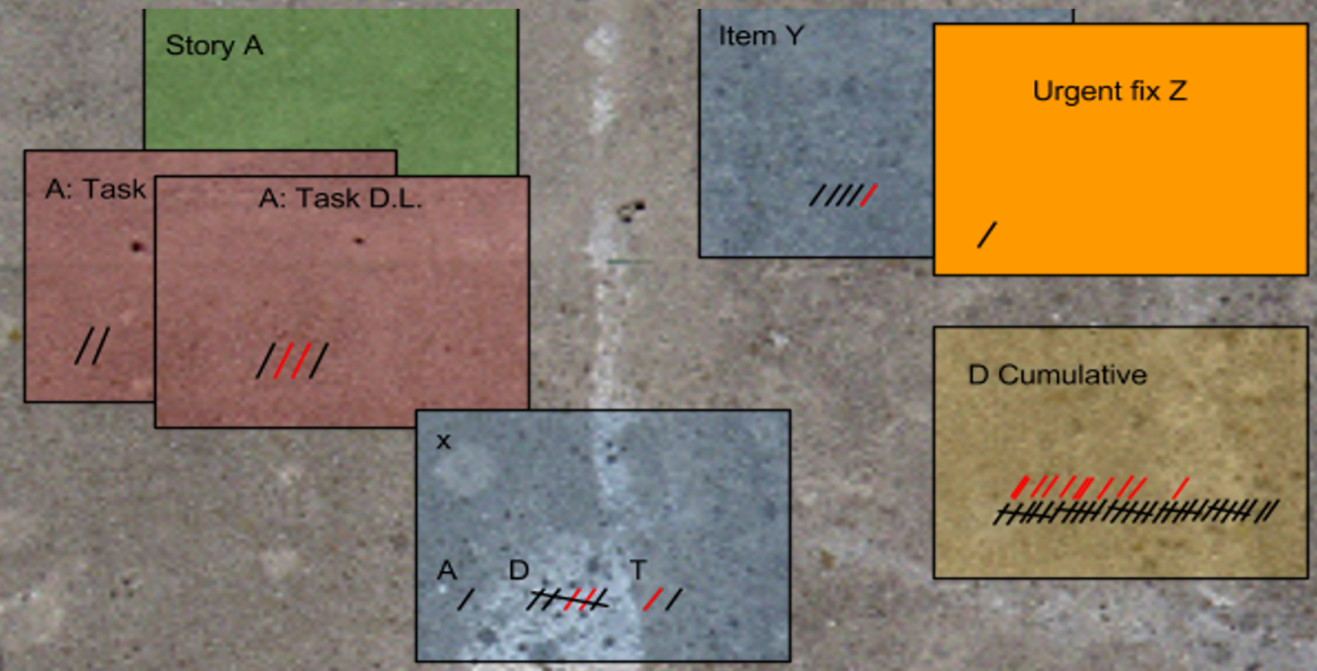
The Prisoner Metrics



a kanbanish tool

The Prisoner Metrics

A kanbanish tool

Antonio Lucca

This book is for sale at http://leanpub.com/prisonersmetrics

This version was published on 2014-02-08

Contents

The Prisoner Metrics

Introduction

The idea started during an internal “coach community meeting”, and originated some concrete practices implemented in many teams during an Agile-Lean corporate initiative.

Basic practices

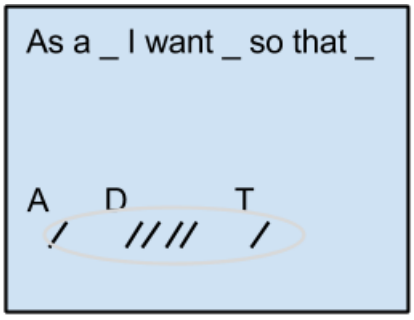
The Prisoner Metrics is a “kanbanish” technique, based on the following practices:

1. tracking regularly, using black checkmarks (‘/’), the time that any card is spending in any stage.
2. tracking with red checkmarks significative impediments found in any card, and the time you are spending to manage them.
3. make some cumulative measures manually, by rearranging the cards using specific rules.

Cycle Time and impediments will be more visible in this way, and the hypothesis that this visibility matters is justified by the theory of nudges[1]

Cycle Time

With Prisoner Metrics you periodically update any card, drawing black checkmarks like ‘/’ on it:



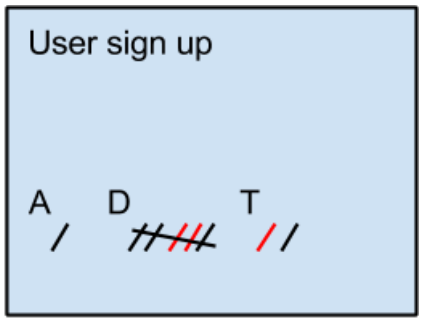
(A, D, and T = Analysis, Development, and Test in this case).

[1] nudges are ways to present alternative in a way that some behavior will be more likely (without forbidding alternatives) http://www.amazon.it/Nudge-Improving-Decisions-Health-Happiness/dp/014311526X

Impediments

Using the red checkmarks, you track the presence of “impediments” and the time you are spending to solve them.

Here is an example of a card that in the Analysis stage spent one day working; in development spent five days: two working, two “blocked”, and one working again; in test it spent one day “blocked” and one working.



To avoid an ambiguous or arbitrary use of “impediment” (or block) you may need to stipulate a clear definition of it. Such definition should be such that the majority of things that you track are the ones that will actually give you opportunity to learn and share new lessons.

The definition of impediment may be refined through an iterative process.

You may need to look at the ratio between the total number of solved impediments and the total number of impediments just tracked. If it is too small, then then you are probably tracking too much respect of what you can actually solve. A prioritization and a limit on the things tracked as impediment may help.

Some scenarios of using the “impediment” mark:

1. You fix the problem now (software project example: you do some refactoring on some area you just now realized is affected by technical debt, and so you’ll use red checkmarks to track the time you’ll spend on it)
2. You take note that the problem exists, but you don’t fix it at the moment (still using the same example of technical debt: the decision to keep the technical debt as it is, and so you use - at most - one red checkmark).

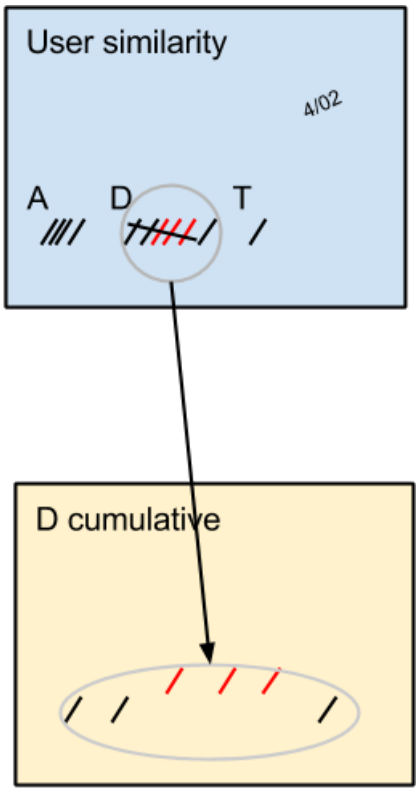
Cumulative measures

Here is one way of doing cumulative measures.

Consider we want to get information like the total time spent by some items (for example during last Sprint - or iteration) in a particular stage, and see the distribution of the impediments and the time spent on them across that stage.

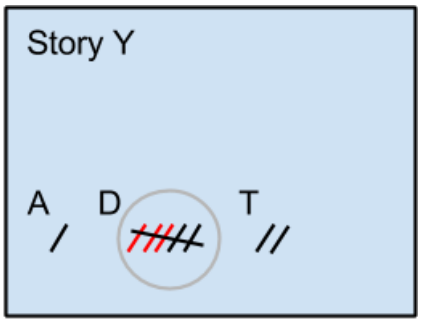
The procedure is based on rewriting somewhere the black and the red checkmarks of the cards of a given period, in a way that respects their original position.

Example:

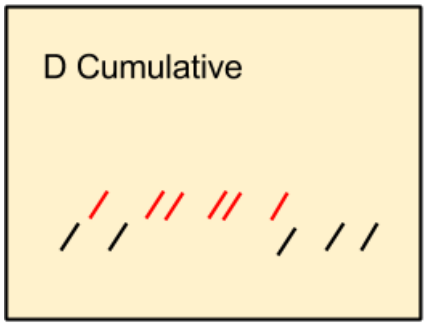


The checkmarks of the ‘D’ stage of a the card are just rewritten to the “cumulative card”, keeping more space between the checkmarks, and with the reds in a different line.

You do again the same for another story.

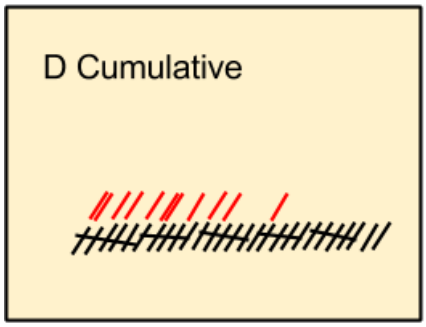


The result of reporting in the same way the marks to the “cumulative” card, summed to the previous one, will be like this:



... and so on, repeated on all the cards of the given period (it can also be a random sample), using the same cumulative card.

At the end of this "rewriting" procedure the cumulative cards may look as follows:



This card shows that "interruption/blocked" distribution is skewed a little bit on the left, among the overall stories. Useful to remember when trying to figure out hypothetical root cause, and possible countermeasures, to set a next P.D.C.A. process improvement step [2].

Calibrating estimates

There are various arguments pro-estimates. For example a pro-estimates argument is that doing estimates create opportunity for conversations about difficulties and scope, or give support to decisions about prioritization in term of cost-benefit tradeoff. Cons-aguments can be to the fact that estimating is an activity that takes time, and that the results may be really poor, when comparing actuals vs estimates.

Beside this, before introducing the estimating activity of cards as a regular activity, you may want to make an assessment of your actual estimating skills first. You can try at first a simple and fast way

[2] "Plan Do Check Adjust". See also Toyota Kata

of doing estimates, where reference sizes are only Small and Large. Being able to this two valued estimates is a necessary condition for being able to do more complex (e.g. multi-sizes) estimates.[3]

Estimating an item as Big means guessing that its actuals (i.e. its total number of its checkmarks - reds and blacks) will be bigger than the actuals of any other item estimated as Small.

For visualization simplicity in this example the green cards are the items that did estimate as "Small" and the pink as "Large". The estimates perfectly fulfill the expectation if Small and Big form two perfect clusters after reordering by actuals:

So the necessary condition that we have seen for doing any kind of estimates is satisfied, and you may want to add a new reference size in the estimates next time (Small, Medium and Large for example).

In the following case there is less correlation:

The actuals of the Small (light green) items are just a little bit more likely to be lower than the actuals of the Big ones, but such chances are no such better than 50%.

In this case, estimates is just a little bit better than tossing coin based guesses.

Theoretically you could also estimate the confidence levels (from 50% to 100%) in advance, and include them in your estimates. It is possible to improve this kind of estimates+confidence level by calibration exercises [4].

An experiment could be the reordering by actuals twice: one time counting only blacks, and then counting blacks and reds (and perhaps, just reds only). I guess that the number of black checkmarks are more related to the "essential complication" of the feature, and the reds to the "extra/accidental complication" of them. I am not suggesting specific way to interpret the results of such kind of experiments, except for the general idea that they may just confirm the impact that unknown have on bad results of the estimates.

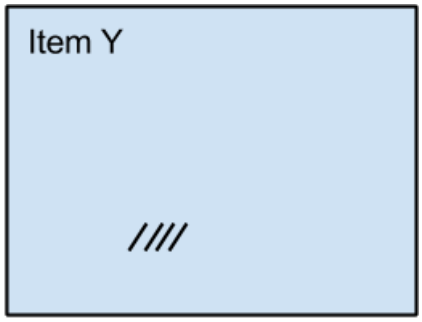
Making explicit evidence of the causes of an Interruption

A block can be caused by task switching on another "more urgent" activity. The card representing the urgent activity can just be placed over the current card, to make even clearer the reason of the

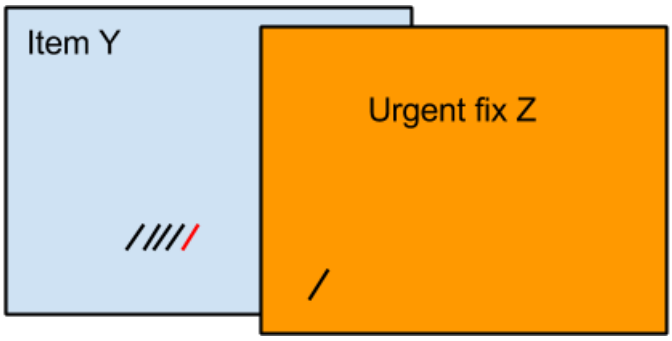
[3] If we can do any kind of estimate then we should be able do, even better, estimates of two size (Small and Large). By "any kind" I mean more valued sizes like S, M, L, XL in sizes or 1,2,3,5, 8, ... in Story Points, and so if we realize we can't do two value estimates, then we cannot do any more complex type of estimates

[4] In "How to Measure Anything" you can find the general idea of how to include uncertainty level in measuring partially unknown values. More important: you can find exercises about calibrating estimates. See also wikipedia entry: Calibrate probability assessment

interruption.



Z is coming and interrupts Y:



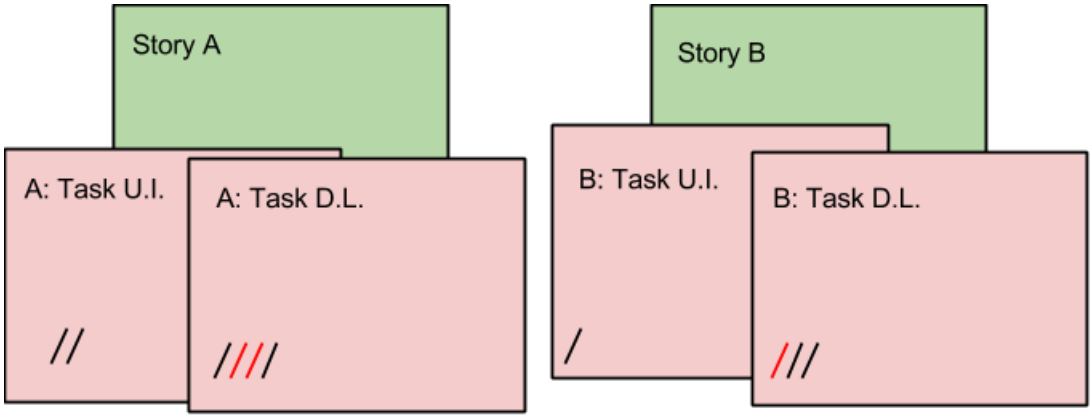
So you keep connection to the source of the interruption. This is particularly useful if the cause is actually external, and you wnant to have conversations about how to remove external interruptions.

Basically, it is an another example of nudges.

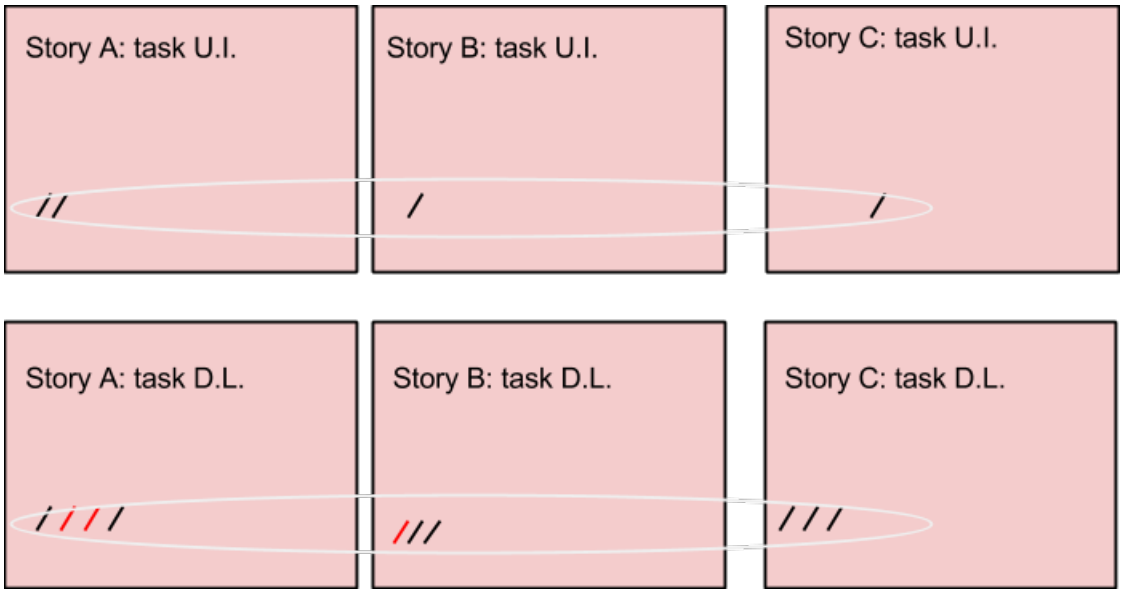
Marking at task level

A team that is working on a complete product, may have two different kind of cards: User Stories and Tasks, where the User story is a “complete feature” visible for some user of the product, and task is just one of the specific technical activity needed to complete the User Story. If the tracking is done on tasks there are other possible aggregate views. In the following example we see that the stories are A and B, and happens that the tasks are two for both the stories, and they belongs the two technical areas respectively: “User Interface” and “Domain Logic”.

The tracking with the ‘/’ at a task level is typically arranged as follows in a board:



To visualize data aggregated for task/technical area: rearrange the task cards horizontally respect to the story, and vertically respect of the task area (works well if the division in task area is quite similar for all the stories, i.e. only User Interface and Domain Logic in this case).



If we look horizontally, we see that the tasks of the D.L. area takes a little bit longer and have more issues, shown by red checkmarks, than tasks in the U.I. area.

Using the cards, we can still switch at a "narrative" level, to talk deeply about the "history of the story/task". It can be an exercise to do during a retrospective.

Possible drawbacks

You need to keep the blame at bay if you want to make sure that people actually cooperate and track problems looking at learning lessons, instead of hiding them. It does not mean that blame stories are

not valid, but more often than not the flaw is in the blame arguments[5], and in the fact that blame is focused on the past rather than on the future.

Another possible issue is the risk of "normalization": seeing too many impediments (red) checkmarks make us see as they are normal[6]: I think this is another good reason for limiting the number of stuff that we track as "impediments" (or block, or delays etc…)

Therefore I'd suggest to be careful in conveying the message, and to make sure that even small improvements actually happens and are shared, to be able to show actual short term wins, particularly if you are a change agent.

A3[7] driven process improvement may help. Just make sure retrospectives or other solution/problem solving focused meetings produce S.M.A.R.T.[8] goals.

Conclusions

The "method" borrows informally ideas based on behavioral economics, nudges, irrational behavior to a generic working flow modellete by a "kanban-like" board and cards. People interested to Prisoner Metrics can be coaches, team leads, facilitators, change agents. A first implementation of Prisoner's Metrics deployed in some teams was based on tracking time, but without any uniform specific way to track impediments.

Key activities are:

- determine the quantum time for tracking (by defaults is one day).
- determine the definition of "blocked status" (or impediments, etc…).
- determine in retrospect actions that may prevent that already known problems happens again.
- consider the idea that team problems are likely to follow the "commons pool resources" pattern, and so can be more efficiently solved by constrained self organization, rather than via top down directed rules[9].
- sometimes you may need a special facilitator called the "kanban guardian" that is responsible for tracking the checkmarks on the board.

[5] Hindsight bias, Fundamental Attribution Error

[6] crafting messages describing issues in a specific field, the protection of the environment for example, may lead to "normalize" them instead of provoking an effect of more caring. (Cialdini, R. B. (2003, August). Crafting normative messages to protect the environment. Current Directions in Psychological Science, 12(4), 105-109)

[7] http://a3thinking.com/

[8] "specific, measurable, attainable, relevant and time-bound"

[9] Eight design principles for Common Pool Resource problem. See also "Software as Tragedy of the Commons"