://www.cs.umd.edu/projects/SoftEng/ESEG/papers/82.78.pdf>
3. Roger S Pressman, *Software Engineering: A Practitioner's Approach*, The McGraw-Hill, 1997
4. Mike Bray, *Object Oriented Analysis*, 1997. Available at <http://www.sei.cmu.edu/str/descriptions/ooanalysis.html>
5. Pankaj Jalote, *An Integrated Approach to Software Engineering*, Second Edition, Springer, 1991
6. Wendy Boggs and Michael Boggs, *Mastering UML with Rational Rose*, Sybex, 1999
7. Edward de Bono, *Serious Creativity: Using the Power of Lateral Thinking to Create New Ideas*, Harper Business, 1993
8. Latest Standish Group CHAOS Report Shows Success Rates Have Improved by 50%, *Business Wire*, 2003. Available at http://findarticles.com/p/articles/mi_m0EIN/is_2003_March_25/ai_99169967. 

Strategy Value Traceability in Enterprise Requirements Elicitation

By Luke Housego

Articulate requirements as measureable enablers of strategy with the help of a value framework

Value creation can be simply viewed as three logically distinct concerns - Why, What and How. These can be interpreted in a business context as Strategy, Requirements, and Design/Planning.

These three concerns in value creation exist within an enterprise to varying levels of articulation - exhaustively documented by a consultancy or possibly just as an idea in the mind of a decision maker or sponsor. But all three will exist, and able therefore to be elicited, logically modeled and measured against.

Each concern is logically distinct, but accountable to the preceding concern. This creates dependency in order of precedence. What is accountable to Why. If the Why changes, the What must be re-assessed. If the What changes, the How must be re-assessed. If How changes independently (say change of which contractor to use, or a low level technical protocol) the Why and What are not really impacted.

In this view, each concern's value, or reason to exist is as an enabler of the preceding

concern. Conversely, the purpose of a preceding concern (Why to What, What to How) is as the determinant of the incrementally more tangible next concern. A strategy is not value in itself, rather it represents potential value, to be moved closer to realization by a What and then How.

REQUIREMENTS IN THE CONTEXT OF STRATEGY

A view of the role of requirements is the reason for, and constraint to, design and planning. Within an enterprise context, requirements are driven by broader than execution need or intent. Ability to articulate value, for all areas of a program of change is important due to the dynamic nature of the process of creating value. Totality of information is never available prior to planning or executing. Change in recognition of need is a certainty. As a result, value based decisions are required constantly.

Traceability of elements of design to defined requirements elicited from an agreed authority at a point in time does not sufficiently



Figure 1: Context of Strategy within the Ongoing Dynamic of Communicating Value Creation **Source:** Infosys Experience

advise strategic value driven decisions of impact. At an enterprise level, decision makers have both authority and willingness to change requirements as needed, to achieve what is of real value to them – their strategy. They are not valued by *on time or on budget or at quality* parameters. They are valued by how well they enable achieving strategic objectives.

To traceably articulate requirements in a context of strategic value, requirements elicitation must devise means of accountability to a logical model of the strategy. The requirements elicitation function must know what it needs from, and if necessary be able to create, this logical model. This capability requires familiarity with how strategy can be articulated and representing its value within a program of change.

STRATEGY FUNCTION FROM A REQUIREMENTS MANAGEMENT VIEW

A definition of strategy for an organization is “how it intends to create value for its shareholders, customers, and citizens.” From a strategist’s perspective, strategy is a dynamic process – it does not cease and is not static. What does vary within this process is the rate of change. An organization’s mission, values and vision changes a lot less frequently than the personal objectives of the members of the organization.

Figure 1 shows the process of communicating and planning value creation. Each element of this process is ongoing and subject to change, with the rate of change incrementally higher from left to right. Again, each phase of this process is dependent and can be traceable to the prior.

With requirements management, there is not so much concern with the decisions behind which strategy was chosen or how it was chosen. Requirements point of interaction with strategy is with methods of communicating and accountability, or logical and metric modeling. This is the link point between the Why and What.

A VALUE MEASUREMENT FRAMEWORK – THE BALANCED SCORECARD

The Kaplan and Norton balanced scorecard tool is now widely adopted and can be generally regarded as a defacto standard in modeling the totality of dimensions of value in an organization [1]. It is a very simple tool, viewing an organization in four basic dimensions of value – Financial, Customer, Internal Process, and Learning and Growth.

Each of these value dimensions are made up of several sub values. The complete picture of a generic organization Balanced Scorecard is presented in Figure 2.

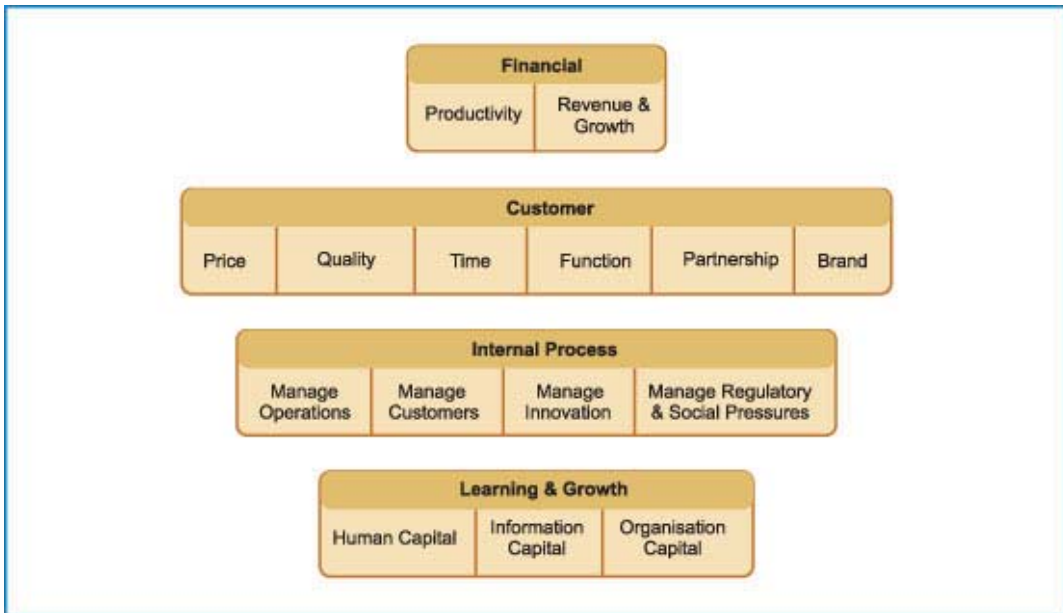


Figure 2: Dimensions of Value within a Balanced Scorecard

Source: Infosys Diagram of a Kaplan & Norton Balanced Scorecard

A balanced scorecard becomes useful to communicating *strategy* as a meta framework to link sets of objectives and their individual measures to higher level meta-measurements indicative of more comprehensive progress during or after execution. Fit for purpose scorecards are often created from the macro organizational scorecard to measure the different values of specific business units, groups, teams and down to individuals (commonly becoming an individuals KPI's). These purpose fit scorecards have a child relationship to higher level scorecards, the measures at each level influencing up, as determined by established relative strategic value.

Organizations have different emphasis on areas of the scorecard as determined by their mission, vision, values and strategy. Thus not all value is equal. When modeling this in a spreadsheet, you can assign weightings to reflect such emphasis. Some will value financial objectives very highly, others learning and

growth. All organizations will have all aspects of the scorecard. By creating methods of measuring each of the dimensions of the scorecard, change in an organization can be measured by impact to each dimension of organizational value, and then by weighting/priority to the organization as a whole.

In eliciting requirements it is unlikely your role will be to advise on strategy, and if the organization uses balanced scorecard already, you should be able to use existing strategy or planning artifacts that will contain the scorecards and objectives and measures. What becomes important in modeling the value relationship between requirements and value is logically defining the relationships of objectives and requirements [Fig. 3].

Objectives have relationships with each other. The relationships can be peer relationships, where one has a value impact upon the other, and also parent-child relationships. In the parent-child

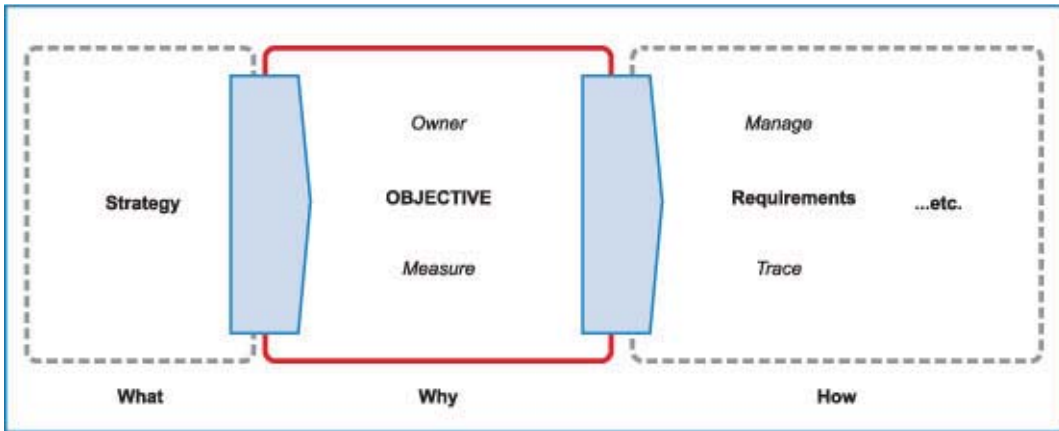


Figure 3: Relationship Structure of Objectives

Source: Infosys Analysis

relationship for objectives it is always a one-to-many relationship, never many-to-many.

A basic relationship structure of objectives in a modern large enterprise would be as shown in Figure 4.

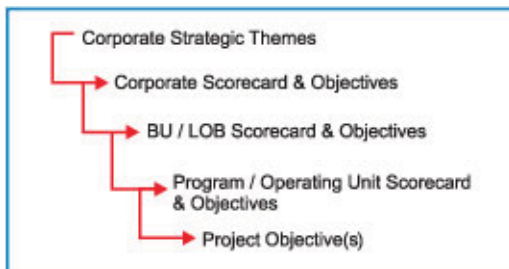


Figure 4: Relationship Structure of Objectives

Source: Infosys Analysis

Some principles to consider when forming an approach to elicit requirements are:

- The only recognizable value of a requirement is its enablement of a strategy - this is intangible and modeled through its relationship to achieving an objective
- For value to be recognized, it must be articulated
- Not articulated is not the same as not important
- A requirement with no value is not required
- Separation of concern - integrity of a strategy, is not impacted by constraints in eliciting requirements.

PRINCIPLES AND VALUES OF ELICITING STRATEGY DRIVEN REQUIREMENTS

When a strategy has been logically modeled to a value framework (such as Balanced Scorecard) it has become a measurable, shared and communicable strategy. The How of requirements engineering can be made traceable to What (objectives!) is wanted to enable the strategy.

IMPACT OF REQUIREMENTS ELICITATION TO A DEFINED STRATEGY

Requirements elicitation is the point where many elements of strategy are first thought through to new levels of detail, thus it will become a source of change or refinement. An exploration of the strategy unfolds as requirements are elicited, necessitating change in at least the lower levels of detail. Often triggered by conflict or constraint,

it is when unexpected that it is a key indicator of need to enhance the strategy definition. Merely the existence of conflict or constraint in requirements is not indicative of need to enhance a strategy definition. The intent of a strategy is not likely to change from writing requirements, but how that intent is described (usually through the objectives) may need adjustment.

Managing this process of interaction between strategy definition and requirements elicitation is important for minimization of cost impact. In requirements engineering, as with any engineering discipline, the cost of resolving a problem exponentially increases as it moves through an engineering process. Recognition of this engineering axiom's applicability to the logical definition and communication of a strategy will lead to significant reductions in problems in the later stages of value creation.

The point of highest visibility of the emergence of issues will always be the more tangible requirements elicitation rather than strategy definition. It is in the interest therefore of the requirements management function to ensure robustness and timeliness of management of the logical definition of a strategy, even when it's management (as is likely) sits within a separate strategy functions responsibility.

Ongoing attentiveness to accountability for shared processes and measures between the strategy and requirements management functions, including requirements elicitation as a feedback to the strategy definition and model is recommended to cater for the anticipated continual change in understanding of how to enable a strategy.

REQUIREMENTS: VALUE TRACEABILITY

The strategy is logically modeled by the relationship of its enabling elements - objectives. As described earlier, these objectives will have both peer and parent/child relationships. A

parent objective is fully modeled when achieving 100% of its child objectives measures completely achieves 100% of its own measures.

Peer relationships only exist where there is relevance. Relevance being success or failure of an objective will impact the actual value of the peer objective.

Objectives are not concerned with How to achieve their goal. They have a goal, and a measure to determine if the goal is met. A pure objective has no requirement beyond success.

Requirements enter the picture as a way to achieve objectives. Importantly, there are always many ways to achieve objectives. Requirements are decisions to enable optimal ways to achieve objectives and thus deliver value as measured against the strategy.

Requirements are elicited from the owner of objectives and from subject matter experts. Elicitation is only relevant from someone with tangibly identifiable decisional (objective owner or advisor) or knowledge relationship (subject matter expert) with the efficiency and effectiveness of an objective.

The totality of actual requirements needed to achieve an objective cannot be known before elicitation, and even then is certain to change. Thus it is not viable to represent value of a requirement in a fixed manner as determined at the point of elicitation (or even any other point in time). A requirements value can be better modeled as a relative function -- a calculation of its relative contribution to the value created by the objective(s) it enables at a point in time.

To model this, a requirement has some attributes like:

- Which objective(s) it enables
- Relative importance of this requirement compared to all other requirements, by each objective it enables.

Relative importance is not rating against a baseline or defined framework. For each objective, a single most important requirement should always be identified (this will/can of course change in time). All requirements are weighted compared to that requirement in importance. Rate half as important with half the metric (i.e., if using a 100 point system for importance, the most important requirement should be rated 100, something half as important should be rated as 50).

An example objective of 'reduce abandoned transactions' may have the following relevant (child) requirements:

1. Reduce user interface response time (100 – most important requirement)
2. Eliminate repeated entry of same information, (20 importance)
3. User interface branding and color scheme, (30 importance)

The formula for determining the relative contributory value of a specific requirement to a single objective is: *Relative Importance of this requirement / Total of all requirements.*

This will give a fraction that can be used as a % of the total contributory value of the objective. This is the current relative value of the requirement for this objective.

Because requirements can often contribute towards multiple objectives, to assess the total contributory value of a requirement you multiply the current relative value of the requirement for each objective by the relative weight of the objective. The sum total of all objectives' weighted current relative value is the current value of a requirement.

Determining the relative importance can be done through a number of mechanisms, including research/market analysis, subject matter expert workshops, or simply direction from a decision

maker or advisor. At the lowest level of rigor, you may just propose relative importance to the objective owner. The level of rigor applied to determining the relative importance is only required to a level for the objectives' owner to have confidence in the identified priority.

MANAGING RELATIVE IMPORTANCE

Timeliness and accuracy of the logical model of the strategy, and stakeholder buy-in to the assessed relative importance of requirements to value enablement, both require ongoing audit by the requirements management function. These are the basis of the integrity of requirements traceability and their value articulation. Tools such as Kepner-Tregoe Decision Analysis [2], Six Sigma and Quality Function Deployment (QFD) [3] can be employed as appropriate where this level of rigour is warranted.

Assessing the relative importance by the contributory value of a requirement is an ongoing, iterating activity. Some principles are helpful within this process to ensure consistency of approach to determination of values and an avoidance of bias to particular concerns or functions within the enterprise. Principles to follow in assessing relative contributory value are as detailed below:

Adequate breadth of views

For each objective, a stakeholder group will need to be formed for regular determination of appropriateness and then relative value of candidate enabling requirement. Adequate breadth across functions and disciplines in the views used to debate appropriateness and relative value should be represented.

A stakeholder group to determine value should consist of interest's representative of the strategy function, relevant business operational functions and implementation functions (i.e., program/project office and advisors).

Avoid 'bias' to normalization

The purpose of determining relative contributory value of requirements is achieved through differentiation. Treating requirements as close to equal (or the common "all are important!" statement) dilutes the usefulness of modeling, limiting ability to advise value decisions.

A technique to avoid this is constant, iterative identification and then grouping of requirements that are common across objectives, or closely related in function/purpose. Name the set and assign a single relative importance. Within the identified set assign relative importance of individual requirements to the value of the whole set.

Separation of Concern

Constraints in design, implementation or operations that can include problems and costs in resourcing, delivery risk, skills, knowledge and technical issues cannot be allowed to impact the assessment of relative importance of a requirement. A requirement is not a design. How the requirement will likely be designed or planned to be implemented is out of the concerns for determining its value. Operational or delivery focused stakeholders for example, can have a bias in this area that may require addressing through agreeing to principles as a requirement to participate in value assessment.

Where constraints are identified that will likely or intuitively impact the viability of a requirement to enable a strategy's objective, re-factor the constraint into a separate requirement and assign a relative importance to the enablement of the value of the objective. Do not overestimate the importance of this concern that is now a requirement to the delivery of the value of the objective. Intuitively it is unlikely to be as important to value creation as the original value creating requirement.

Cost (or other constraints) should not be a concern in deciding how important a specific requirement is to the value created by an objective. Artificially building a cost/benefit assessment into the elicitation process is inappropriate for this stakeholder group, and it will bias design, planning and value proposition assessments, all which will be consumers of the elicited requirements value traceability.

Requirements traceability is not balancing the value delivered with the cost of delivering it. The traceability to strategy enablement is only one of many inputs into these decisions on cost/benefits and value assessment. These are appropriately managed by separate processes or functions within a program of change.

CONCLUSION

Enterprise requirements elicitation occurs within a context of value as established by a strategy. For requirements to be traceable to the desired value - the strategy through the relationship of the objectives that enable it must be logically articulated, modeled and maintained against a value framework.


As the nature of requirements is one of change throughout the lifecycle of the requirements elicitation process, their value can be more readily seen as a function (rather than a static value) of the sum of the relative contributions to the elements of the logical strategy model that they enable.

Possessing a capability to articulate the relationship of requirements to a strategy and model their contributory value is a powerful tool for programs of change. It gives insight to tangibly advise on impacts to value of requirements change.

Every large program of change will need to assess change to requirements and advice of the impact to value. Confidence by those

responsible to achieve objectives in the rigor, depth and responsiveness of this advice can be a defining element in their relationship with the requirements management function.

REFERENCES

1. Robert Kaplan and David Norton, *Strategy Maps: Converting Intangible Assets into Tangible Outcomes*, Harvard Business School Press, 2004
2. Kepner-Tregoe *Decision Analysis*, 2008. Available at www.kepner-tregoe.com
3. Thomas Pyzdek, *The Six Sigma Handbook, Revised and Expanded: The Complete Guide for Greenbelts, Blackbelts, and Managers at All Levels*, McGraw-Hill, 2003
4. James D Miller, *Game Theory at Work, How to Use Game Theory to Outthink and Outmaneuver Your Competition*, McGraw-Hill Companies, 2003
5. J Zachman, *Enterprise Architecture : A Framework*. Available at www.zifa.com. 

A Novel Approach to Software Requirements Analysis

By Anjaneyulu Pasala PhD and Ravi Prakash Gorthi PhD

Behavioral slicing of requirements is a path-breaking method to discover ambiguities and incompleteness early in the SDLC

Getting functional requirements right is one of the most important goals of requirements analysis. Failure to deliver successful software is attributed to the erroneous gathering of functional requirements and analysis. Karl's research report on inspecting requirements suggests that approximately 50% of software product defects originate in requirements [1]. Hence, functional requirements analysis is arguably the most important aspect for the success and effectiveness of software projects. The three important activities of requirements analysis are elicitation, analysis and documentation of requirements. Thus developed requirements describe how the envisaged system should behave or interact with its intended agents/users in the defined environment. Therefore, it is important to identify, analyze and document interactions between the systems to be developed and agents/users of the systems.

In case of functional requirements gathering, the interactions are termed as implied scenarios because they are typically

system-input and system-output in nature [2]. Validation of these implied scenarios support the requirements elaboration. Currently, scenario-based specification languages such as Message Sequence Charts are popular for specifying scenarios that describe patterns of interactions among entities and thus are useful for capturing low level interactions among entities [3]. However, these charts are non-standard and are difficult to represent the user level requirements of a complex system.

Since the late 80's, Object Oriented Analysis and Design (OOAD) has been playing an important role in software development [4] [5] [6]. OOAD offers a set of models like - functional/ logical, static and dynamic models. However, with the introduction of use-case models as a part of UML [7], rational unified process [8] and essential unified process [9], the requirements analysis changed drastically. It has often been observed that use-case model is an effective technique for functional requirements analysis and it drives the design and implementation

of the system. Use-case models also majorly improve the communication between customers and software development teams. Also, the dynamic behavior of the system for each use-case is modeled using another UML model called use-case activity diagram (UCAD). One of the concerns during the use-case modeling is its completeness for a given complex software project. Though the use-case model based requirements analysis offers rich modeling capabilities, it does not sufficiently structure the functional behavior of the system. Also, we have researched and found that the above methods do not offer a mechanism to extract, analyze and structure the units of behavior from a complex behavior of a to-be-developed software system.

The objective of our approach is to structure the requirements in a form that can be presented to users to get the right feedback and validation. On similar lines, we propose a novel technique to structure the UCAD so that they are effectively used for elicitation, analysis and documentation of the software system's functional requirements. The proposed approach is based on a novel concept called *behavioral slicing using unit of behavior* [10]. Unit of behavior is defined as a tuple consisting of user/system input, followed by system processing and finally the system output under various conditions. This approach has been used in a few live projects and it has been observed that it effectively captures and analyses the system functional requirements. There are also some important additional benefits from our approach to software development teams as discussed in the later sections.

OUR APPROACH TO REQUIREMENTS ANALYSIS

Our approach to functional requirements analysis consists of the following steps:

- 1) **Identify Use-cases:** Initially, identify high level use-cases for all the external actors of the envisaged system [4].
- 2) **Build Structured Activity Diagrams:** For each use-case, develop the structured activity diagram. Each complex function should be decomposed into an ordered sequence of units of behaviour, using 'behavioral slicing' technique (as detailed in the next section). All exceptions are also included in the activity diagram for a given use-case scenario so that, it is complete in functionality.
- 3) **Validate Requirements:** The users or clients will review the structured activity diagrams (developed in step 2) and validate the requirements, including the exception flows. Refine the activity diagrams as required by the end user to incorporate the suggested changes/feedback.

STRUCTURING UCADS

This section explains in detail on how to structure UCAD using *behavioral slicing* technique to facilitate a better requirements analysis.

Brief Introduction to Activity Diagram

An activity diagram effectively captures the behavior of a system being modeled and is fairly easy to construct and understand [7]. An activity diagram consists of nodes and edges between these nodes. Nodes represent the various user actions, system outputs, system processes and conditions and edges represent the transitions from node to node. The activity diagram has five nodes:

- Initial node
- User Action node
- System output/System process node

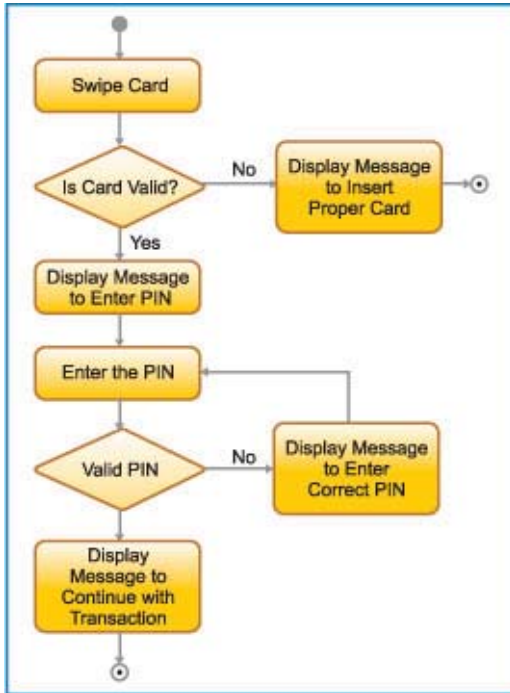


Figure 1: Activity Diagram for ATM

Source: Infosys Research

- Condition node
- Final node

In addition to these essential nodes, the activity diagram has two important nodes, viz., fork and join, to represent the parallel activities of the system. Other than the initial node, each activity diagram will have one or more instance of each type of node. Figure 1 shows an activity diagram of an ATM PIN validation, drawn using one of the UML based tools.

Introducing Unit of Behaviour

In this sub-section, we describe the novel concept called *behavioral slicing using unit of behavior* to develop and structure the activity diagrams. The system functionality is characterized as a collection of units of behavior. Each unit of

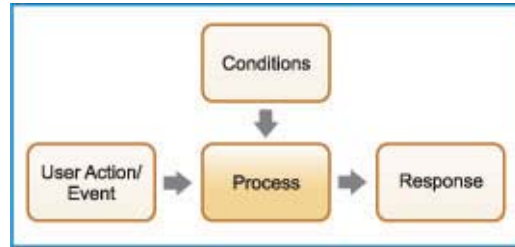


Figure 2: Unit of Behavior in General

Source: Infosys Research

behavior is seen as a system response to the user actions under certain specified conditions, as shown in Figure 2 [11]. Therefore, each complex functional requirement is structured as an ordered sequence of units of behavior. A unit of behavior is a tuple $\langle UI, SP, C, SO \rangle$, where,

UI: User Inputs

SP: System Process

C: A Set of Conditions on System State

SO: System Output

In an activity diagram, the components of the unit of behavior are: (1) user action node, (2) system processing node, (3) conditions node and (4) system output node. Conditions nodes are optional for each unit of behavior being specified. Therefore, each use-case behavior is sliced into a sequence of units of behavior.

To illustrate this idea consider the use-case 'PIN validation' in the ATM system. One of the units of behavior in this case is:

UI: user swipes the ATM card

SP: system reads and validates the card

C: if the card is valid

SO: system displays 'Enter PIN Number' message.

One will find that there are four units of behavior in this use-case that are shown in Figure 3 overleaf. The unit of behavior technique facilitates the user to decompose a complex use-case into a sequence of sub-use cases. This

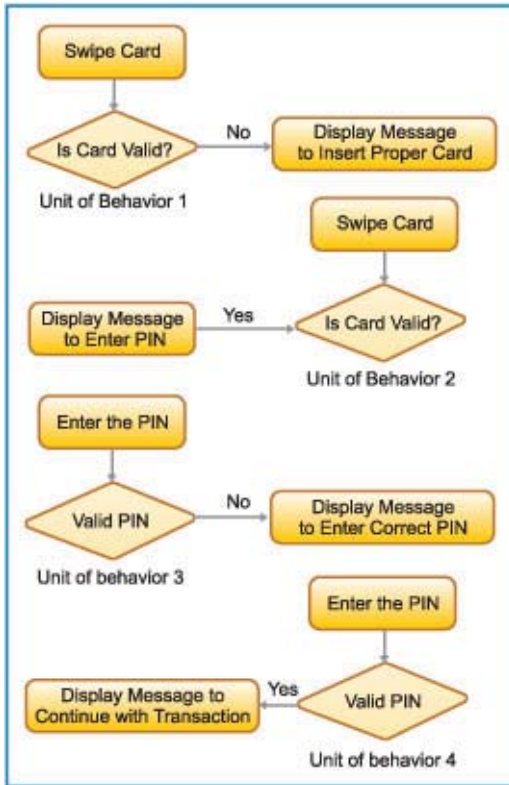


Figure 3: Example of Units of Behavior
Source: Infosys Research

decomposition also facilitates the exceptions to be recorded in the form of units of behavior.

CASE STUDY TO ILLUSTRATE OUR APPROACH

Consider the case study on course registration system of an academic institute to illustrate the effectiveness of the proposed approach to requirements analysis.

Requirements Elicitation: Using the course registration system, the registrar sets up the curriculum for a semester. One semester program of a curriculum will have multiple course offerings. Students should select 4 primary courses and 2 elective courses. Once a student registers for a

semester, the billing system is notified so that the student can be billed for the semester. Students can use the system to add / drop courses for a fixed period of time after registration. Professors use the same system to receive their course rosters. Users of the registration system are assigned passwords that are used during login, for validation.

Requirements Analysis Using our Approach: The complex behavior captured by above requirements are analysed to identify the sequence of behavioral units. Initially the use-cases are found for all the external actors: *Student, Registrar and Professor*. The set of use-cases in the example system are: (1) Register for a course, (2) Authenticate User (login validation), (3) Maintain course information, (4) Request course rosters, etc. The use-case, *Register for a course* is analyzed using the *behavioral slicing* technique and is decomposed into the following sequence of units of behaviour:

Unit of behavior # 1: (UI: student selects the academic semester he / she is in, SP: system validates student’s request, C: if valid then SO: system displays available courses for the selected semester).

Unit of behavior # 2: (UI: student selects a course, C: system verifies whether the pre-requisites are satisfied and if NOT then SO: system displays the message “pre-requisites are not satisfied”).

Unit of behavior # 3: (UI: student selects the course, C: system checks whether (the pre-requisites are satisfied) AND (the course is NOT a primary course) AND (the student has NOT selected more than two alternate courses) and if so then SO: system displays the message “You (the student) are added to class” and also displays the initial menu).

Similarly, one can extract other units of behavior of the same use-case as documented in Figure 4.

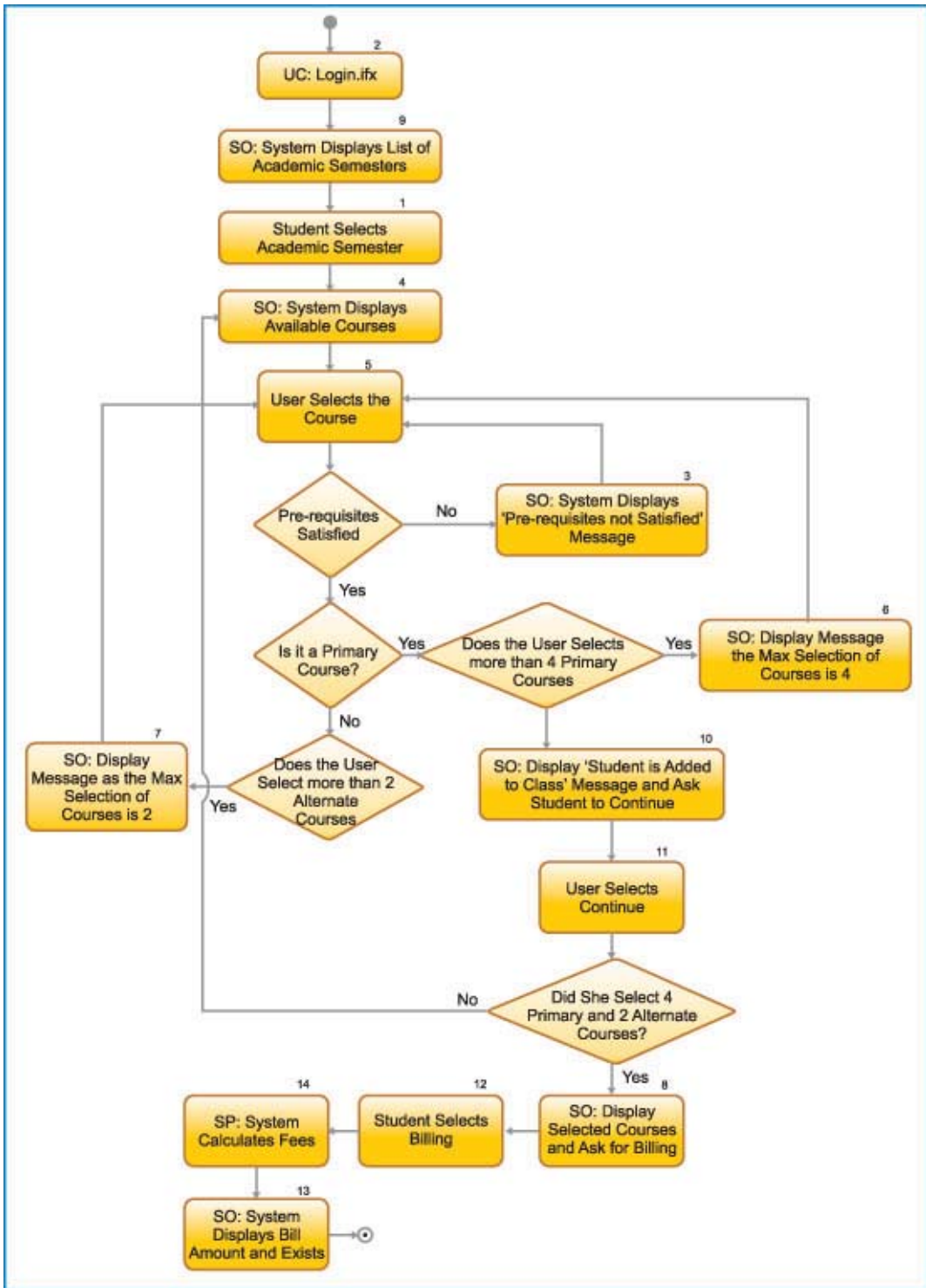


Figure 4: Structured UCAD for Course Registration Use Case Source: Infosys Research

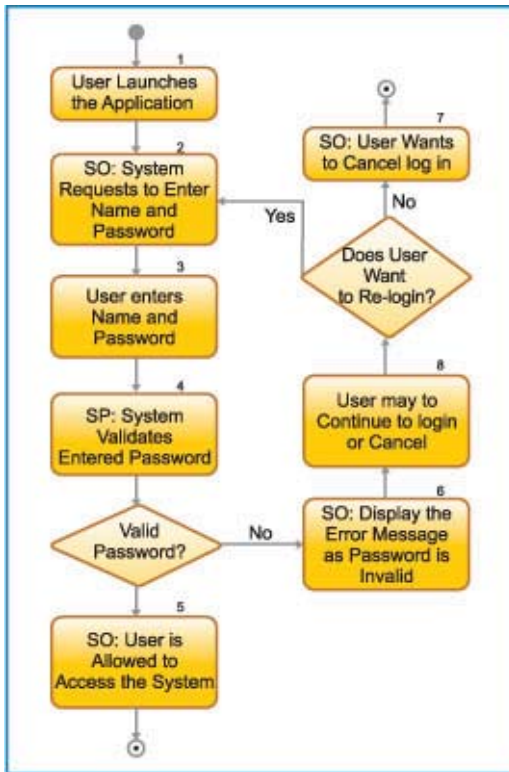


Figure 5: UCAD for User Login Use-case
 Source: Infosys Research

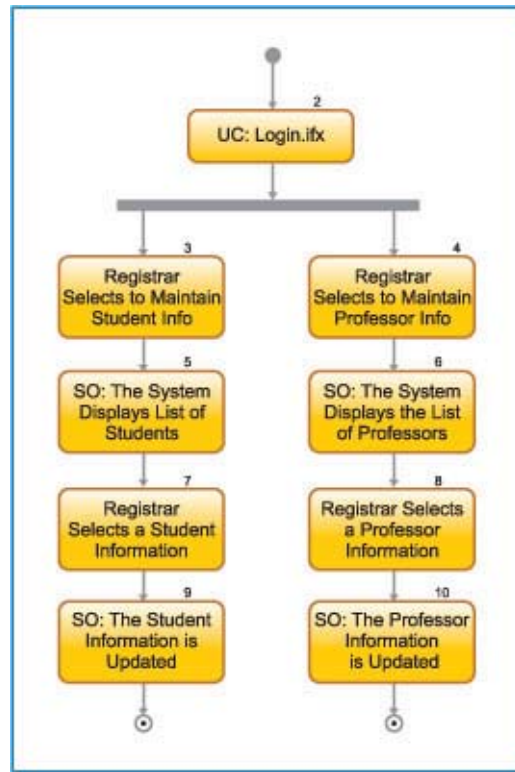


Figure 6: UCAD for Maintaining Course Information
 Source: Infosys Research

Further, one can build the structured UCADs for the remaining use cases of this case study, as shown in Figures 5 and 6.

Requirements Documentation: The thus developed structured UCADs will form the documentation of the system requirements (we can use one of the UML based tools such as IBM Rational Rose to document the requirements).

The functionality of use case *Register for a course* includes the *user authentication* use case functionality. *User authentication* use case functionality is shared among several use cases. Therefore, our approach proposes a creation of separate UCAD for shared use case and the same diagram is referred in other UCAD

through the concept of linking of UCADs using the file name. In our example, login.ifx file represents the structured behavior of *user authentication* use case and the same file is linked in the other two activity diagrams. These are shown in Figures 4 - 6.

BENEFITS OF OUR APPROACH

In addition to getting the right functional requirements of a given system, our approach has the following benefits:

- It facilitates the validation of requirements at the requirements analysis stage itself. Thus, identifying gaps in the requirements at an early stage leads to saving of costs.

- It facilitates automatic generation of UCADs by structuring the textual representation of use case as described in [10].
- It facilitates automatic generation of test cases [10]. Each unit of behavior needs to be validated. Therefore, each unit of behavior is extracted as one test case step and each path starting from initial node to final node consisting of several units of behaviors is extracted into one test scenario. The number of test scenarios is equal to number of paths in the activity diagram. UCAD shown in Figure 5 generates 3 test cases consisting T1 - 1->2->3->4->5, T2 - 1->2->3->4->6->8->7 and T3 - 1->2->3->4->6->8->2->3->4->5, where T1, T2 and T3 are test cases and numbers 1 to 8 represent node numbers.
- It effectively facilitates change management by automatically recommending regression test cases given two versions of activity diagrams of changed software [12]. It can identify the changes made to functions in new version of the activity diagram, based on the impact of these changes and recommends regression test suite to validate these changes. It also identifies the changed paths and generates the test cases as explained previously. To facilitate the identification of changes to nodes a concept called node numbers or ID are introduced.


CONCLUSION

Behavioral analysis consisting of user input and system output will help find errors in functional requirements as it helps the system analyst to adopt a systematic and structured approach to

gather functional requirements. This structured approach also helps clients to understand the requirements and exceptions easily so that they can validate the functional requirements. There are several benefits that our approach provides the development team across all phases of software development lifecycle such as automatic generation of test cases and regression test case generation.

REFERENCES

1. Karl E Wieggers, *Inspecting Requirements*, 2001. Available on <http://www.stickyminds.com>
2. Emmanuel Letier et al., *Monitoring and Control in Scenario-Based Requirements Analysis*, International Conference on Software Engineering (ICSE'05), 2005
3. David Harel and PS Thiagarajan, *Message Sequence Charts*, 2003. Available at http://www.comp.nus.edu.sg/~thiagu/public_papers/surveymsc.pdf
4. Grady Booch, *Object-Oriented Analysis and Design with Applications*, 2nd edition, Addison-Wesley Longman, 1994
5. James R Rumbaugh et al., *Object-Oriented Modeling and Design*, Prentice Hall, 1991
6. Ivar Jacobson et al., *Object-Oriented Software Engineering-A Use Case Driven Approach*, Addison-Wesley, 1992
7. Dan Pilone and Neil Pitman, *UML 2.0 In a Nutshell*, O'Reilly, 2005
8. Philippe Kruchten, *Rational Unified Process - An introduction*, Addison-Wesley, 1999
9. Ivar Jacobson et al., *EssUP: The Essential Unified Process - An introduction*. Available at <http://www.ivarjacobson.com/products/essup.cfm>
10. Ravi Gorthi and Kailash KP Chanduka,

- Model-Based Automated Test Case Generation, SETLabs Briefings, Vol 6, No 1, 2008. Available at <http://www.infosys.com/research/publications/toc-software-validation.asp>
11. Paul Gerrard, Testing Requirements, EuroSTAR, Belgium 1994. Available at <http://www.gerrardconsulting.com/TestReqs/TESTREQS.html>
12. Ravi Gorthi et al., Specification-based Approach to Select Regression Test Suite to Validate Changed Software, Asia Pacific Software Engineering Conference, 2008. 
-

Requirements Management in Multi-project Environments

By Manideepa Chatterjee, Sandeep Anand M, Tejal Prakash Nemane

Identifying requirements overlaps in a multi-project scenario can successfully reduce time and cost

Ever wondered why funds earmarked for IT developments fall insufficient? How does a firm cope up when a development project of an IT department grows complex and involves numerous cross projects and cross application dependencies? Corporate houses are increasingly making sizeable investments in their IT departments to align their enterprise systems to a strategic architectural blue print for business benefits but are they adequately prepared for the challenges that lie within?

Projects under such programs most often think, plan and execute their business focused solely on their individual project and application limitations. Given the scenario, the challenge for business/system analysis is not merely to provide a holistic and end-to-end picture of the enterprise in its to-be state but also to effectively manage the gaps and overlaps in parts of business requirements spread across the program.

This challenge can be met by implementing effective requirement management strategies in the *requirements planning* and *elicitation* phase to

produce a suite of requirements that seamlessly represent a single business objective of the program. Also, communicating impacts of these sets of business requirements to enterprise architecture helps in creating an end-to-end picture of the to-be state.

Typically requirements management involves identifying stakeholders and resources and defining activities for requirement elicitation adhering to existing standards of the organization. However, in a cross-project-dependent environment, since a high level business goal might percolate across projects, the business goal needs to be analyzed. Issues to be considered here are the number of projects involved, the criticality and dependency between the projects, integration feasibility in requirements and also different strategies that can be implemented. Some of the key issues in such multi-project environments are scope definition, schedule dependencies between projects, communication and enterprise architecture blueprint adherence.

This paper explores challenges and strategies for requirement management in a multi-project environment and also looks at a case study of one such cross-project dependency-heavy program in a Fortune 100 healthcare company and provides key guidelines to a business analyst on ways to tackle the roadblocks faced therein.

The key question here is – *What should the requirements management strategy be in order to deliver business requirements in complex multi-project environments?*

STRATEGIES TO APPROACH CHALLENGES

As per the workflow in Figure 1 a combination of strategies can be adopted for requirements management in multi-project environment. These strategies will help to tackle challenges in scope and schedule dependency, as well as issues of missing or overlapping requirements.

Initial activities in the analysis phase include defining business objective of the IT program and identifying all the different project stakeholders. It also involves performing impact analysis of the as-is state of the systems to identify and setup risk/issue mitigation strategies considering the level of cross-dependency between the projects. It is equally pertinent to draw links, if any, between these projects and identify overlaps and gaps in requirements. The next phase involves defining scope and schedules of the identified projects. Collating and handling common requirements forms the next step. This includes eliciting requirements with each project's stakeholders, followed by a combined business review and approaching common requirements through generalization and specification. During each stage of the process, different project teams spread across the globe have to be communicated effectively.

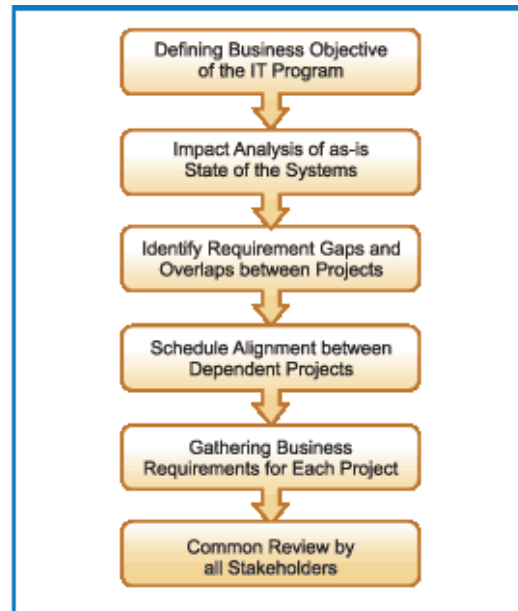


Figure 1: Business Analysis Workflow in Multi-project Environment

Source: Infosys Experience

INITIAL ACTIVITIES IN THE ANALYSIS PHASE

It is important to understand the business objective of an IT program. This knowledge, combined with that of the as-is state of the systems involved will help identify the systems impacted. It will also give a high level idea of the extent of impact and cross-dependency between applications and projects.

The way proposed to do this would be to identify stakeholders and hold the initial requirements gathering sessions with each of the project/system teams separately, followed by requirements elicitation with the subject matter experts of each of the projects/systems involved. Stakeholders represent anyone materially affected by the outcome of the project. Requirements documents need to be maintained individually till stakeholders of each of the projects /systems sign off. During this phase the overlaps of requirements' scope should be identified. Also

gaps in requirements should be covered by ensuring that all desired business functionalities are in scope for at least one project or the other.

The business analyst should then be able to plan a complete set of requirements activities such that the result is a clear, concise set of requirements on which the solution can be based. The resource types required to complete each activity also need to be defined.

SCOPE AND SCHEDULES OF THE IDENTIFIED PROJECTS

The primary goal and high level scope of the projects involved should be identified and demarcated to be elaborated later as per the format of requirements. When multiple projects are involved, initial kick-off meeting needs to be held with the stakeholders of each project individually, to know expectations. There could be stakeholders outside the project team like business owners who should be included in these communications. All the individual goals and high level requirements can be collated under a single charter and then discussed in subsequent meetings, where all the stakeholders are represented. These meetings will serve to eliminate differences or integration issues and throw a single project charter with all the high level requirements, listed along with the prototyping that all stakeholders will agree to.

As the high level requirements listed under the single charter start developing into more detailed ones, more often than not, these tend to show dependencies on each other. The requirement of one project might be dependent on the outcome of the analysis or detailed requirement of another project. Also, the successful release of one project will determine the next release schedule of the next set of requirements. This ripple effect mandates that the schedules of the projects are aligned such that each project gets a timely input from the other, if required.

APPROACH TO COLLATE AND HANDLE COMMON REQUIREMENTS

Successful requirements engineering in a multiple project environment requires identifying common requirements. One of the most effective ways to approach the common requirements is through the specialization/generalization hierarchies as shown in Figure 2 overleaf. These hierarchies can further divide the supersets to subsets of requirements. Generalization is the process of identifying and grouping the common requirements that share the same features across different projects involved. Also, these common requirements, completely or partially, address the required functionalities of multiple projects. So they can be grouped as a set of common requirements and more generalization can result in more expedited and cost effective implementation of multiple projects. The other phase is the specialization that collates specific requirements to each of the projects involved. This acts as the subset and each subset will have a superset of general or common requirements. The whole process defines modularity. Some of the requirements in the subset will have connection to the common requirements and the connections or dependencies need to be drawn out before the design and implementation phases.

Some challenges along with their solutions when handling common requirements are as depicted in Figure 2.

In some projects the set of common requirements turn out to be specific requirements as a result of constraints like implementation criteria, change in scope, etc. In such scenarios, the common requirements should be taken out of the superset and moved to the subset, and the existing relationships vis-à-vis other projects need to be altered. This will mean additional effort and change in timeline that needs to be considered as well.

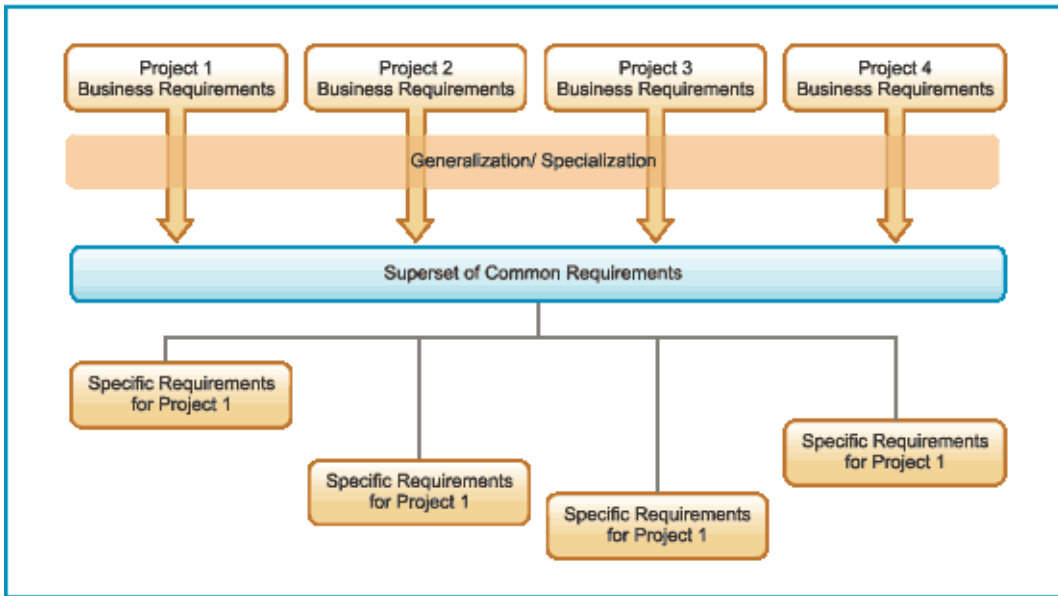


Figure 2: Approaching Common Requirements

Source: Infosys Experience

At some point the subset of specific requirements for each project might turn out to be common requirements with one or more projects involved. In such a scenario a one-to-one or one-to-many relationship needs to be established with the common requirements of other projects. This will mean a detailed analysis at each project level viz., the common requirements the project they share the functionalities with, release timeline of all the linked projects and the corresponding dependencies that need to be carried out and documented.

There might be situations where two projects with varying timelines have some common requirements. This poses a major challenge as same functionalities need to be implemented at different stages of the linked projects. In such a situation, the common requirements can be implemented together, as both the projects are in the same phase with a little variance in release dates. If the two projects have

a large difference in timelines, then they need to be implemented separately, though the design and build effort will be minimal.

In order to minimize efforts, a link needs to be established between multiple projects and a relationship model needs to be etched through requirements reviews. This model should reflect the link, the effect of breaking the link, repercussions of changes in the projects, commonalities between applications and the functionalities they serve. This can be considered for similar overlapping projects. Each new project needs to find a set of requirements that can be moved to the superset of common requirements.

Once the relationship model is created, projects need to be evaluated in the light of this model and an assessment has to be done whether the functionality will be thoroughly covered by a particular common requirement implementation.

EFFECTIVE COMMUNICATION BETWEEN MULTIPLE PROJECT TEAMS

The requirements of all projects and the multiple inter-relationships of project management functions are analyzed, rationalized and integrated to determine agreed and achievable program objectives that align to organizational goals, strategies and objectives, as stated during strategic planning. Impact of the external projects should be analyzed with the help of all stakeholders and subject matter experts.

Requirements elicitation meetings should involve all the business teams. The meetings should be tracked through minutes and action logs and circulated in all the teams for their confirmation. Whenever there are further meetings scheduled with the multiple project teams the action log needs to be updated on call and a confirmation be documented. Also, whenever there are changes to the common requirements for instance, common requirements becoming specific ones or conflict in release dates – in such cases, separate meetings need to be arranged with all the affected teams to discuss ways to handle issues or changes. The changes need to be documented and the sign-off be obtained from the respective project stakeholders to ensure that there is no misinterpretation or gap. If there is any risk or issue that arises in scope, timeline or budget due to the change, it needs to be raised against the assignee and communicated to all the teams involved in the change. This will ensure a smooth change management even when the project has moved to a different phase. The requirements traceability matrix should be used till the development cycle, as it helps to track whether each requirement is mapped against the design and testing work object.

A REAL TIME REQUIREMENTS MANAGEMENT PROBLEM

This case study describes the requirement management, risk and issue management techniques adopted by multiple project teams managed by different portfolios, to successfully implement a Health Maintenance Organization (HMO) based healthcare product on two different applications of a healthcare enterprise.

The new product was designed to be sold in the fall of '07. The program included multiple projects to support the new HMO based product on different domains and systems to be implemented in various releases. We were required to implement projects for upstream domains like *plan sponsor* set up and *member enrolment* in the earlier releases and then implement projects for the *claims domain* and other downstream systems.

The primary focus of this program was on two projects, one owned by the plan sponsor domain and the other owned by the claims domain. These two projects had to support the new HMO based product on a plan sponsor domain application or membership system and a claim domain application or claim reporting system. Membership system and claim reporting system had a lot of business processes that were inter dependent and led to a cross-domain impact.

The initial plan was to implement the product on Membership System (MBRS) prior to Claim Reporting System (CRS), since some of the claim business processes were dependent on plan sponsor data feeds. But as the project progressed, a major risk got introduced when the plan project was found to be behind schedule. Accordingly a decision was taken to implement plan and claim projects in the same release.

SOLUTION

A flowchart was developed to identify overlapping/dependent business processes that

the list of impacted applications as these were part of the overall IT program that was implementing the new HMO based product across the enterprise. Also, these upstream systems had changes as part of some other separate projects under the overall program.

Some of the CRS and MBRS business processes were conjoint. As each of the projects were following separate paths, the common impact on both the projects were overlooked assuming that the changes will be taken up by the other project. Setting up joint discussions between the projects was a successful idea since it:

- Immensely helped the project teams in creating the flowchart that demarcated the changes in each project. It also helped the development and testing teams to understand the common impact on both the projects and thus design and test accordingly.
- Identified the impacts to the MBRS and Claims Reporting System (CRS) processes due to changes in the upstream applications. The CRS project team was not aware of the changes to the plan upstream systems until it was revealed in these joint sessions with the MBRS project.
- Helped the projects in tracking testing interfaces required for integration testing.

The CRS application had a business process called 'New York Health Care Reform Act (NYHCRA act of 1997) State text file generation' and the inputs to this process were files from the MBRS and CRS process. Requirements for the state text file generation were common to both the projects. Since the schedules of the projects had similar timelines, a risk was introduced

when the NYHCRA requirements from CRS team were ready for review and the schedule of the MBRS project was slipping. This was noticed when the requirement specifications of the NYHCRA project were passed on from CRS team to the MBRS project team for sign-off. A risk was introduced for this to track the MBRS systems requirements till completion and it was decided to conditionally sign off on the NYHCRA requirements and then visit them again during the MBRS requirements review and approval phase. Tracking spreadsheet was maintained and weekly meetings were set up to mitigate any other unforeseen risk. As a proactive measure, meetings were also set up with the two upstream projects to make sure that they were on schedule.

Based on the learning from the requirement phase the design, development and testing milestones were tracked across the projects to make sure that the schedule was on track. Strategy for integration testing between the two systems was developed that helped in capturing many defects for the inter-dependent processes in the integration testing phase itself.


During the testing phase, a requirement change was discovered in one of the upstream systems that led to requirement changes in the membership and claim reporting systems as well. An issue was initiated and tracked by the CRS project team. The multiple project teams (MBRS, claim reporting and the two upstream systems, policy data entry and data migration system) worked hand-in-hand to maintain requirements. A strategy was prepared to involve the other teams in any requirement change such that the impact on the other team could be determined upfront. This was helpful in tracking requirement changes. Though it was challenging to successfully implement the project in this multiple project scenario, adopting the

discussed requirement management strategy made implementation possible with the slightest of hitches.

CONCLUSION

Multiple dependent projects need to understand the challenges in maintaining dependent and non-dependent requirements. The workflow proposed in this paper ensures timely scope demarcation, thereby avoiding missed functionalities and duplicity of effort in all the stages of the project lifecycle. Also, an early identification of impacted systems will lead to better architectural blue print adherence. This workflow can help the multiple project teams in maintaining the inter-dependent requirements by utilizing some of the techniques explained here. Identifying common requirements also reduces testing effort required as the number of test cases can be reduced across the projects.

REFERENCES

1. International Institute of Business Analysis. Available at <http://www.theiiba.org/>
2. http://www.irm.com.au/business_analysis_papers.htm
3. The Executive Guide to Business Analyst and Project Management Terminology. Available on www.whitepapers.techrepublic.com
4. Neville Turbit, Business Analysis using "Method H"™. Available on www.projectperfect.com.
5. Tony de Bree, What is Agile Analysis? November 2007. Available on <http://www.modernanalyst.com/tabid/115/articleType/ArticleView/articleId/197/What-is-Agile-Analysis.aspx>
6. Pierre M A Buuron, How to achieve Business Excellence through IT, December 2002. Available at <http://www.nyenrode.nl/download/lectures/buuron.pdf>
7. Jeanne Ross, Enterprise Architecture: Driving Business Benefits from IT, April 2006. Available on www.papers.ssrn.com. 

Effective Business Analysts Build Cathedrals

An effective business analyst understands the rationale behind why she does what she does, asserts the consulting editor Huai-Ling Ch'ng

Once upon a time, there was a traveler who came across three stone cutters. He approached each of them and asked: "What are you doing?" The first quickly responded, "I am a stonecutter and I am cutting stones." The second warmly responded, "I am a stonecutter and I am earning a living for my beloved family." The third joyously responded, "I am a stonecutter and I am building a cathedral."

Business analysts elicit business requirements and translate these to system specifications. Good business analysts seek to analyse and document the business requirements from a user or business context. Truly effective business analysts are able to pinpoint the rationale behind the business requirements, articulate these clearly to the customer, and conceptualise and accurately paint the end state solution in line with the goals of the business.

It is those business analysts who align themselves with the third stonecutter, who are able to identify their purpose and realise a larger vision. They understand the rationale for why they are doing what they are doing. They understand the bigger picture.

The title of this issue, 'Effective Business Analysis', can perhaps be best summed up by this stonecutter fable. Not only will the effective business analyst infuse herself with the same

enthusiasm and overall purpose like the way the third stonecutter did, but will also be able to create a sense of excitement, sell the bigger picture and lead the team along the journey.

As we bring this issue to a close, let us now look at some of the opportunities and challenges in the business analysis profession today and how the effective business analyst might respond to these.

BUSINESS ARCHITECTURE

There is a lot of talk today about business architecture. Business Architecture is an oft used term but is not clearly understood across most organisations. Business architecture covers - how the business is structured or organised, what its goals and objectives are and what it does to achieve these goals (i.e., its business processes).

Effective business analysts are able to work together with enterprise architects to help build the business architecture and ensure end-

to-end traceability of project requirements back to business context and goals, whilst doing this in not too rigid a way that causes them to be inflexible to business changes.

BUSINESS PROCESS MANAGEMENT

Business is not static. Companies need to manage their changes, understand when and how business processes change, or are improved upon. Business Process Management helps manage the dynamism of this business change, by setting up a value for management and governance structure against the documented business processes along its journey to achieve business goals.

Effective business analysts understand the value chain. At the same time, being well versed on the different interaction points of business process management they work effectively with the different stakeholders involved on management and governance aspects.

BEYOND THE PROJECT LEVEL

Projects are rarely undertaken in isolation, but in the context of larger programs of work. Not only are there cross-project impacts but organisations should also reuse whatever they can wherever possible, to help improve productivity.

Effective business analysts operate beyond the project level and focus on the program or enterprise level. They help organizations stay competitive by leveraging off past learnings. They handle change management effectively and ensure traceability. Most importantly however, they understand the needs of the end customer and are able to balance out business priorities.

BUSINESS CASE DEVELOPMENT

Developing business cases and conducting feasibility assessment is increasingly becoming part of the role that business analysts perform.

Effective business analysts approach business case development with an entrepreneurial mindset, as if their own funds were at stake. They build a compelling case, but at the same time ensure value is added. By lending their expertise in tying in the strategic importance of their work, supported by their business architecture, effective business analysts help deliver far more appropriate and relevant solutions to seize new business opportunities or solve existing business problems.

INDUSTRY ACTIVITY

The business analyst profession is also undergoing a change. There is increased awareness of the profession as a result of efforts from industry bodies in setting up industry standards, training and certification programmes and developing a body of knowledge for the profession. Effective business analysts ideally are not only certified, but are also passionate and will work towards increasing the profile of the business analyst profession.

CONCLUSION

There are indeed challenges ahead as the marketplace moves and changes as it does. But as long as business analysts understand the rationale for why they are doing what they are doing, they will continue to build many cathedrals in the years to come.

Author Profile

Huai-Ling Ch'ng is a senior business solutions architect with Infosys Australia, with over 13 years of experience working in requirements engineering and business consulting roles across major enterprises. He is presently working on setting up a BA Center of Excellence in Infosys, globally. Huai has an MBA and is also both CBAP® and PMP certified. He can be contacted at huailing_chng@infosys.com.

Index

- Approach 18, 37, 41, 45, 54, 69
 - Integrated 37, 54
 - Requirements Engineering 18
 - Structured 37, 41, 45, 69
- Balanced Scorecard 12, 56-58
- Claim Reporting System, also CRS 75-77
- CMMI 42
- Diagram 12, 30-34, 48-49, 51-52, 64-65, 67-69
 - Activity 48, 64-65, 67-69
 - Business Context 12
 - Class 34
 - Collaboration 51-52
 - Context 31-32, 34
 - Data Flow, also DFD 34
 - Jackson Context/Problem 30-32
 - Objective Linking 48-49, 52
 - Problem 30, 32-34
 - Role Activity, also RAD 30, 34
 - Structured Activity 64
 - Use-Case Activity, also UCAD 64
- EFT 19
- Framework 3-6, 8-10, 13, 20, 29-31, 33-35, 56
 - Classic Enterprise Architecture 4
 - Composite EBA 3, 8, 10, 13
 - COTS Package 20
 - Enterprise Architecture 4, 6, 13
 - Federal Enterprise Architecture, also FEAF 4, 6
 - Japanese Government Enterprise Software Architecture 4
 - The Open Group Architectural, also TOGAF 3, 4, 6, 13
 - S2S 29-31, 33-35
 - Value Measurement 56
 - Zachman 3-6, 8-9, 13
- Gartner 4, 13
- General Semantics 21-22, 24, 26, 28, 79
- Health Maintenance Organization, also HMO 75, 77
- Management 29, 80, 42-45, 56, 59-60, 62, 69, 71-72, 75, 78, 80
 - Business Process 29, 80
 - Change 42-44, 69, 75, 80
 - Issue 75
 - Project 45, 75, 78
 - Requirements 29, 43, 45, 56, 59-60, 62, 71-72, 75
- Membership System, also MBRS 75-77
- Methodology 4, 6-7, 52
 - Avancier 4, 7
 - Enterprise Architecture, also EA 7
 - OLAR 52
 - TOGAF's Architectural 6
- New York Healthcare Reform Act, also NYHCRA1997 77
- Object Oriented Analysis and Design, also OOAD 63, 69
- Object Oriented Programming and Systems, also OOPS 52
- Objective Linked Approach for Requirements Development, also OLAR 47-49, 51-53
- Proof of Concept, also POC 17-18, 20
- Rational Rose 52, 54, 68
- Requirements 16-18, 21-23, 28, 30-31, 33, 35, 37-45, 48, 55-56, 58-59, 61, 63-64, 66, 68-69, 72-73, 75, 77, 79-80
 - Analysis 30, 33, 35, 38, 45, 63-64, 66, 68-69
 - Documentation 68
 - Elicitation 17-18, 21-23, 28, 37, 41-44, 55-56, 58-59, 61, 66, 72, 75, 79
 - Engineering 16, 18, 21, 31, 35, 37-45,

58-59, 73, 80

Gathering 48, 63, 72

Management, see management
requirements

Specification 77

Value Traceability 59

Storyboard 15, 18-20, 43

Strategy, Context and Process,
also SCP 29, 31, 35

SDLC 34-35, 38, 43-44, 63

UML 52, 54, 63, 65, 68-69

Visio 52



SETLabs Briefings

BUSINESS INNOVATION through TECHNOLOGY

Editor
Praveen B Malla PhD

Consulting Editor
Huai-Ling Ch'ng

Copy Editor
Sudarshana Dhar

Graphics Editor
Ravishankar SL

Web Editor
Ramesh Ramachandran

ITLS Program
Ajay Kolhatkar PhD

Marketing Manager
Vijayaraghavan T S

Production Manager
Sudarshan Kumar V S

Distribution Manager
Suresh Kumar V H

How to Reach Us:

Email:
SETLabsBriefings@infosys.com

Phone:
+91-080-41173871

Fax:
+91-080-28520740

Post:
SETLabs Briefings,
B-19, Infosys Technologies Ltd.
Electronics City, Hosur Road,
Bangalore 560100, India

Subscription:
vijaytsr@infosys.com

**Rights, Permission, Licensing
and Reprints:**
praveen_malla@infosys.com

Editorial Office: SETLabs Briefings, B-19, Infosys Technologies Ltd.
Electronics City, Hosur Road, Bangalore 560100, India
Email: SetlabsBriefings@infosys.com <http://www.infosys.com/setlabs-briefings>

SETLabs Briefings is a journal published by Infosys' Software Engineering & Technology Labs (SETLabs) with the objective of offering fresh perspectives on boardroom business technology. The publication aims at becoming the most sought after source for thought leading, strategic and experiential insights on business technology management.

SETLabs is an important part of Infosys' commitment to leadership in innovation using technology. SETLabs anticipates and assesses the evolution of technology and its impact on businesses and enables Infosys to constantly synthesize what it learns and catalyze technology enabled business transformation and thus assume leadership in providing best of breed solutions to clients across the globe. This is achieved through research supported by state-of-the-art labs and collaboration with industry leaders.

Infosys Technologies Ltd (NASDAQ: INFY) defines, designs and delivers IT-enabled business solutions that help Global 2000 companies win in a flat world. These solutions focus on providing strategic differentiation and operational superiority to clients. Infosys creates these solutions for its clients by leveraging its domain and business expertise along with a complete range of services. With Infosys, clients are assured of a transparent business partner, world-class processes, speed of execution and the power to stretch their IT budget by leveraging the Global Delivery Model that Infosys pioneered. To find out how Infosys can help businesses achieve competitive advantage, visit www.infosys.com or send an email to infosys@infosys.com

© 2008, Infosys Technologies Limited

Infosys acknowledges the proprietary rights of the trademarks and product names of the other companies mentioned in this issue. The information provided in this document is intended for the sole use of the recipient and for educational purposes only. Infosys makes no express or implied warranties relating to the information contained herein or to any derived results obtained by the recipient from the use of the information in this document. Infosys further does not guarantee the sequence, timeliness, accuracy or completeness of the information and will not be liable in any way to the recipient for any delays, inaccuracies, errors in, or omissions of, any of the information or in the transmission thereof, or for any damages arising there from. Opinions and forecasts constitute our judgment at the time of release and are subject to change without notice. This document does not contain information provided to us in confidence by our clients.

NOTES

Authors featured in this issue

ANJANEYULU PASALA

Anjaneyulu Pasala PhD is a Senior Research Associate in Test Automation Lab at SETLabs, Infosys. He has research interests in component-based software engineering, software testing, UML testing profilers and web-services architectures. He can be reached at Anjaneyulu_Pasala@infosys.com.

ANSHUMAN PRAMANICK

Anshuman Pramanick is a Business Solution Architect with Infosys Australia. He has expertise in Banking and Finance domain business architecture and requirements management. He can be reached at Anshuman_Pramanick@infosys.com.

ASHISH CHANDRA

Ashish Chandra was a Process Consultant in Enterprise Quality Solutions practice of Infosys. He has experience in the area of CRM, process definitions and process deployment.

BADHRI NARAYANAN C.S.

Badhri Narayanan C.S. is a Senior Programmer Analyst with Manufacturing unit of Infosys. He has several years of experience in eliciting and analyzing business and system requirements. He can be contacted at Badhri_Sabarinathan@infosys.com.

ESWAR GANESAN

Eswar Ganesan is a Technical Specialist in SETLabs, Infosys. Eswar focuses on research in the area of Business Architecture. His other interests include Business Process Modeling, Process Analysis and Industrial B2B research. He can be reached at eswar_ganesan@infosys.com.

LUKE HOUSEGO

Luke Housego is a Senior Business Consultant at Infosys Australia. He has over 13 years of experience consulting in technology and business architecture. Luke can be contacted at luke_housego@infosys.com

MANIDEEPA CHATTERJEE

Manideepa Chatterjee is a Programmer Analyst with Insurance, Healthcare and Life Sciences unit of Infosys. She has around 5 years of experience of working on various critical IT projects. She can be contacted at manideepa_chatterjee@infosys.com.

MANU GOEL

Manu Goel is a Senior Project Manager with Banking and Capital markets unit of Infosys. He has several years of experience in executing IT projects and has been involved in important business analysis and consulting engagements. He can be contacted at manu_goel@infosys.com.

NANDINI TOLEY

Nandini Toley is a Business Analyst with Insurance, Healthcare and Life Sciences unit of Infosys. Her primary area of expertise is in the structural planning and analysis of Medicare products and services. She can be contacted at nandini_toley@infosys.com.

RAMESH PATURI

Ramesh Paturi is a Senior Technical Architect in SETLabs, Infosys. His interest includes Business-IT alignment covering Business Architecture, Business Process Modeling and Requirements Analysis. Ramesh can be reached at ramesh_paturi01@infosys.com.

RAVI GORTHI

Ravi Gorthi PhD is a Principal Researcher and Head of Test Automation and Knowledge Engineering Labs at SETLabs, Infosys. He specializes in the areas of software testing, business rules engines, intelligent agents and knowledge-driven IT solutions. He can be reached at Ravi_Gorthi@infosys.com.

RAVISHANKAR N

Ravishankar is a Principal Consultant in Enterprise Quality Solutions practice of Infosys. He has over 15 years of IT experience in areas like software development, project management and software quality improvement. He can be reached at RavishankarN@infosys.com.

SANDEEP ANAND

Sandeep Anand is a Programmer Analyst with Banking and Capital markets unit of Infosys. He has experience in implementing big funded projects at financial services sector clients. He can be reached at sandeep_anand@infosys.com.

TEJAL PRAKASH NEMANE

Tejal Prakash Nemane is a Programmer Analyst with Insurance, Healthcare and Life Services unit of Infosys. She has over 6 years of experience in IT and is currently working as a Business Analyst with a leading Healthcare insurance client in USA. She can be reached at tejal_nemane@infosys.com.

TUSHAR SHARMA

Tushar Sharma is a Senior Consultant in Enterprise Quality Solutions practice of Infosys. He has experience in business process management, IT delivery process enhancement, quality assurance, project management and process consulting. He can be contacted at Tushar_sharma01@infosys.com.

Subu Goparaju
Vice President
and Head of SETLabs

"At SETLabs, we constantly look for opportunities to leverage technology while creating and implementing innovative business solutions for our clients. As part of this quest, we develop engineering methodologies that help Infosys implement these solutions right first time and every time."

For information on obtaining additional copies, reprinting or translating articles, and all other correspondence, please contact:

Telephone : 91-80-41173871

Email: SetlabsBriefings@infosys.com

© SETLabs 2008, Infosys Technologies Limited.

Infosys acknowledges the proprietary rights of the trademarks and product names of the other companies mentioned in this issue of SETLabs Briefings. The information provided in this document is intended for the sole use of the recipient and for educational purposes only. Infosys makes no express or implied warranties relating to the information contained in this document or to any derived results obtained by the recipient from the use of the information in the document. Infosys further does not guarantee the sequence, timeliness, accuracy or completeness of the information and will not be liable in any way to the recipient for any delays, inaccuracies, errors in, or omissions of, any of the information or in the transmission thereof, or for any damages arising there from. Opinions and forecasts constitute our judgment at the time of release and are subject to change without notice. This document does not contain information provided to us in confidence by our clients.

Infosys[®]

POWERED BY INTELLECT
DRIVEN BY VALUES