Column

Managing Agile Results at Several Levels of Organization

by Kai and Tom Gilb

In our previous Column (**The Top 10 Critical Requirements are the Most Agile Way to Run Agile Projects, August 2012**, Ref 3) we emphasized the necessity of controlling delivery of value to stakeholders by using multidimensional top level quantified project objectives.

In this column we are going to take you a step further. We are going to present the idea of multiple levels of control of the agile project.

It is not enough that an agile project simply deliver the IT system's top level qualities like, usability, security and adaptability. Even though even *that* direct delivery and management of quality levels is missing from *most* agile methods in practice. This is what we discussed in the previous column.

If the project is large and complicated, meaning the project is delivering results to many and varied stakeholders, like a hospital system. You will need to manage the delivery of value to each type of stakeholder. Usability for nurses is not the same as for a surgeon or administrator.

In addition, if the project is expected to have results for the large organization (Hospital level for example), then a third level of management is required, to relate the stakeholder results to the larger organization.

Kai Gilb practiced this successfully on a project for the 'Bring' organization in Norway (Ref. 2). This method turned a failed Scrum project (sales went drastically down when the new web portal was delivered) into a successful project.

The process Kai used can be viewed as one that stands above, and controls the Scrum process. We call it a *Value Management* process.

The stakeholder vision level specifies the most critical values required by each stakeholder. The prioritization step determines which stakeholders will get 'how much' of their value, delivered in the next delivery cycle (Sprint). This prioritization is based on several factors such as deadlines, levels delivered to date, minimum tolerable levels and other factors. (*See depth paper Managing Priorities ref 4*)

The detailed 'modeling' of the value flow is done using Impact Estimation tables (ref. 1).

1. on the left side, critical objectives (like **Profit**) are named with a tag, which cross references a more-detailed specification (with Scale of Measure and Numeric Goal level)
2. On the top side a series of strategy columns (like **Training Costs**) are referenced (more detail on *them* is specified elsewhere under *that* tag)

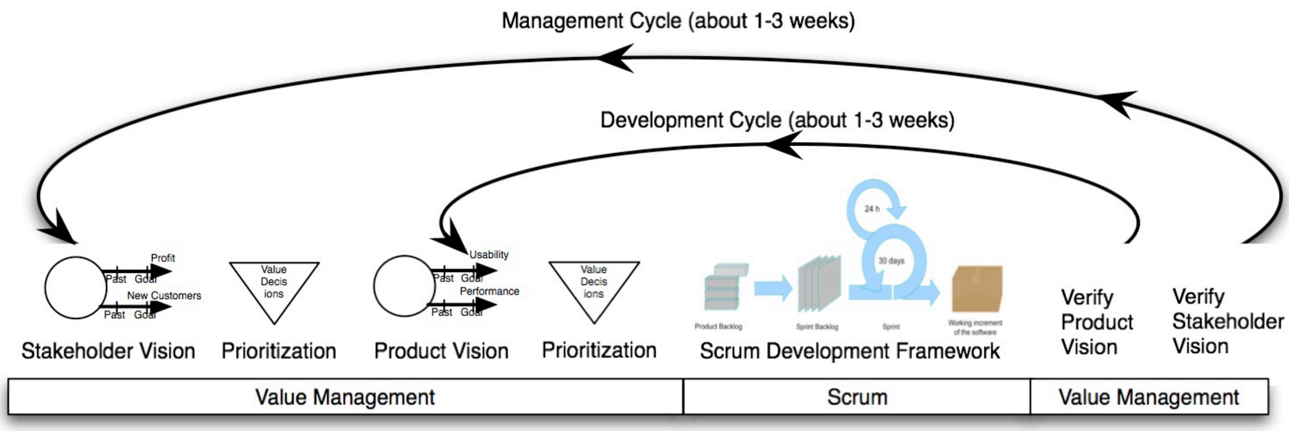
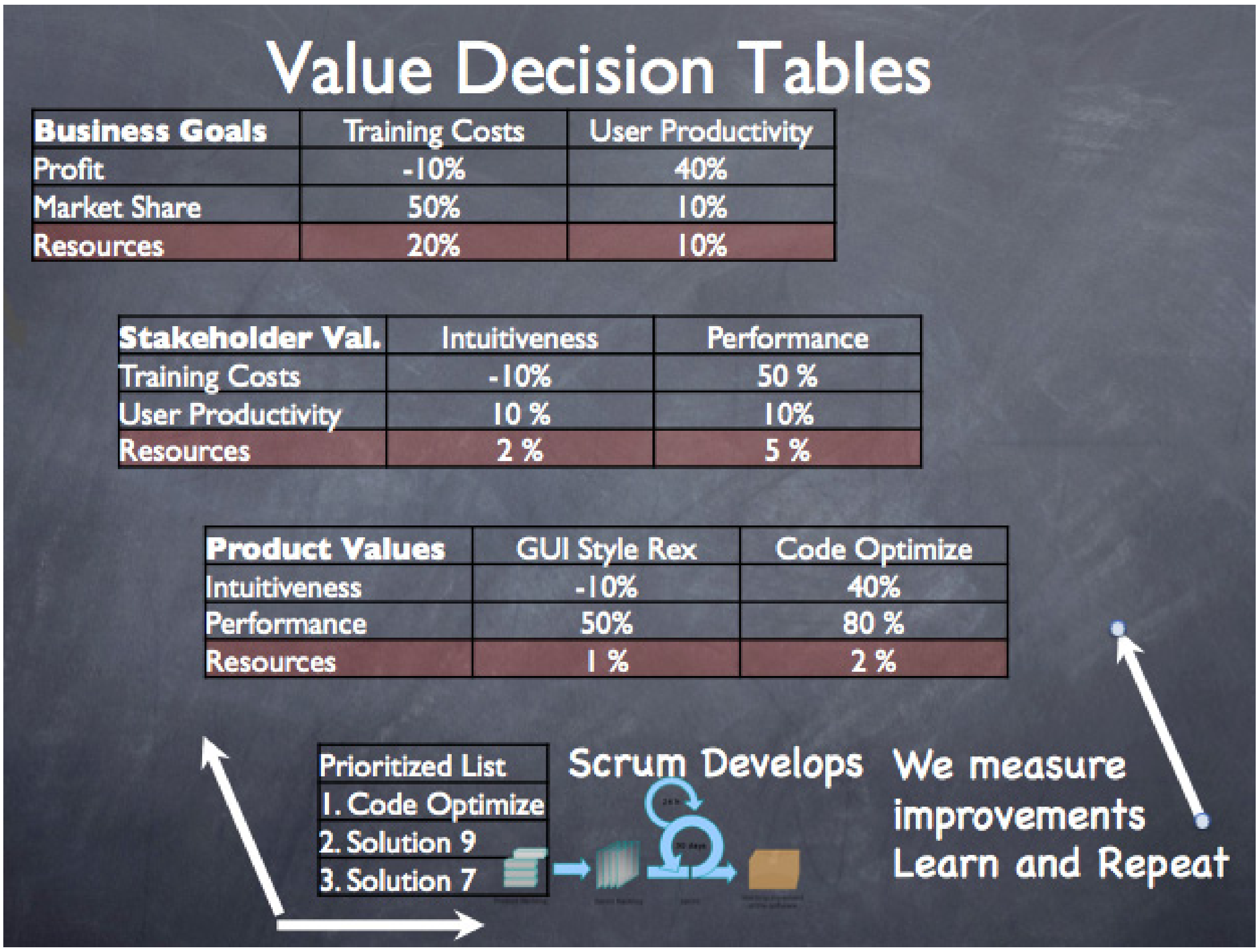


Figure 1: Two levels of result management, above a Scrum process. The Business level is missing here. We are just managing delivery of results to stakeholders.



Value Decision Tables

Business Goals	Training Costs	User Productivity
Profit	-10%	40%
Market Share	50%	10%
Resources	20%	10%

Stakeholder Val.	Intuitiveness	Performance
Training Costs	-10%	50 %
User Productivity	10 %	10%
Resources	2 %	5 %

Product Values	GUI Style Rex	Code Optimize
Intuitiveness	-10%	40%
Performance	50%	80 %
Resources	1 %	2 %

Prioritized List
1. Code Optimize
2. Solution 9
3. Solution 7

Scrum Develops

We measure improvements Learn and Repeat

Figure 2: The use of Impact Estimation Tables (Ref. 1) as 'Value Decision' Tables.
This is a very simplified view of the real model used at Bring. It includes the Business level.

3. The estimated impact, on reaching to Goal levels, due to the strategies is primarily given as a percentage of the 'distance to the Goal level'. 0% means no impact, 100% means all the way to the Goal level, for stated conditions (such as when, where, for which stakeholder)
4. The resources % is a % of a budget, or project time scope.

You will see that the higher level strategies are reused as objectives in the level below. In this way there is a direct numeric and dynamically track-able (through project delivery cycles) relationship between the top levels and the level of design for the IT system (the prioritized list, the backlog). This allows management to see the connection between IT design and architecture, and its impact on the rest of the organization; at a stakeholder level and at the general organization level.

Here is a sample explanation of how it works, based on the tables above.

1. Code Optimization (a design strategy, to be implemented by developers in Scrum) contributes an estimated 80% to Performance Goal level.
2. The Performance attribute (of the IT system) contributes (when at its Goal level) an estimated 50% towards the reduction of Training Costs.
3. Training Costs (when at its Goal level) contributes an estimated 50% towards reaching our Market Share Goal level.

The 'developers' are not concerned with this process above the coding. Unfortunately they don't even consider, and plan for, the larger system (data, machines, people). They are narrowly concerned with source code, frameworks, testing and bugs. That is why projects fail, if left to traditional developers alone, and to Scrum in isolation. There is no manageable connection to the real world.

You cannot expect a developer to develop this management results framework. At best it is the responsibility of the Product Owner function to manage it. But if we expect real value-to-stakeholder to flow, *and that is the main point of Agile*, then someone (management) must articulate these values, their relation, and track them incrementally, until all Goals are met ('Done').

Smaller projects do not need this management framework. But since Agile is being used for non trivial projects, there is a point where we need to introduce relevant value management. The small additional effort (5%?, a few days) is a small price to pay for ensuring real success.

> about the authors

Tom Gilb and Kai Gilb

Tom Gilb and Kai Gilb have, together with many professional friends and clients, personally developed the methods they teach. The methods have been developed over decades of practice all over the world in both small companies and projects, as well as in the largest companies and projects.

Tom Gilb

Tom is the author of nine books, and hundreds of papers on these and related subjects. His latest book 'Competitive Engineering' is a substantial definition of requirements ideas. His ideas on requirements are the acknowledged basis for CMMI level 4 (quantification, as initially developed at IBM from 1980). Tom has guest lectured at universities all over UK, Europe, China, India, USA, Korea – and has been a keynote speaker at dozens of technical conferences internationally.

Kai Gilb

has partnered with Tom in developing these ideas, holding courses and practicing them with clients since 1992. He coaches managers and product owners, writes papers, develops the courses, and is writing his own book, 'Evo – Evolutionary Project Management & Product Development.'

Tom & Kai work well as a team, they approach the art of teaching the common methods somewhat differently. Consequently the students benefit from two different styles.

There are very many organizations and individuals who use some or all of their methods. IBM and HP were two early corporate adopters. Recently over 6,000 (and growing) engineers at Intel have adopted the Planguage requirements methods. Ericsson, Nokia and lately Symbian and A Major Mulitnational Finance Group use parts of their methods extensively. Many smaller companies also use the methods.