Earned Value Management (EVM) for Agile Software Projects

Olga Yatskevich

07/15/2017
11 minutes

Agile software projects deliver business value by embracing change. Whatever business gains, this puts extra responsibility on project management:

- How to control resources in forever evolving conditions?
- How to ensure clarity and precision of cost calculations?

On the other hand, every stakeholder wants to know:

- How well is the project doing?
- Will it be on time/within budget?
- How much more money/time will it take to complete this amount of work?
- Do we need to take longer and accomplish all the features or cut short and make do with bare necessities?

Uncertainty about using Agile is still strong since Scrum framework seems to be missing the guidelines on cost and schedule control.

Performance metrics in Scrum projects

In fact, Agile employs a number of quantitative metrics for various parameters:

- **Story/feature/epic count** – tracks functionality on a release level and beyond;
- **Story points** – estimate overall size and effort for a release;
- **Velocity** – a measure of sprint-sized working functionality and overall progress pace;
- **Sprint burndown** – a daily estimate of work remaining in a sprint;
- **Release burnup** – a comparison of actual effort against planned for a release.

However, businesses don't find these metrics transparent and informative enough in terms of cost efficiency control. Agile EVM includes this missing cost component as an **Earned Value** metric.

What does Agile EVM do better than a release burnup chart

Agile EVM is more informative

Agile EVM provides the data of **cost**, **performance** and **schedule** allowing to:

- Compare monetary expressions of value planned, earned and consumed at any given moment
- Calculate cost and schedule efficiency
- Make forecasts for the end-of-the-project time and cost

Agile EVM embraces change

It also deals efficiently with changes in plans. The most unhealthy practice in software projects is to allow informal changes to the technical baseline without changing the cost of schedule baselines–a practice commonly known as scope creep. In Agile EVM, scope change is reflected in a variable known as *Scope Floor*. Cost and time-related metrics register this change and other planning variables are adjusted to meet new conditions.

Agile EVM ties together abstract and time-bound valuations of effort

Another reason why Agile EVM is more informative is that it allows to tie in Scrum's abstract valuations of size and complexity (user stories and story points) with more realistic metrics of time and cost. As Jeff Shurts puts it, 'You'll read over and over again in the literature that a story point is not(!) a representation of time, or of effort – merely one of scope or complexity. It bears relation to effort only through another agile measure – velocity'. The rationale behind abstract valuations is simple: Agile shifts the focus from deadly deadlines towards generating business value.

Where does Agile EVM derive from

Earned Value Management is not a product of Agile environment. It has been around since the 1960's, a recognized project management technique, the subject of in-depth study by the Project Management Institute's (PMI) College of Performance Management and now included as a standard in the "Guide to the Project Management Body of Knowledge". Lately it has also become a successful practice in Scrum, introduced via Agile framework by Tamara Sulaiman, Brent Barton, and Thomas Blackburnin 2006.

Agile EVM: key concepts, formulae, and output

Here are the concepts of Agile EVM and their mathematical representation.

Short	Concept	Definition	Formula
EPC	Expected Percent Complete	The amount of value planned to be delivered over the period in question	Current sprint / Total planned sprints (PS)
APC	Actual Percent Complete	The amount of value actually delivered over the period in question	Story points completed / Total planned story points (PRP)
PV	Planned Value	The budgeted cost for the work scheduled to be completed up to a given point in time. Formerly known as **BCWS** (Budgeted Cost for Work Scheduled)	EPC * BAC
EV	Earned Value	The budgeted amount for the work actually completed during a given time period. Formerly known as **BCWP** (Budgeted Cost for Work Performed)	APC * BAC
AC	Actual Cost	Corresponds to budgeted costs in PV for release. Formerly known as **ACWP** (Actual Cost for Work Performed).	(actual sum added manually)
BAC	Budget at Complete	The initial budget for the release – coincides with PV for the entire project	(agreed upon at the inception)
SV	Schedule Variance	The (monetary) difference between the planned and actual time to complete a certain workload	EV – PV
CV	Cost Variance	The (monetary) difference of value incurred and cost to produce a certain piece of work over a certain amount of time	EV – AC
SPI	Schedule Performance Index	An efficiency indicator for measuring the rate at which the project's throughput is meeting initial schedule expectations.	EV / PV
CPI	Cost Performance Index	An efficiency indicator for measuring the value produced for each $1 of actual cost.	EV / AC
EAC	Estimate at	Expected *total* cost of the project at its end	EAC = BAC / CPI_{cum}

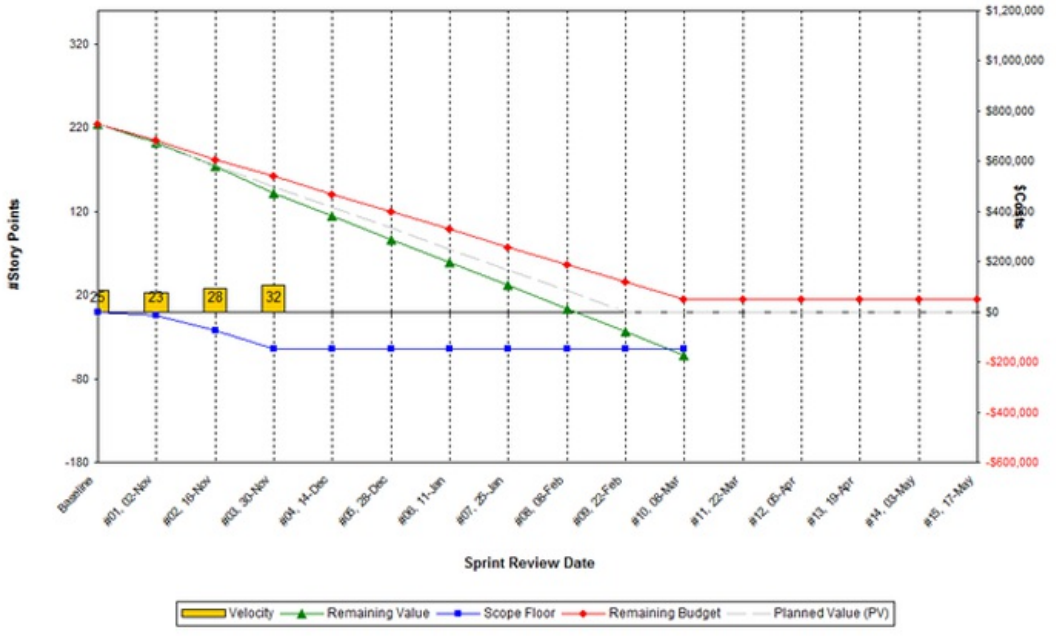
Short	Complete Concept	Definition	Formula
			EAC = AC_{cum} + BAC − EV_{cum} EAC = AC + ((BAC − EV) / CPI)
ETC	Estimate to Complete	Expected *additional* cost needed to complete the project or*from the point of calculation*	ETC = (BAC − EV_{cum}) / CPI_{cum} ETC = EAC − AC_{cum}
VAC	Variance at Complete	The difference between a planned and actual project's cost	VAC = BAC − EAC
TCPI	To-Complete Performance Index	Projected efficiency index	TCPI = (BAC-EV_{cum}) / (BAC-AC_{cum})

They are visualised in a spreadsheet with both automated and manual input fields.

fx Remaining Value (EV)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Sprint	1	2	3	4	5	6	7	8	9	10	11	12	13	
2	time per sprint (h)	80	80	80	80	80	80	80	80	80	80	80	80	80	
3	time cost ($/h)	5	5	5	5	5	5	5	5	5	5	5	5	5	
4	Story Points (at the start)	100													
5	Actual Story Points (+Scope Floor)	100	100	120	120	110	110	110	120	130	130	130	130	130	
6	Story point price	$40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	$ 40.00	
7	Scope floor	0	0	20	20	10	10	10	20	30	30	30	30	30	
8	(+) Story points added (cum)	0	0	20	20	20	20	20	30	40	40	40	40	40	
9	(-) Story points removed (cum)	0	0	0	0	10	10	10	10	10	10	10	10	10	
10	BAC	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	$4,000.00	
11	Story points planned (it)	10	10	10	10	11	12	12	12	13	13	14	13	13	
12	Story points planned backlogged	90	80	90	80	59	47	35	33	30	17	3	-10	-23	
13	Story points planned (cum)	10	20	30	40	51	63	75	87	100	113	127	140	153	
14	Expected Percent Complete	10.00%	20.00%	30.00%	40.00%	51.00%	63.00%	75.00%	87.00%	100.00%	113.00%	127.00%	140.00%	153.00%	
15	Story points done (it)	10	10	11	5	11	11	11	11	11	11	11	11	11	
16	Story points done (cum)	10	20	31	36	47	58	69	80	91	102	113	124	135	
17	Story points backlogged	90	80	89	84	63	52	41	40	39	28	17	6	-5	
18	Actual Percent Complete	10.00%	20.00%	25.83%	30.00%	42.73%	52.73%	62.73%	66.67%	70.00%	78.46%	88.92%	95.38%	103.85%	
19	PV	$400.00	$800.00	$1,200.00	$1,600.00	$2,040.00	$2,520.00	$3,000.00	$3,480.00	$4,000.00	$4,520.00	$5,080.00	$5,600.00	$6,120.00	
20	EV	$400.00	$800.00	$1,033.33	$1,200.00	$1,709.09	$2,109.09	$2,509.09	$2,666.67	$2,800.00	$3,138.46	$3,476.92	$3,815.38	$4,153.85	
21	AC	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	$440.00	
22	AC total	$440.00	$880.00	$1,320.00	$1,760.00	$2,200.00	$2,640.00	$3,080.00	$3,520.00	$3,960.00	$4,400.00	$4,840.00	$5,280.00	$5,720.00	
23	SV	$0.00	$0.00	-$166.67	-$400.00	-$330.91	-$410.91	-$490.91	-$813.33	-$1,200.00	-$1,381.54	-$1,603.08	-$1,784.62	-$1,966.15	
24	CV	-$40.00	-$80.00	-$286.67	-$560.00	-$490.91	-$530.91	-$570.91	-$853.33	-$1,160.00	-$1,261.54	-$1,363.08	-$1,464.62	-$1,566.15	
25	SPI	$1.0	$1.0	$0.9	$0.8	0.84	0.84	0.84	0.77	0.70	0.69	0.68	0.68	0.68	
26	CPI	$0.9	$0.9	$0.8	$0.7	0.78	0.80	0.81	0.76	0.71	0.71	0.72	0.72	0.73	
27	ETC	$3,960.00	$3,520.00	$3,789.68	$4,106.67	$2,948.94	$2,366.90	$1,830.14	$1,760.00	$1,697.14	$1,207.84	$728.14	$255.48	-$211.85	
28	EAC	$4,400.00	$4,400.00	$5,109.68	$5,866.67	$5,148.94	$5,006.90	$4,910.14	$5,280.00	$5,657.14	$5,607.84	$5,568.14	$5,535.48	$5,508.15	
29	VAC	-$400.00	-$400.00	-$1,109.68	-$1,866.67	-$1,148.94	-$1,006.90	-$910.14	-$1,280.00	-$1,657.14	-$1,607.84	-$1,568.14	-$1,535.48	-$1,508.15	
30	TCPI	1.01	0.90	0.83	0.79	0.64	0.53	0.42	0.37	0.34	0.24	0.15	0.05	-0.04	
31	Comments			New 20 Story Points are added as CR					new 10 SP	new 10 SP added					
32	Status														

They are also presented as a graph.



Release burndown chart for Agile EVM with scope increase (Courtesy of Chris Fortuin)

Step-by-step Agile EVM implementation via spreadsheet

A spreadsheet – is a perfect format for Earned Value Management. It automates basic calculations but allows for control and adjustment, thus providing simple and clear Agile EVM implementation.

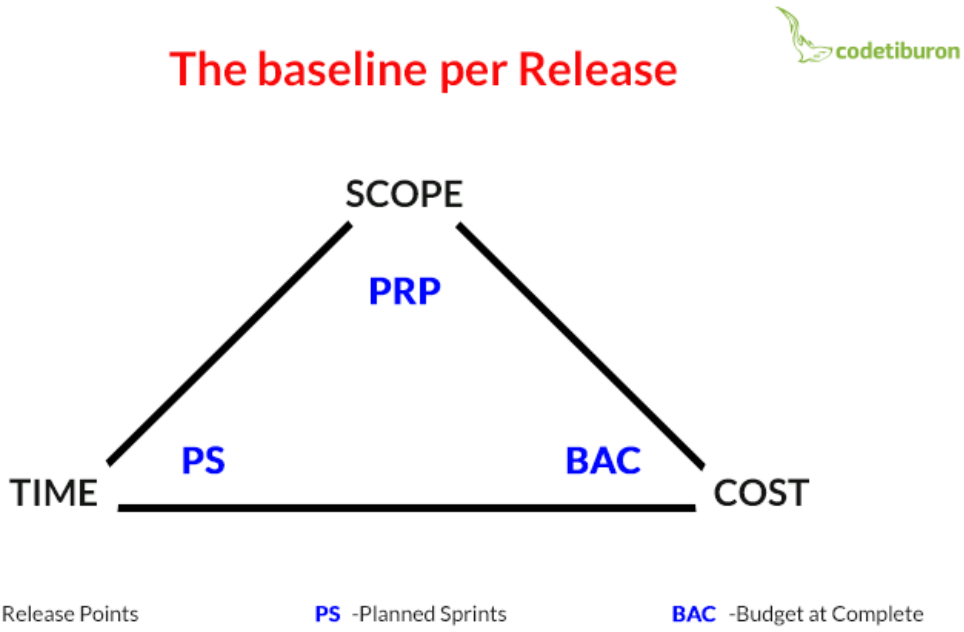
Project-planning stage

Planning any project involves estimating scope, time, and cost – that's the **project manager triangle**. The planning stage provides the necessary input data to use as a baseline for EVM calculations. Of course, we can only use historic data and intuition so far.

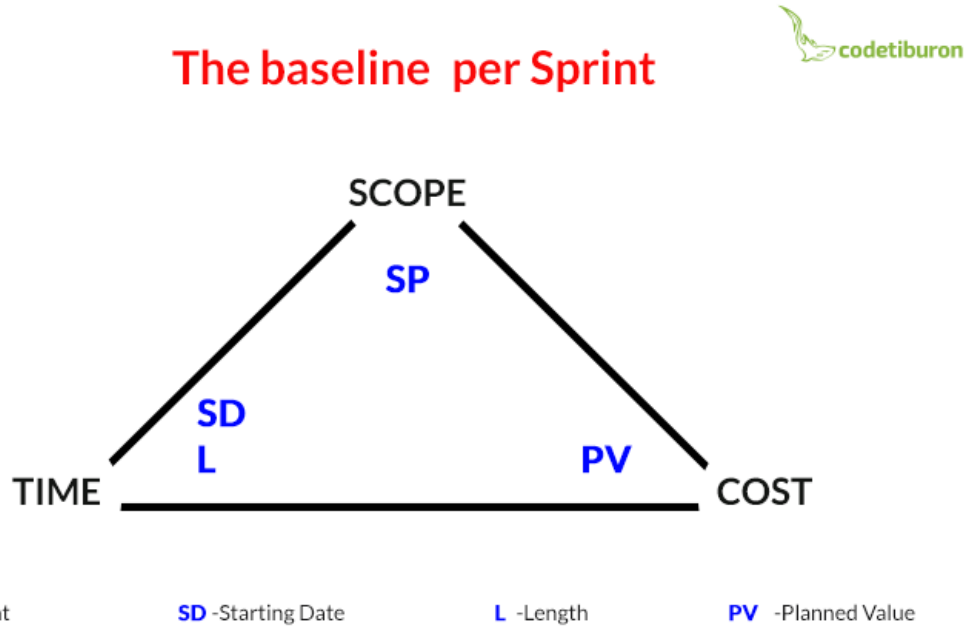
We are particularly interested in planning on the release and sprint levels.

What metrics to use as input data

When planning for a **release**, we estimate the *scope* in 'planned release points' (PRP), *time* – in 'planned sprints' (PS), and *cost* – in 'budget at complete' (BAC).



For a **sprint**, the currency for *scope* is 'story points' (SP), for *cost* – 'planned value' (PV), and *time* metrics include 'starting date' (SD) and 'length' (L).



All in all, following Sprint Zero, we have such input data:

Short	Metric	Definition

Short	Metric	Definition
PS	Planned Sprints	Total number of planned sprints
BAC	Budget at Complete	Budget planned for the release
L	Length	Sprint length (in calendar days)
SD	Start Date	Start date of the project
PRP	Planned Release Points	The number of story points planned for the release, i.e. the total scope (in points)

Deriving project data

After each sprints, we collect the actual data. We have the metrics for:

- The number of sprints completed
- The number of story points done
- the actual cost (AC)

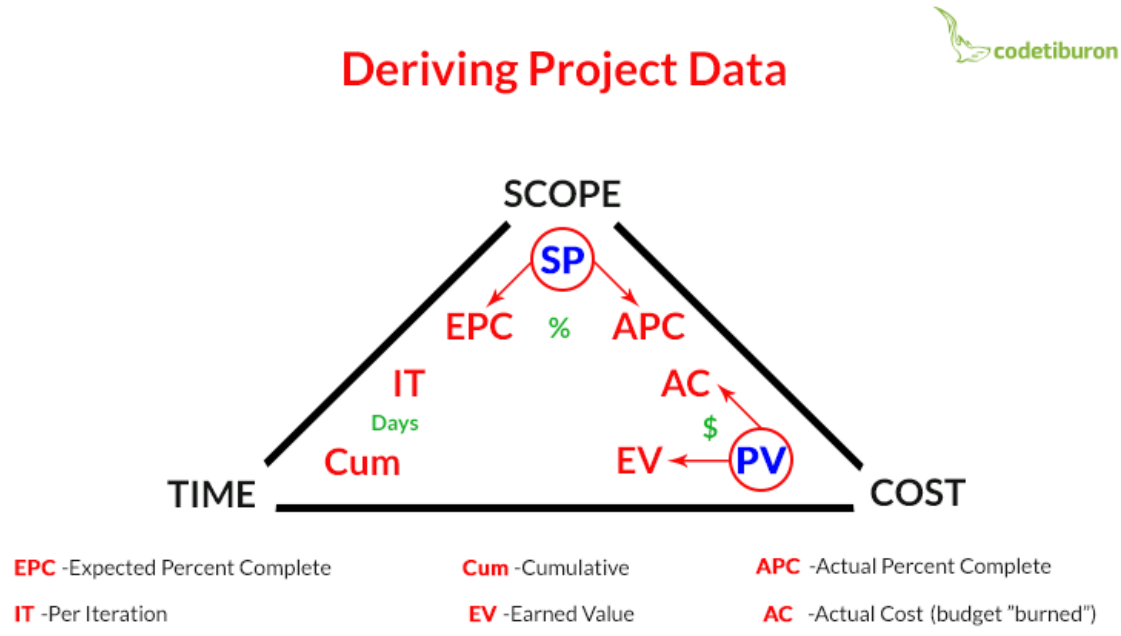
With these values, we can calculate and compare:

- the percent of scope done (APC, actual percent complete) against planned (EPC, expected percent complete)
- the monetized value earned (EV, earned value) over 'burned' (AC, actual cost)

The *time* variable (number of sprints completed) is expressed via concepts 'per iteration' ($_{it}$) and 'cumulative' ($_{cum}$) in our calculations.

Deriving these metrics helps:

- compare guesses for cost, time, and scope with facts;
- determine efficiency (of planning and performing);
- rebaseline the project estimates for more accurate forecasts.



Measuring project performance

Now we can see how well we are doing The Cost Performance Index (CPI) and Schedule Performance Index (SPI) give a measure of efficiency. They show **how efficiently you are actually spending your budgeted costs and time** as compared to how efficiently you have planned to spend them.

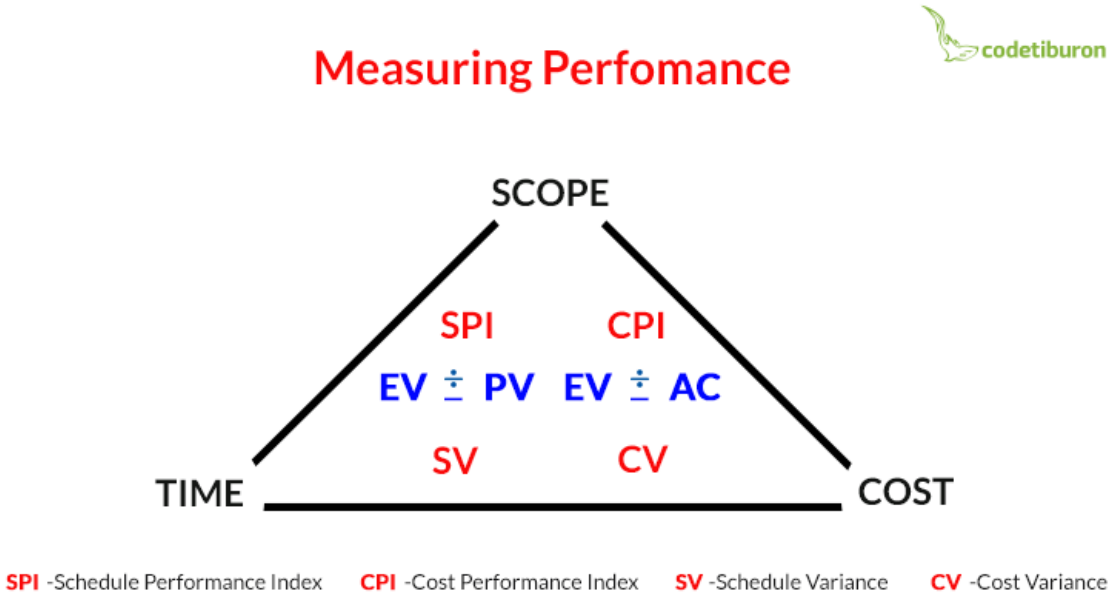
For CPI, divide Earned Value by Actual Cost.

CPI > 1	CPI = 1	CPI < 1
Under Budget	On Budget	Over Budget
EV > AC	EV = AC	EV < AC

SPI is Earned Value divided by Planned Value.

SPI > 1	SPI = 1	SPI < 1
Ahead of schedule	On schedule	Behind schedule
EV > PV	EV = PV	EV < PV

Another way to measure efficiency is by calculating schedule and cost variance: SV = EV – PV; CV = EV – AC**Positive** variance is good while **negative** means trouble: your project is costing you more than you've planned or earned so far. This can indicate one of these: either your estimates are bad, or you're moving too slow. In any case, if you continue at this pace, you'll probably have to reschedule the release date or cut functionality. Readjust your plan and keep stakeholders informed.



Dealing with scope change

Change is inevitable in Agile projects. It may come from within (greater size of stories is revealed over time, which means more story points; velocity change; team size change) or without (more/less functionality is needed than originally planned; re-prioritization takes place). Agile EVM optimises the value of iterations when the new input data automatically become a new baseline for performance calculations.

When dealing with scope change, Agile EVM differentiates between the concepts of 'change' and 'adjustment'. 'Changes' happen accidentally, without intention, whereas 'adjustments' are done on purpose by the PO for precision and efficiency.

Here are some cases of 'changes':

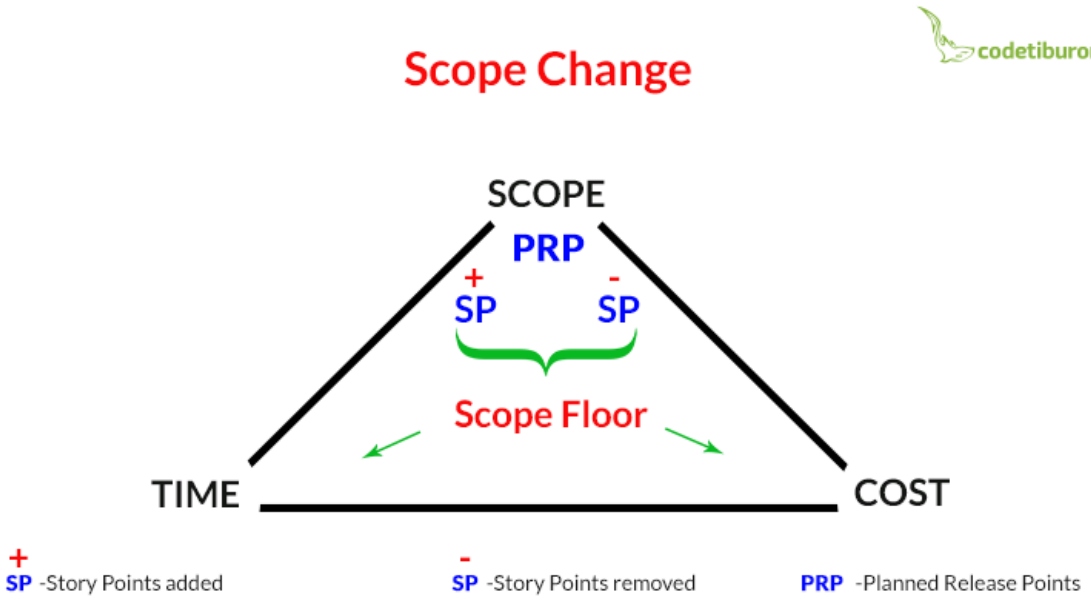
- velocity fluctuates because defects are found during development or features are not accepted during Sprint Review or for many other reasons;
- scope may change after feedback for Sprint review or when defects are found in production.

There is no need for adjustment here.

When do we 'adjust' the scope or velocity variables?

- if PO adds a new feature to Product Backlog;
- when a developer is leaving the team (but do not re-adjust the velocity if a new developer is added).

In Agile EVM, scope change is reflected in a variable known as Scope Floor (SF). This is actually scope 'adjustment' due to adding new SP or removing planned SP undone.



Making forecasts about cost and schedule

You can't find a better predictor of near-term future performance than near-term past performance. Since every sprint is a new baseline, instead of using the INITIAL estimate in total number of Story Points, Agile EVM calculations always use the LATEST estimate in total number of Story Points to calculate CPI, SPI, EAC, etc. – to achieve greater precision of predictions.

What sort of forecasts can we make?

First of all, we predict what total cost to expect at the end of the project based on the changes to the project's scope and schedule. *Estimate at Completion* (EAC) is best calculated using the formula **$EAC = AC_{cum} + BAC - EV_{cum}$** that embraces changing conditions.

Next, *Estimate to Complete* (ETC) gives us a forecast for the costs to be injected from the current date to the end. Either of these formulae can do the job:

- **$(BAC - EV_{cum}) / CPI_{cum}$**
- **$ETC = EAC - AC_{cum}$**

We can also derive information about cost discrepancy between the planned and would-be consumed budget long before the project's finishing line. Here's how to derive *Variance at Complete* (VAC):

- **$VAC = BAC - EAC$**

Finally, we can predict the project's projected efficiency based on our current metrics. Here's the formula for the *To-Complete Performance Index*:

- **$TCPI = (BAC-EV_{cum}) / (BAC-AC_{cum})$**

Forecasts allow us to make informed decisions about the best option of when, at what cost, and with what feature set to go to release

To conclude

Successful implementation of Agile EVM depends on timely data input and accurate estimation. But the benefits are worth the effort.

The possibility of visualization as well as bringing time-bound and abstract estimates closer together makes Agile EVM metrics a **perfect communication tool** with businesses.

The technique for cost and time control **adds value to Agile development** over the traditional, waterfall, model and as such makes this methodology more reliable to those who would otherwise shy away from Agile.

Resources

1. Tamara Sulaiman. AgileEVM: Measuring Cost Efficiency Across the Product Lifecycle.
2. Tamara Sulaiman, Brent Barton, Thomas Blackburn. Agile EVM: Earned Value Analysis in Scrum Projects.
3. Tamara Sulaiman, Hubert Smits. Measuring Integrated Progress on Agile Software Development Projects.

agile, cost management of agile software projects, earned value management

Did you enjoy the read? Was it useful? Your sentiment helps us to create better content. Use the reactions to assess the article. Or leave us a note in the comments. We are out here to boost your tech savvy.

Pff

Meh

Hmm

Mmm

Wow!

4.6/ 5.0 rating (68 votes)

2 Leave a Reply

Connect with

newest oldest most voted

D Duy Hoang

Very helpful post. May I have the spreadsheet mentioned in the post?

0

3 months ago

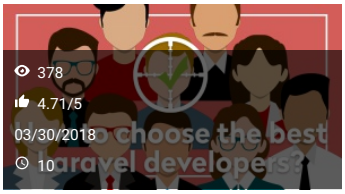
Olga Yatskevich

Thank you for taking an interest, Duy Hoang! The spreadsheet is currently unavailable. But you may find some useful tools on Chris Fortuin's site (see the link in the article).

0

3 months ago

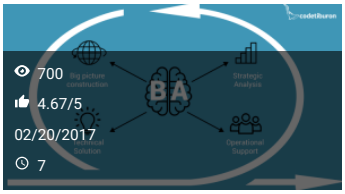
ALSO READ RELATED ARTICLES



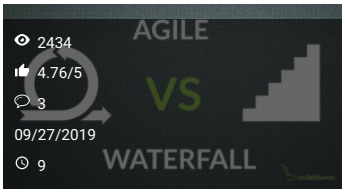
How to choose the best Laravel developers

How to Manage Cost and Time of Agile Software Project

Software Project Estimation Practices at CodeTiburon



Business Analyst in a Scrum Team of Software Developers



Agile or Waterfall: Choose the Right Approach to Your Software Project Management

codetiburon