Earned Business Value Management

– See that You Deliver Value to Your Customer –

Jo Erskine Hannay, Simula Research Laboratory,
Hans Christian Benestad, ExpertWare AS, **Kjetil Strand**, PROMIS AS

The order in which you send your backlog items into construction determines when stakeholders will be able to reap benefit from what functionality. This can have substantial impact on market timing, enterprise earnings and project manager survivor rate. There are several ways to order a backlog, and sophisticated methods and tools exist to do so—for example, in release planning. But the important point we'll make here, is that no matter what scheme for backlog ordering you choose to use, you ought to be explicit on the order in which you realize potential business value. To this end, we'll present methods to express business value relative to cost in your backlog and methods to monitor how much potential business value you're realizing along the way—in addition to cost expended. Given the central role proclaimed to business value in Agile, we said in [9] that you should assign *benefit points* to your project's product elements (epics and stories); with at least the same vigor and rigor with which you assign story points. To do this, assign points (for example, in a benefit poker session using the Fibonacci scale) according to how much you think an epic contributes to the project's distinct objectives (Fig. 1). Objectives, which are part of the business case for the project, express the effect in/on the organization that the project's deliverables are intended to induce. The objectives may, in turn, be assessed to contribute to the enterprise's planned returns to varying degrees. The fact that objectives may not represent equal value is then reflected by balancing the benefit points accordingly. We summarized all this in [9] into a core practice of *Benefit Point Estimation for Epics*.

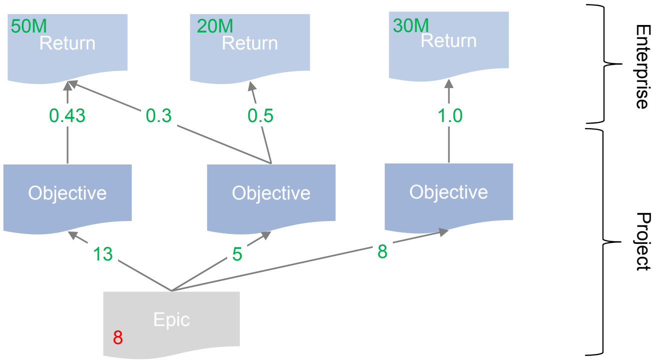


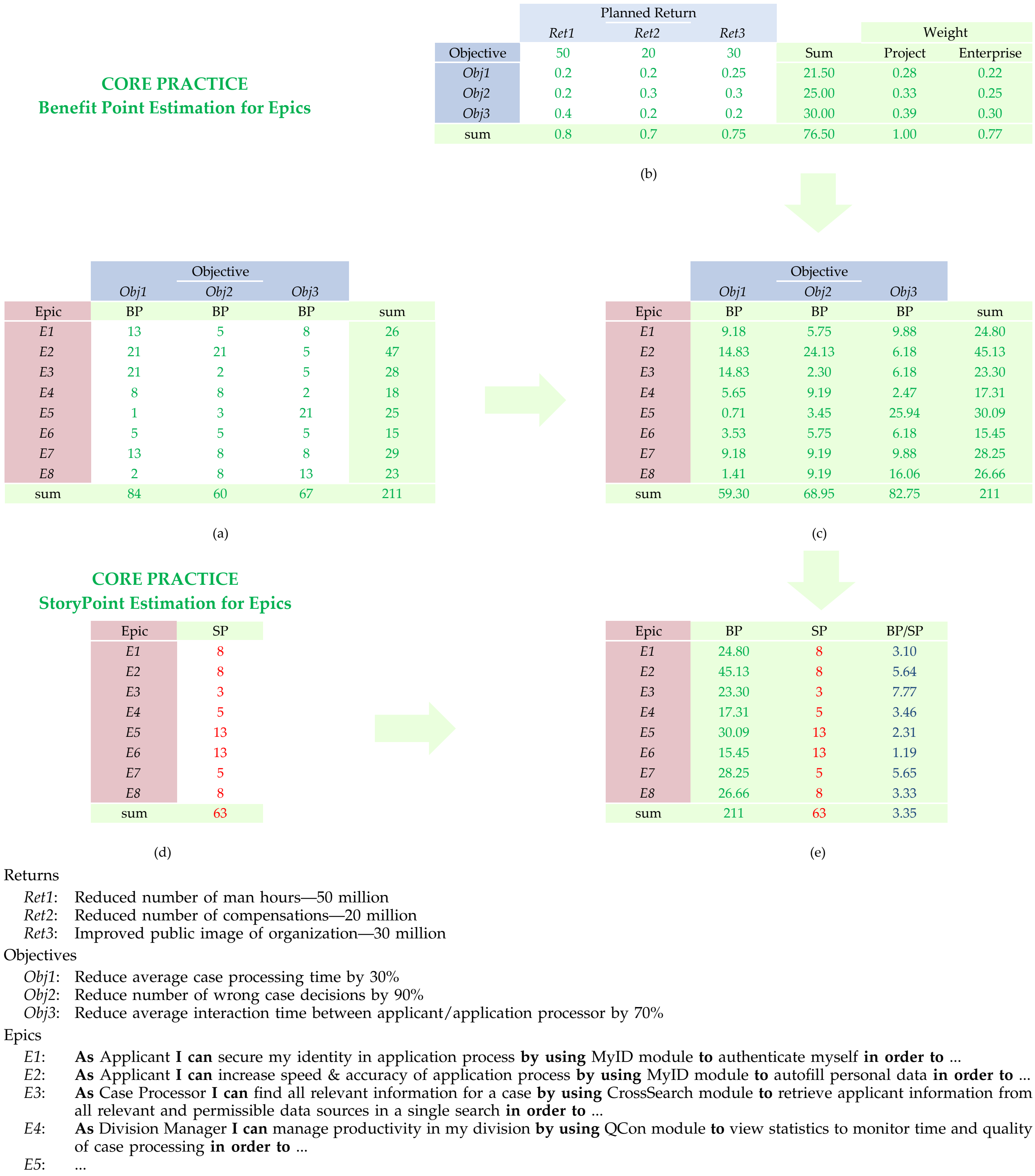
Fig. 1. Product element with both associated story points (8) and benefit points (13, 8, 5). Objectives contribute to various degrees to planned enterprise returns.

As an example from the public service domain, Fig. 2 shows Returns *Ret1*–*Ret3*, Objectives *Obj1*–*Obj3* and Epics *E1*–*E8*. Fig. 2(a) exemplifies the first estimation task to be done in the core practice— to provide business value estimates for epics in the form of benefit points. For example, using the Fibonacci sequence familiar from planning poker, epic *E1* has been estimated to contribute to Objectives *Obj1*, *Obj2*, *Obj3*, respectively, 13, 5 and 8 benefit points (BP); in all 26 benefit points. The total number of benefit points assigned in this manner is 211 in this example. Fig. 2(b) exemplifies the second estimation task to be done—to estimate how much each objective contributes to returns. The total strategically planned return in *Ret1*, *Ret2* and *Ret3* is 100 million. The project's objectives *Obj1*, *Obj2* and *Obj3* are estimated to contribute 21.5 million, 25 million and 30 million, respectively, to that return; in all 76.5 million. Thus, the project's objectives, once fulfilled, contribute unevenly toward the return of the project, and only partly to the enterprise's strategically planned return. Then, Fig. 2(c) shows the benefit points automatically balanced due to objectives having different value. Your tasks only involve providing estimates for the parts with white background in Figure 2. The green parts can be automatically generated by your tool (e.g., Excel).

This core practice effectively links the project's product estimates to the business case and to strategic plans. The methodology is to harness and systematize stakeholders' insights and project learning; rather than to employ sophisticated tools for calculating estimates that, by the way, fair no better on average than expert estimation [14]. So, methods must be simple, support expert's cognitive processes and give sufficient, rather than optimal, results [8]. Although simple, the core practice supports powerful monitoring techniques.

You can use benefit points in combination with story points to obtain means to monitor and adjust your project. Assigning story points (another core practice) is a common activity done routinely in projects, so we'll assume you know how to do this; e.g., in planning poker sessions. However, we'll make a few remarks in the context of benefit/cost management.

Benefit manifests itself after deployment, so to get a sensible benefit/cost measure, cost estimates should include post-deployment cost in addition to development cost. Traditionally, story points reflect development cost only. However, it's common to assume that lifecycle cost is proportional to, or linearly dependent on, development cost; e.g., [13], depending on domain and



CORE PRACTICE
Benefit Point Estimation for Epics

	Planned Return				Weight	
	Ret1	*Ret2*	*Ret3*			
Objective	50	20	30	Sum	Project	Enterprise
Obj1	0.2	0.2	0.25	21.50	0.28	0.22
Obj2	0.2	0.3	0.3	25.00	0.33	0.25
Obj3	0.4	0.2	0.2	30.00	0.39	0.30
sum	0.8	0.7	0.75	76.50	1.00	0.77

(b)

	Objective			
	Obj1	*Obj2*	*Obj3*	
Epic	BP	BP	BP	sum
E1	13	5	8	26
E2	21	21	5	47
E3	21	2	5	28
E4	8	8	2	18
E5	1	3	21	25
E6	5	5	5	15
E7	13	8	8	29
E8	2	8	13	23
sum	84	60	67	211

(a)

	Objective			
	Obj1	*Obj2*	*Obj3*	
Epic	BP	BP	BP	sum
E1	9.18	5.75	9.88	24.80
E2	14.83	24.13	6.18	45.13
E3	14.83	2.30	6.18	23.30
E4	5.65	9.19	2.47	17.31
E5	0.71	3.45	25.94	30.09
E6	3.53	5.75	6.18	15.45
E7	9.18	9.19	9.88	28.25
E8	1.41	9.19	16.06	26.66
sum	59.30	68.95	82.75	211

(c)

CORE PRACTICE
StoryPoint Estimation for Epics

Epic	SP
E1	8
E2	8
E3	3
E4	5
E5	13
E6	13
E7	5
E8	8
sum	63

(d)

Epic	BP	SP	BP/SP
E1	24.80	8	3.10
E2	45.13	8	5.64
E3	23.30	3	7.77
E4	17.31	5	3.46
E5	30.09	13	2.31
E6	15.45	13	1.19
E7	28.25	5	5.65
E8	26.66	8	3.33
sum	211	63	3.35

(e)

Returns

Ret1: Reduced number of man hours—50 million
Ret2: Reduced number of compensations—20 million
Ret3: Improved public image of organization—30 million

Objectives

Obj1: Reduce average case processing time by 30%
Obj2: Reduce number of wrong case decisions by 90%
Obj3: Reduce average interaction time between applicant/application processor by 70%

Epics

E1: **As** Applicant **I can** secure my identity in application process **by using** MyID module **to** authenticate myself **in order to** ...
E2: **As** Applicant **I can** increase speed & accuracy of application process **by using** MyID module **to** autofill personal data **in order to** ...
E3: **As** Case Processor **I can** find all relevant information for a case **by using** CrossSearch module **to** retrieve applicant information from all relevant and permissible data sources in a single search **in order to** ...
E4: **As** Division Manager **I can** manage productivity in my division **by using** QCon module **to** view statistics to monitor time and quality of case processing **in order to** ...
E5: ...

Fig. 2. Core Practices *Benefit and Story Point Estimation for Epics* and running example. (a) Epics' contribution to objectives, (b) Objectives' contribution to returns, (c) Resulting balanced benefit points, (d) Story points for life-cycle cost. (e) Benefit point to story point ratio. Expert estimation on white background. Tool calculation on green background.

type of system. Under that assumption, story points can be assigned as before, and it is simply the monetary value of a story point that determines whether it reflects life-cycle cost or development cost only. Our methods apply regardless of that assumption, but if you do make that assumption (and it's warranted), some of the methods can take on a simpler form. In any event, for our running example, we assume the story points (SP)

TABLE 1

Benefit/cost in terms of points.

Epic	BP	SP	BP/SP
E3	23.30	3	7.77
E7	28.25	5	5.65
E2	45.13	8	5.64
E4	17.31	5	3.46
E8	26.66	8	3.33
E1	24.80	8	3.10
E5	30.09	13	2.31
E6	15.45	13	1.19
sum	211	63	3.35

TABLE 2

Benefit/cost in money. 1BP=0.36 million, 1SP=0.6 million.

Epic	Benefit	Cost	Benefit/Cost
E3	8.45	1.80	4.69
E7	10.24	3.00	3.41
E2	16.36	4.80	3.41
E4	6.28	3.00	2.09
E8	9.67	4.80	2.01
E1	8.99	4.80	1.87
E5	10.91	7.80	1.40
E6	5.60	7.80	0.72
sum	76.50	37.80	2.02

given in Fig. 2(d).

You can then immediately calculate the BP/SP ratio in Fig. 2(e) to get a relative benefit/cost measure. You can divide story points by benefit points, because both types of points are on a so-called *ratio scale*. If you use *categorical* or *nominal* schemes, say, MoSCoW [12]—which gives you four categories of importance—Must, Should, Could, Won't -have—to assess benefit, you cannot divide your benefit estimates by cost. To get a benefit to cost measure out of MoSCoW, you could order product elements according to increasing cost within each category, and then order the backlog by selecting the ordered elements in Must, then Should, Could and Won't. However, it's entirely possible that an element in a less important category may have a higher actual benefit/cost ratio than a given element in a more important category, due to low cost. Without a sound measure of benefit/cost provided by ratio scales, you would not become aware of such incidents.

We'll now show how to use the core practice in further project management activities. The first activity we will look at is, simply, to order the epics backlog.

Activity: *Order Epic Backlog*

Given story points and benefit points for epics, you can start to make informed decisions on the order in your backlog. There are several possible tactics, and we'll go into some detail of one particular tactic and discuss others as we go along.

Table 1 is Fig. 2(e) sorted according to decreasing BP/SP ratio. Our position here is that your *main tactic* should be to plan to elaborate and develop epics in that order. This maximizes business value for customer, relative to cost, early. (We'll comment later on other factors relevant to order—e.g., risk, uncertainty, dependencies. For now, we focus on the overall main tactic.)

This is markedly different from maximizing business value at all cost and different from maximizing business value within a fixed cost or schedule bound. The relevant idiom here is that it may be OK to go over budgeted cost if your estimates show that you'll generate all the more benefit by doing so; i.e., that the benefit/cost ratio is sufficiently high. Just as relevant, it's OK to stop construction when the benefit/cost ratio falls below a certain level. The ultimate goal for the project is to fulfill the project objectives, thereby realizing business value, as long as this does not cost more than it gains.

The Iron Triangle of project management (Fig. 3 left), where Quality is thought to be the result of balancing the three factors Scope, Schedule and Cost, has been debated in the agile community—and perhaps abandoned for the Agile Triangle (Fig. 3 center), where business value, or benefit, is brought forward as a prime factor, in recognition of the "value for customer" mantra. Perhaps, because trying to balance elements inadvertently leads to one of the elements (e.g., Cost) becoming a goal in itself, rather than a means to achieve Quality, the Agile Triangle puts Quality as a distinct factor.

But although the intention behind these triangles is to strike a balance between factors, the intrinsic polarization makes it tempting to put one factor up against the others. We contend that benefit and cost should not be polarized; but to the contrary, integrated into a single metric. Therefore, the Benefit/Cost Triangle (Fig. 3 right) has Benefit/Cost as a factor. Quality refers to technical quality including architecture, and Schedule is the remaining constraint, since Scope is really a part of Benefit. We will here look at the tactic of maximizing Benefit/Cost (subject to Schedule and Quality).

A powerful aspect of using relative sizes such as benefit points and story points is that you can assign actual monetary values to your points, according to current knowledge. The initial values you set, prior to project learning, would be based on the business case for the project. As an example, Table 2 shows the same information as in Table 1, but where the monetary value (0.36 million) of a benefit point is set at the total estimated return of the project (76.5 million) divided by the total number of benefit points given to the project (211 BP), and the monetary value representing life-cycle cost of a story point is set at 0.6 million; say based on structured stakeholder meetings and past experience

TABLE 3
Detailing into stories for planned first release *E3*, *E7*, *E2*.

Epic	Story	BP	Benefit	Part of Epic	BP	Benefit	SP	Cost	Part of Epic	SP	Cost	Benefit/Cost	
E3		23.30	8.45				3.00	1.80				4.69	
	E3A			0.7	16.31	5.91			0.6	1.80	1.08		5.47
	E3B			0.3	6.99	2.53			0.4	1.20	0.72		3.52
E7		28.25	10.24				5.00	3.00				3.41	
	E7A			0.6	16.95	6.15			0.2	1.00	0.60		10.24
	E7B			0.3	8.48	3.07			0.3	1.50	0.90		3.41
	E7C			0.1	2.83	1.02			0.5	2.50	1.50		0.68
E2		45.13	16.36				8.00	4.80				3.41	
	E2A			0.5	22.57	8.18			0.2	1.60	0.96		8.52
	E2B			0.1	4.51	1.64			0.2	1.60	0.96		1.70
	E2C			0.2	9.03	3.27			0.3	2.40	1.44		2.27
	E2D			0.2	9.03	3.27			0.3	2.40	1.44		2.27
E4		17.31	6.28				5.00	3.00				2.09	
E8		26.66	9.67				8.00	4.80				2.01	
E1		24.80	8.99				8.00	4.80				1.87	
E5		30.09	10.91				13.00	7.80				1.40	
E6		15.45	5.60				13.00	7.80				0.72	
sum		211.00	76.50		96.68	35.05	63.00	37.80		16.00	9.60	2.02	3.65

TABLE 4
Detailing into stories for next epic *E4* in line.

E4		17.31	6.28				5.00	3.00				2.09	
	E4A			0.3	5.19	1.88			0.2	1.00	0.60		3.14
	E4B			0.2	3.46	1.26			0.3	1.50	0.90		1.39
	E4C			0.3	5.19	1.88			0.4	2.00	1.20		1.57
	E4D			0.2	3.46	1.26			0.1	0.50	0.30		4.18

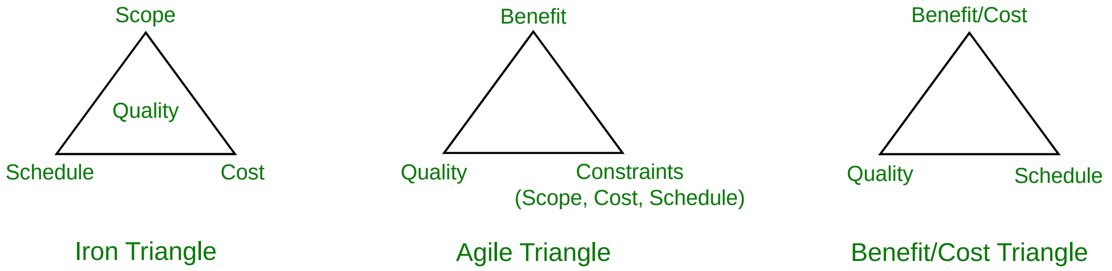
from earlier projects with similar characteristics. For example, structured discussions may have established the development cost of a story point to 0.3 million. Then a linear model of post-deployment cost might suggest that life-cycle cost is twice that of development cost. Thus, the life-cycle cost estimate is 37.8 million and the life-cycle benefit estimate is 76.5 million.

With monetary values, benefit and cost have the same denomination. With the values set as above, it is evident that, according to initial estimates, Epic *E6* has a benefit/cost ratio below 1; which means that this epic, as a whole, should not be put into construction, since it will return less benefit than it costs.

Activity: Order Story Backlog

Suppose now that the project deploys in releases, and that Epics *E3*, *E7*, *E2*, in harmony with our tactic, are wisely planned for the first release. In line with just-in-time detailing, this is the point at which those epics should be elaborated into stories.

Suppose Epics *E3*, *E7*, *E2* are elaborated into stories as indicated in the "story" column of Table 3. Now,



Iron Triangle

Agile Triangle

Benefit/Cost Triangle

Fig. 3. Iron Triangle, Agile Triangle and Benefit/Cost Triangle.

benefit points and story points must be assigned to stories somehow, since stories are what actually get put into construction via release backlogs.

Benefit points distributed on stories

Considerations of business value should be held at the level of epics, *not* at the level of stories. At the level of stories, you should only consider how much each story contributes to realizing its epic's estimated benefit; see Fig. 4. Why? Because stories specify functionality at a level of detail and granularity which usually makes it hard to relate to objectives in the business case [4]. Secondly, it's important to keep your expert estimation as local and simple as possible. Therefore, you should consider only one level of the relationship at a time: For epics, look at their relationship to objectives. For stories, look at their relationship to epics.

There are various ways in which you can estimate user stories' proportions of epics contribution to benefit; see Assigning benefit points to stories frame. In our example, the result of that task appears in the first "Part of Epic" column (white with green numbers) of Table 3; and the resulting portion of the epic's benefit points and monetary benefit appear in the "BP" and " Benefit" columns immediately to the right.

Story points distributed on stories

Unlike what we recommend for benefit point estimation, story points are commonly estimated directly at story and task levels, often by new planning poker sessions done by scrum teams. If subcontractors deliver code to the project, they might also use in-house methods for cost estimation. In any event, story points for stories are usually not estimated by simply assessing their contribution to epics' cost. The reason why this is fine, is that cost estimations retains relevance all the way down from strategy to construction, especially when subcontractors are involved at the construction level.

It's still advantageous to express story points at the story level in terms of proportions of epic's story points (Fig. 4), since this enables you to relate directly to epics. For our example, we'll assume the proportions of story points for epics as in the second "Part of Epic" column (white with red numbers) of Table 3; and the resulting portion of the epic's cost in the "SP" and "Cost" columns to the right.

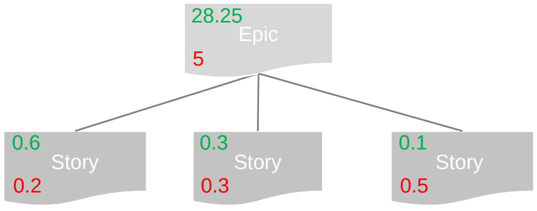


Fig. 4. Proportions of epic's benefit and story points distributed on stories.

Ordering the story backlog

In Table 3, we note that although the three epics *E3*, *E7*, *E2* selected for the first release are the ones that are expected to deliver most benefit for cost, the individual stories within them may not all be as beneficial. Notice that story *E7C* has an unfortunate benefit-cost ratio, and should probably not be put into construction.

The basic principle for ordering the story backlog is straightforward: Order the backlog according to decreasing benefit/cost. If you then put stories into construction in that order, the story next in line will always be the one that is foreseen to generate the most benefit relative to cost in the remaining backlog. If you plot accumulated estimated benefit against accumulated estimated cost as you put your backlog into construction, you get a realization curve with a steep incline easing off, showing how you plan to generate benefit potential faster than cost potential. See Fig. 5 for our example. This is the tactic of maximizing the benefit/cost ratio; i.e., promoting the Benefit/Cost element as most important in the Benefit/Cost Triangle in Fig. 3.

You should assess the information you have available at any point of time and consider revising your plan. Already now, you could plan for dropping *E7C* from the current release. If you do so, you have available capacity in the release to do something more useful. Suppose you take time to elaborate the next epic *E4* in the prioritized line, and that we get the stories as shown in Table 4. In place of the 1.5 in cost for *E7C*, you can plan to spend 0.9 on *E4D* and *E4A*; the two most benefit/cost-efficient stories in this next epic. These have a total estimated benefit of 3.14. If you only had cost to guide you, you might be tempted to fill up the planned capacity of the first release by a full 1.5 cost by choosing, say, *E4C* and *E4D*, but this would just give you the same benefit at a higher cost. Fig. 6 shows the revised plan, where *E7C* has been bumped down the line and out of this release, and where *E4D* and *E4A* have been included into the release instead. Table 5 shows the details of the ordered stories in this revised release. (We'll discuss the two rightmost columns in Table 5 shortly.)

Note that for utilizing available capacity as in this case, we used a different tactic; namely maximizing business value within a fixed cost. This is a variant of the Knapsack problem which is inherent to release planning. So while having an overall tactic for the project of maximizing benefit/cost, you might have to adhere to a fixed cost bound when adjusting a given release; thereby relating temporarily to the Agile Triangle rather than the Benefit/Cost Triangle (Fig. 3).

Now, in an attempt to optimize your plan at this stage, you could elaborate all epics and find the most benefit/cost-efficient stories from the remaining suite of epics and insert those into the free capacity of the release. This would require you to invest more cost earlier—to elaborate the epics—when project experience and knowledge is lower than it might be at a later stage. This is, as it sounds, against agile principles. Still, elaboration

TABLE 5

Stories first release revised sorted by decreasing benefit/cost. Cumulative points and monetary values for cost and benefit estimates and actual or adjusted cost and adjusted benefit.

	Cumulative					
Story	BP	Benefit	SP	Cost	AB	AC
E7A	16.95	6.15	1.00	0.60	6.15	1.20
E2A	39.52	14.33	2.60	1.56	10.24	2.54
E3A	55.83	20.24	4.40	2.64	16.15	2.87
E4D	59.29	21.50	4.90	2.94	17.40	3.77
E3B	66.28	24.03	6.10	3.66	19.94	5.21
E7B	74.75	27.10	7.60	4.56	23.01	6.11
E4A	79.95	28.99	8.60	5.16	24.89	7.01
E2C	88.97	32.26	11.00	6.60	28.17	11.33
E2D	98.00	35.53	13.40	8.04		
E2B	102.51	37.17	15.00	9.00		

has to be done at some point in time, and the decision when to do this is a refinement of whatever tactic you are following, which we leave for another discussion.

Activity: Monitor Earned Business Value

A common way of measuring a project's efficiency is by the metrics of *Earned Value Management* (EVM). In general, EVM relies on having means to quantify work done. Agile accommodates this nicely in its product elements and product backlog.

But you must now brace yourself, because one uses the term "value" in the Earned Value Management regime in a way that confounds cost with value; which really is unfortunate, since people tend to believe that costlier things have inherently more value [1].

Anyway, consider the project in some period of time p, at the end of which you've decided to assess project efficiency. The period may represent a sprint, a release or the entire project up till now. Then,

- *Planned Value* PV is the estimated cost for the user stories planned for completion in p.
- *Earned Value* EV is the estimated cost for those user stories which are actually completed in p.
- *Actual or Adjusted Cost* AC is the cost for the user stories that were completed in p. If your cost estimates are for development only, then AC is the actual cost incurred developing the user stories. If your cost estimates are for life-cycle cost, and you're assuming that life-cycle cost is proportional by a factor L to development cost then AC is the actual development cost multiplied by L. Since AC in the latter case retains an estimate element, you can call this Adjusted Cost rather than Actual Cost.
- *Cost Performance Index* CPI = EV/AC.

Fig. 7 illustrates this for our example. The period in question is the first release, and we planned for producing ten stories *E7A–E2B* with total estimated cost PV = 9.00 (from Table 5). However, the project only managed to complete the first eight stories *E7A–E2C* in time. The total cost estimate for those eight are EV = 6.60. We now have the actual development cost for those eight. We're assuming that life-cycle cost is proportional by a factor 2 to development cost, so based on that, the adjusted cost for the eight stories is AC = 11.33 (from Table 5). In the example, the actual cost for the eight stories developed is more than the estimate for the planned ten. So, the project is both behind schedule and above planned cost. The CPI = 6.60/11.33 = 0.58 is well below 1. A CPI below 1 indicates that you're spending more cost than

Assigning Benefit Points to Stories

Benefit estimation for a story should not relate to objectives directly, but indirectly via the benefit points of its epic. In [9] we suggested a syntax for epics which explicitly mentioned objectives. Here, to help you to think of stories in terms of their contribution to its epic, you can use the following syntax, where objectives are explicitly not mentioned.

> Story: **As** <stakeholder A> **I can** <perform actions d in domain D> **by using** <functionality f in system S> **to** <perform actions s in S> **in order to** <contribute to Epic E>

Epics can be very high level and represent large chunks of functionality. You might be faced with more stories than you can comfortably keep track of when distributing benefit points from the epic. The solution? Adapt available distribution techniques, based on assessing relative importance. We compared four possible techniques in an experiment [3], [2]. From this, the recommended technique is pairwise comparison facilitated by the Analytical Hierarchy Process (AHP) [16], which is easily implemented in a tool. Here's how it goes: The technique is based on the idea that it is cognitively extremely taxing to compare a lot of items in one go. Instead, only consider two items at a time by assessing their relative importance to their epic. But surely this extremely local procedure completely ignores the whole picture and all the relationships between items. Yes, and no. From your judgments on two items at a time, the AHP algorithm deduces a ranking of all the items. The essential detail is that the AHP produces a ranking even in the face of your inconsistencies: Unless you have extraordinary capabilities, your local pairwise comparisons will likely imply *EX* > *EY* and, at the same time, *EY* > *EX* for some stories EX and *EY*. The AHP computes a measure for this inconsistency—the consistency index (CI). In line with *satisficing* rather than optimizing [18], you can make an educated choice as to a "good enough" CI. In standard AHP, all possible pairs of stories in an epic must be compared, which can be fine for a moderate number of stories. In our experiments, we implemented a method to reduce the number of required comparisons [11] (with the penalty of having to be more consistent) so that AHP may be used also for a large number of stories.

Pairwise comparison is a core element of judgment cognitive processes [15]. Using a method that directly supports your cognitive processes is a good way to go to obtain better expert estimates.

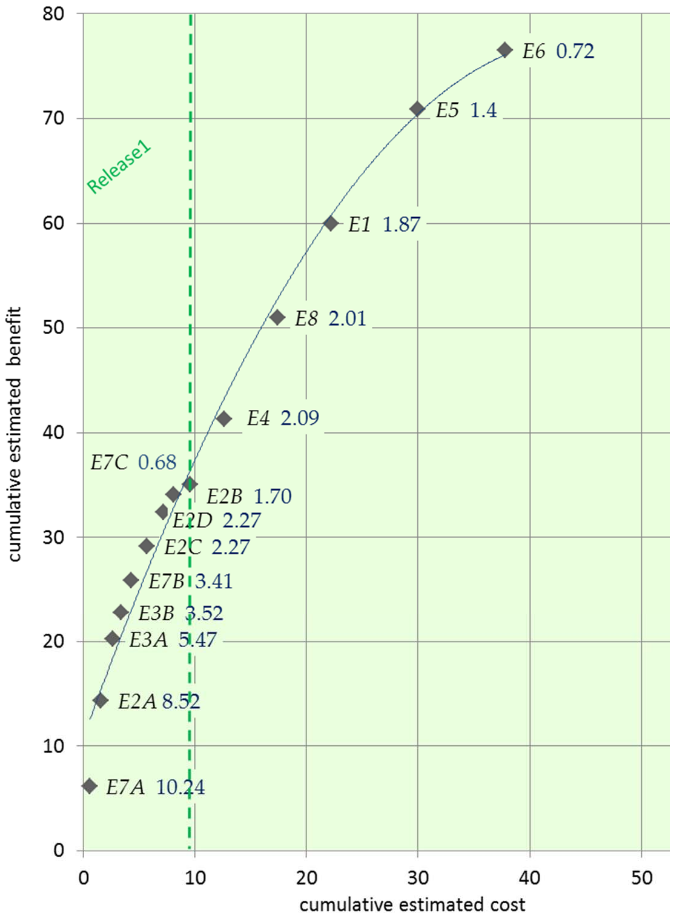


Fig. 5. Planned realization curve for release left of green line. Remainder of queue (non-elaborated epics) right of green line. Benefit/cost values in blue.

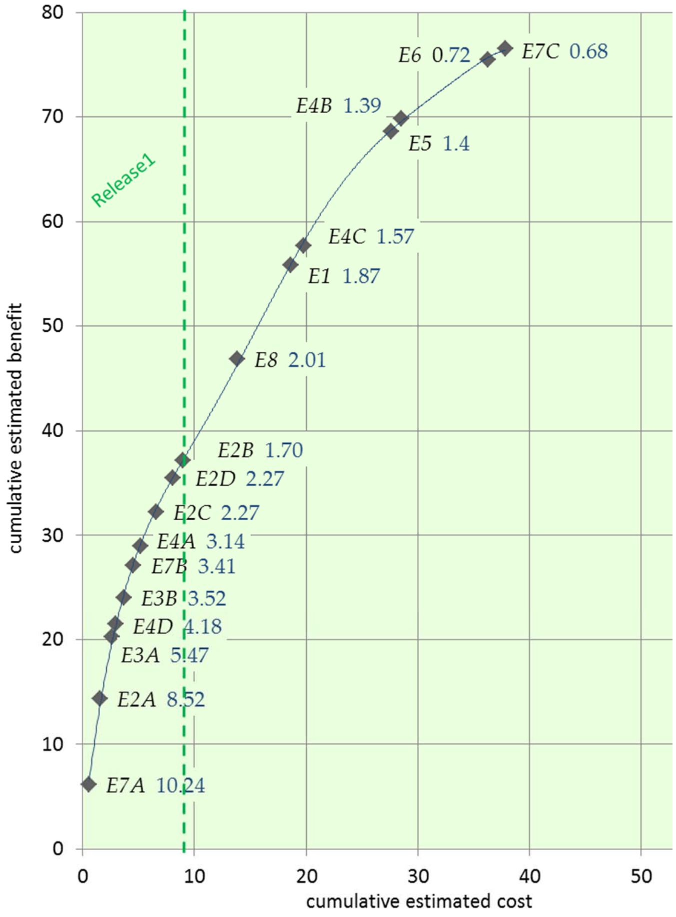


Fig. 6. Planned realization curve for release revised left of green line. Remainder of queue right of green line. Benefit/cost values in blue.

you reckoned for, and the obvious recommendation for a low CPI is to take action so that you get a better CPI in the next period.

As much sense as that is, the CPI is merely a measure of how much functionality is being produced; not how valuable that functionality is. EVM is designed to support management by the Iron Triangle (Fig. 3 left). Therefore, let's define measures for business value.

- *Planned Business Value* PBV is the estimated business value for the user stories planned for p.
- *Earned Business Value* EBV is the estimated business value for the user stories developed in p.
- *Adjusted Benefit* AB is the re-estimated benefit for the user stories that were completed in p.
- *Benefit Performance Index* BPI $=$ EBV/AB.
- *Benefit Cost Performance Index* BCPI $=$ EBV/AC.
- *Adjusted Benefit Cost* ABC $=$ AB/AC.

In our example, PBV $= 37.17$ and EBV $= 32.26$ (from Table 5). Although EBV is less than PBV, the BCPI $= 32.26/11.33 = 2.85$ is well above 1. A BCPI lower than 1 means you're investing more money than you're expecting to gain, and in this case you should consider alternative investments for your money. A cleverly prioritized project will start with a high BCPI, earning much business value compared to the cost expended in the beginning of the project. Here, after the first release, the project's velocity with regards to cost is not good, but the project's velocity with regards to business value is acceptable relative to cost. Such balanced information is important when reporting to project management and project/product owners, but it is also important for virtually every stakeholder on the project, because it gives a wider picture which includes the progress in terms of value for customer, not only amount of functionality.

In traditional EVM, Actual Cost is the expenditure of development. We have generalized this to Adjusted Cost to account for post-deployment cost which has not yet incurred. For our Earned Business Value Management (EBVM) regime, we define the analogous *Adjusted Benefit* AB which is a re-estimate of benefit based on experience from using increments deployed from the project or from other re-estimates of benefit due to e.g., changes in external factors such as legislation, dependencies on the evolution of other systems, and so forth. For our example, let's imagine that *E2A* was found to be overrated by 50% once stakeholders saw the story's functionality in action, so that AB $= 28.17$ (from Table 5), which gives a BPI $= 32.26/28.17 = 1.15$. The BPI is a pure business

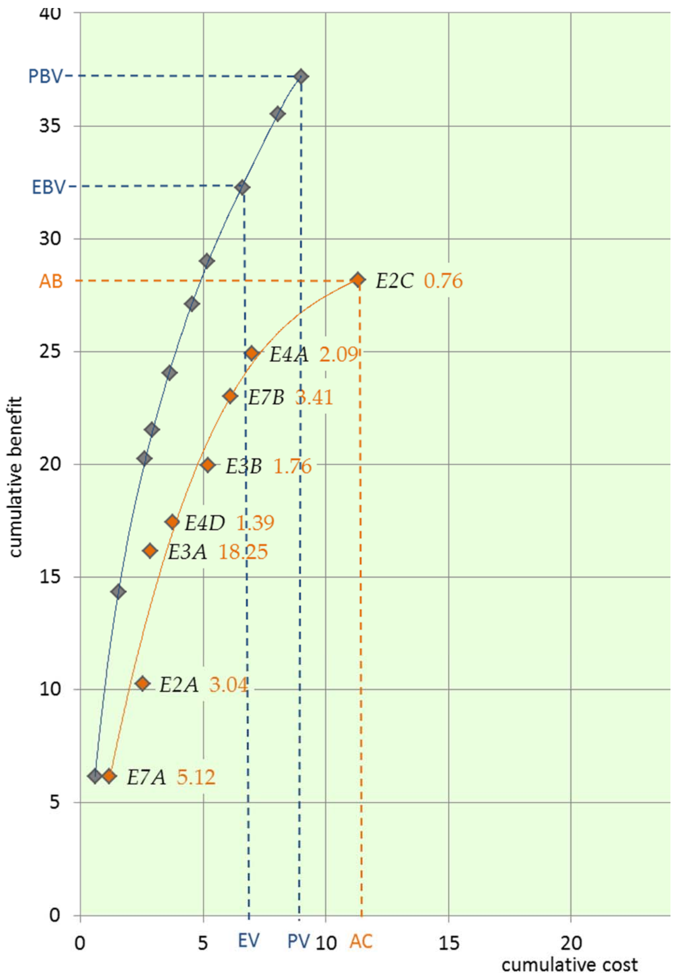


Fig. 7. Planned realization curve (blue) and actual realization curve with AB/AC values (orange) first release.

value metric, and values greater than 1 mean that you're generating less business value than reckoned for. Still, the *Adjusted Benefit Cost* ABC $= 28.17/11.33 = 2.49$ so we're fairing quite good in actuality.

There are a number of further metrics one can derive from the basic ones of both EVM and EVBM. You can now construct your own dashboard for monitoring project efficiency in terms of cost and benefit. You can see that benefit points and story points are at the core of how we define the EVM and EVBM metrics here. In fact, it can be advantageous to make benefit points and story points even more explicit; see Agile EBVM in Practice frame. Benefit points and story points give you means to define a host of metrics that tap directly into your construction line, that at the same time, give meaningful indications in terms of the business case.

Dependencies

Functional, temporal and architectural dependencies between product elements are common place. In addition, worldly factors such as available expertise, illness, conflicts, external constraints, etc. may all influence when stories are put into construction.

We do not treat dependencies as such in this article, and it's important to realize that the perfectly benefit/cost-ordered backlog is an input to the release planning stage, where dependencies are dealt with in full. Our approach is integral to more detailed dependency handling. For example, Cleland-Huang and Denne [7] give a thorough account on the consequences that dependencies have on cost and benefit realization, and they present a heuristic which approximates the optimal ordering of dependency-heavy product elements with respect to return on value (ROI) in the net present value (NPV) regime. Assigning points to product elements would provide the necessary cost and benefit estimates prior to applying such heuristics. Due to dependencies, your backlog might end up differently than perfectly benefit/cost-ordered, but because you assign benefit points and story points, you can keep track of the project's planned and actual productivity even in the turmoil of dependency-driven release planning.

With that said, we claim that dependencies can also be the result of unhealthy architectural work and divisions of functionality into pieces that do not make operational sense. The focus on organizational agility has brought forth concepts such as Minimum Marketable Feature (MMF), Minimum Business Increment (MBI) or Minimum Viable Product (MVP). All these notions embody minimal product elements that add value to customer; the flip-side of the coin being that a product element that is involved in dependencies does not bring value in and of itself. Further, the present focus on capabilities and services [20], [6] stresses the development of independent pieces of functionality that persist over time and in multiple contexts at both business levels and technical levels [10]. If you are in line with these architectural modes, then whenever strong dependencies arise, opportunity may be taken to reconsider how your functionality is divided into pieces. For example, product elements that exhibit strong dependencies may more sensibly be combined into one element.

Wrap Up

With both story points and benefit points in your vocabulary, you can enhance your capability to systematize project knowledge and project learning on aspects that matter the most; namely those of business value. Here, we showed how you can order your product backlog and keep track of productivity, in terms of, not only cost, but also business value. In that light, it's pertinent to ask how one would think of running projects aimed at delivering value for customer, on metrics of cost alone. In another discussion, we'll show how risk and uncertainty can be integrated into this approach.

Acknowledgements

The authors are grateful to collaborators at PROMIS, Metier Academy, the participants on the IT Project Professional (ITPP) certification program and practitioners

Agile EBVM in Practice

When applying Earned Business Value Management, we’ve found it useful to relate to alternative, but equivalent, expressions for CPI and BCPI, that clearly separate points and monetary value. Although less streamlined in definition than the expressions in the main text, our experience is that project stakeholders intuitively understand these metrics better, and that they increase the transparency of the project state. They stimulate you to use EBVM purely based on points and monetary values that may be more accessible, such as total budgeted cost and benefit. We’ve found the effort required to collect data and calculate the metrics to be almost negligible. You’ll likely need to try this out in practice, e.g., in a spread sheet to get to grips with the larger number of expressions, but once you’ve done that, we think you might find this way simpler. Consider, for a given period p,

- PSP—the planned story points
- ESP—the earned story points
- TSP—the total number of story points assigned in the project
- FSP = ESP/TSP—the proportion of the total number of story points that is earned

In our example, the period in question is the first (revised) release (Table 5 and Fig. 7), and PSP = 15, ESP = 11, and TSP = 63. Then, FSP = 11/63 = 17.5%. Further, consider

- VSP—the monetary value of a story point you’re using for the period
- TPV = TSP ∗ VSP—the estimated total life-cycle cost—or total planned value— given VSP
- FC = AC/TPV—the proportion of total planned value that is committed

In our example, VSP = 0.6 million, TPV = 37.80 million, and FC = 11.33/37.80 = 29.97%. With simple math, you’ll verify that

- CPI = FSP/FC

For our example, CPI = 17.5/29.97 = 0.58; the same as calculated earlier the standard way.
Now, consider, for a given period p,

- PBP—the planned benefit points
- EBP—the earned benefit points
- TBP—the total number of benefit points assigned in the project
- FBP = EBP/TBP—the proportion of the total number of benefit points that is earned

In our example (Table 5 and Fig. 7) PBP = 102.51, EBP = 88.97, and TBP = 211. Then, FBP = 88.97/211 = 42.17%. Further, consider

- VBP—the monetary value of a benefit point you’re using for the period
- TPBV = TBP ∗ VBP—the estimated total life-cycle benefit—or total planned business value— given VBP
- FB = AB/TPBV—the proportion of total planned business value that is committed

In our example, VBP = 0.36 million and TPBV = 76.50 million, and FB = 28.17/76.5 = 36.82%. It is easily verified that

- BPI = FBP/FB
- BCPI = FBP/FC ∗ TPBV/TPV

For our example, BPI = 42.17/36.82 = 1.15 and BCPI = 42.17/29.97 ∗ 76.50/37.80 = 1.41 ∗ 2.02 = 2.85; the same as calculated earlier the standard way.

Some of the cost metrics (PSP, ESP, TSP, VSP, TPV) are in line with ideas in e.g., [5], [19], [17]. VSP is often understood as the budgeted baseline cost per story point fixed throughout the project. But you can also choose to have a dynamic VSP reflecting project experience, depending on what kind of “Agile” you’re committed to. You can even equip your dashboard with a fixed VSP and a dynamic VSP’. The same goes for VBP, of course.

applying the methods in this article for giving valuable feed back, comments, and insights to this material.

References

[1] D. Ariely, Ed., *Predictably Irrational: The Hidden Forces That Shape Our Decisions*. Harper Perennial, 2010.

[2] H.C. Benestad and J.E. Hannay, “A comparison of model-based and judgment-based release planning in incremental software projects,” in *Proc. 33rd Int’l Conf. Software Engineering (ICSE 2011)*. ACM, 2011, pp. 766–775.

[3] H.C. Benestad and J.E. Hannay, “Does the prioritization technique affect stakeholders selection of essential software product features?” *Proc. 6th Int’l Symp. Empirical Software Engineering and Measurement (ESEM)*, pp. 261–270, 2012.

[4] M. Cohn and R. Martin, *Agile Estimating and Planning*. Prentice Hall, 2005.

[5] CollabNet, “Monitoring Scrum projects with agile EVM and earned business value measures,” 2013.

[6] M.H. Danesh and E. Yu, “Analyzing IT flexibility to enable dynamic capabilities,” in *Advanced Information Systems Engineering Workshops*, ser. Lecture Notes in Business Information Processing. Springer, 2015, vol. 215, pp. 53–65.

[7] M. Denne and J. Cleland-Huang, “The incremental funding method: Data-driven software development,” *IEEE Software*, vol. 21, no. 3, pp. 39–47, May/June 2004.

[8] G. Gigerenzer and P.M. Todd, Eds., *Simple Heuristics that Make Us Smart*. Oxford University Press, 1999.

[9] J.E. Hannay, H.C. Benestad, and K. Strand, “Benefit points—the best part of the story,” *IEEE Software (to appear)*, 2016.

[10] J.E. Hannay, K. Brathen, and O.M. Mevassvik, “Agile requirements handling in a service-oriented taxonomy of capabilities,” *Requirements Engineering*, 2016, online http://dx.doi.org/10.1007/s00766-016-0244-8.

[11] P.T. Harker, “Incomplete pairwise comparisons in the analytic hierarchy process,” *Mathematical Modelling*, vol. 9, no. 11, pp. 837–848, 1987.

[12] International Institute of Business Analysis, “A guide to the business analysis body of knowledge,” 2009.

[13] C. Jones, *Estimating Software Costs: Bringing Realism to Estimating*, 2nd ed. McGraw-Hill, 2007.

[14] M. Jørgensen, “Forecasting of software development work effort: Evidence on expert judgement and formal models,” *Int’l J. Forecasting*, vol. 23, pp. 449–462, 2007.

[15] T. Mussweiler, “Comparison processes in social judgment: Mechanisms and consequences,” *Psych. Review*, vol. 110, no. 3, pp. 472–489, 2003.

[16] A. Perini, F. Ricca, and A. Susi, “Tool-supported requirements

prioritization: Comparing the AHP and CBRank methods," *Information and Software Technology*, vol. 51, no. 6, pp. 1021–1032, 2009.

[17] D. Rawsthorne, "Calculating earned business value for an agile project," http://www.danube.com/system/files/CollabNet_WP_Earned_Business_Value_041910.pdfl, 2010, accessed May 2013.

[18] H.A. Simon, *The Sciences of the Artificial*, 3rd ed. MIT Press, 1996.

[19] T. Sulaiman, B. Barton, and T. Blackburn, "AgileEVM— earned value management in Scrum projects," in *Proc. IEEE AGILE 2006*. IEEE Computer Society, 2006, pp. 7–16.

[20] The Open Group, *SOA Reference Architecture Technical Standard*, 2011, doc. no. C119.