Editor: Suzanne Robertson ■ The Atlantic Systems Guild ■ suzanne@systemsguild.com

Managing Requirements for Business Value

John Favaro

Requirements engineers are often asked the question, "What will the investment in requirements give to our business?" John Favaro uses his experience in financial engineering to help us answer this question by providing some tools for quantifying the value of requirements.
—Suzanne Robertson

"Managing requirements for business value? What's there to manage? Aren't they just…there?" Such perplexity was perfectly understandable in the good old days, when all you had to do was get the requirements from your customer, design your system, and produce one of those awful conformance matrices that demonstrated that you had implemented each and every one of the requirements in your system. Against this dreary backdrop of conformance matrices and the like, it is indeed hard to imagine a role for requirements in the development and execution of business strategy. But, as the song says, the times, they are a changin'. We are now looking outside the traditional boundaries and developing ways of getting maximum business value from our investment in requirements. This column discusses three ways of realizing this value: managed requirements process, requirements agility, and contractual innovation.

Toward requirements reuse

The first important change in recent years has been the emergence of a true, actively managed requirements process, which replaces the passive approach of the past where requirements simply arrived, if you were lucky, on a one-way street from the customer. An analyst who masters the requirements process can become an active participant in the strategic conception of a system or product. He or she can elicit and formulate requirements in such a way that the path from requirement to implemented system feature is illuminated in all of its consequences, both technical and economic. Of course, we already know that a robust requirements process is a key factor in resolving problems early in the life cycle, with all the familiar economic benefits. But an actively managed requirements process gives you much more than that. Requirements analysts and customers alike can discover a new flexibility. Although there are always a few nonnegotiable requirements (such as no loss of human life), the vast majority are suitable for examination, reformulation, negotiation, or adaptation. As unexpected consequences are uncovered (for example, the projected high cost of implementation), the analyst can cooperate with the customer to search for other, equally satisfactory requirements formulations. The economic tools of cost–benefit analysis for this process have been well understood for years.

A full cost–benefit analysis of a requirement (or group of requirements) needs an investment in time and resources. Furthermore, assessing the cost–benefit of requirements is more difficult than design or implementation, because requirements are the furthest upstream in the development process. Consequently there are more unknown factors; it can take a full development cycle before the complete economic impact of a requirement is known.

Now perhaps you can understand why requirements researchers are studying techniques for creating reusable requirements.[1] The first time you conceive a system based on a requirement, estimating the costs and benefits might be difficult. But after carrying through the system to implementation, you will have a much better idea of the costs and benefits triggered by that requirement. A well-formulated, measurable, reusable requirement—including a full cost–benefit analysis as part of its description—is every bit as valuable as a reusable software module.

Agile requirements

The second important change has been the emergence of strategies for confronting the *bête noire* of the requirements process: vague and changing requirements. These strategies are best reflected in a new generation of software life cycles known as agile processes (see www.agilealliance.org). Some prominent examples of agile processes include Alistair Cockburn's Crystal Methods, Bob Charette's Lean Development, and Jim Highsmith's Adaptive Software Development. The common denominator of these agile processes is the iterative development paradigm, which breaks up the tradition of upfront requirements elicitation and analysis. No longer do you fire and forget requirements and then move on to the next phase. Requirements may be introduced, modified, or removed in successive iterations. As Kent Beck (chief evangelist of Extreme Programming, the most visible of the agile processes) exhorts us, requirements management should "embrace change."[2] Business conditions change, giving rise to new requirements; requirements thought to be critical turn out not to be as the customer sees the first versions of the system. Requirements that once were vague become crystal clear as uncertainty is resolved; a requirement once thought to be rigid could be negotiated and reformulated to permit several alternative features that could satisfy it.

Such changing conditions provide opportunities for the strategist to increase the value of his process. However, the traditional tools of cost–benefit analysis that apply so well to the noniterative requirements process have proven less adequate to help the requirements analyst examine the economic value of his newfound strategic flexibility—and this brings me to the third important change in recent years. I'd like to draw now on my interaction with Kent Beck over the past few years to discuss some cutting edge ideas about the relationship between strategy and finance and their affect on requirements management.

Contractual innovation

Evaluating the financial impact of strategic decisions has been the subject of great debate since the dawn of economics as a discipline. In a list of the Top 10 Unsolved Problems in Corporate Finance first compiled by the legendary financial authors Richard Brealey and Stewart Myers in 1981 and unchanged in last year's sixth edition of their classic textbook *Principles of Corporate Finance*, the financial impact was ranked Number 1.[3]

Evaluating the financial impact of strategic decisions has been the subject of great debate since the dawn of economics as a discipline.

In recent years, people have placed hope in a new branch of financial theory known as contingent claims analysis—or more popularly, real options—made possible by the breakthroughs in the 1970s in option pricing theory. In this approach, the opportunities created by strategic flexibility are evaluated with the financial tools of option pricing theory.

Let's take an example from XP. Suppose the customer requires your application to provide access to an enterprise-wide knowledge management system that he or she is contemplating introducing in a few months. A simple cost–benefit analysis on the system features that would satisfy this requirement is positive, say, $10 implementation cost versus $15 in benefits. But an enormously uncertain environment undermines the cost–benefit analysis. The customer admits that the uncertainty ("volatility") of his estimate is as much as 100 percent. If the knowledge management system is never introduced, then it will have been a waste of time to provide the access capability; however, if it is introduced, the access capability could become far more valuable than originally envisioned. The customer says that the uncertainty will be resolved in a year. The XP process permits the strategy of waiting until a future iteration to take up the requirement. Is there any way to calculate the economics of this alternative strategy? The tools of option pricing theory can in fact calculate the value of waiting to be slightly less than $8—more than the $5 of benefit accrued by immediate development.[4]

The option to delay implementing a requirement is an example of the way that contingent claims analysis is making a profound impact on the requirements process in the form of *contractual innovation*, a result of the new discipline of financial engineering born with the advent of option pricing theory. Kent likes to say that unlike fixed-scope traditional contracts, XP contracts have optional scope: every iteration provides a formal decision point in which the

FILL?

customer can change direction, abandoning requirements, introducing new requirements, or selecting between alternative requirements.

For example, you sign such contracts not for a fixed set of functionality, but for a team's best effort for a fixed period at a fixed price. The precise scope to be implemented will be negotiated periodically over the life of the contract, much as professional services contracts are run today. Changes in the team's actual velocity and the relative estimates attached to each feature are factored into these scope negotiations, as are changes in the perceived relative value of these features.

Option pricing theory yields some surprising insights to the economic value of such contracts. For instance, there is an exotic type of option known as a best-of or rainbow option. The owner possesses two options, of which only one can be exercised. The rainbow option has the most value when the alternatives are negatively correlated—that is, when the same conditions that increase one alternative's value will decrease the other's. This implies that a contract's value is enhanced by contradictory requirements. For example, two requirements each specifying the application to run on a different platform is a rainbow option. If the choice can be delayed to some later time, it adds value for the customer, letting him hedge the ultimate platform choice in the contract.

Similarly, a requirement whose utility is uncertain (estimated cost and value are close) gains value by inclusion in an optional scope clause, because the option to delay implementation has demonstrable economic value. Where contractual flexibility exists to select among alternative requirements, add requirements, or even to abandon requirements, economic value is added.

Adding value

What does all this mean for you as a requirements analyst? As the requirements process evolves to embrace increased strategic flexibility, and the new financial tools of contingent claims analysis mature, requirements become an important currency in system characteristic negotiation. By learning to create reusable requirements with a companion cost–benefit analysis, you bring valuable material to the table from the very beginning. By studying the new generation of agile development processes, you become fluent in the strategic possibilities to add value to the requirements process over the entire product life cycle. By learning something about the new tools of financial analysis introduced in this column, you can better understand how strategic flexibility in the requirements process adds value.

For that is what the requirements process should be about. If you remember nothing else from this column, remember this—stamp it on your forehead if necessary: The purpose of the requirements process should not be to "cover all eventualities," or to "limit the damage," or to "minimize risk," or even to "satisfy the customer." The purpose of the requirements process is to add business value. It is a subtle shift in perspective for the requirements analyst, but it makes all the difference because it puts you in the position of managing requirements to make the most of your strategic opportunities.

References

1. S. Robertson and J. Robertson, "Chapter 12: Reusing Requirements," *Mastering the Requirements Process*, Addison-Wesley, Reading, Mass., 1999, pp. 218–234.
2. K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, Reading, Mass., 1999.
3. R. Brealey and S.C. Myers, *Principles of Corporate Finance*, McGraw-Hill, New York, 2000.
4. M. Amram and N. Kulatilaka, *Real Options: Managing Strategic Investment in an Uncertain World*, Harvard Business School Press, Cambridge, Mass., 1999.

John Favaro is an independent consultant based in Pisa, Italy. He is European co-chair of the IEEE Technical Subcommittee on Software Reuse and a founding member of the steering committee of the Society for Software Education. He has degrees in computer science from Berkeley and Yale. Contact him at Via Gamerra 21, 56123 Pisa, Italy; jfavaro@tin.it.