Brief Kanban risk management

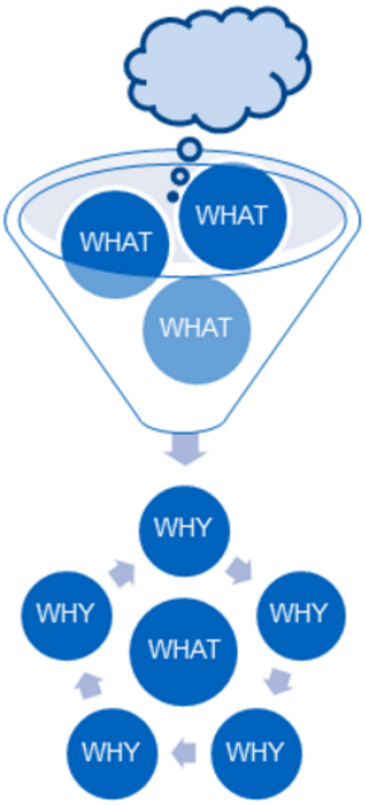
Most of certified agile masters think that a great deal of explicit risk management becomes unnecessary when a project uses an agile approach. With a short iterations, single-minded focus on working software, heavy emphasis on automated tests, and frequent customer deliveries help teams avoid the biggest risk most projects face—that of eventually delivering **nothing**. Many agile projects rejects any form of explicit risk management. I think it is better always actively manage risk in any types of methodologies.

As a definition of the risk, it is the “uncertainty that matters” and if this uncertainty happen it impacts one or more objectives in either a negative (threats) or a positive (opportunities) manner. Risk management is about identifying, addressing, and eliminating sources of risk before they become a threat to the company. There is a common agile way how to manage the risks.

Identifying the risk

Risk identification is a process of creating a list of the risks that threaten the project. It is harder that we can imagine but it is still possible. We have to think accurately

about uncertainties and its effects.



For identifying the risks, Alan Moran suggests to use the what/why approach. During each iterations you can ask a team members to brainstorm about “what” might happen and write it down. The second step will be identifying several “why’s” for each “what”.

Analyzing and prioritizing the risk

Analyzing the risk is process of assessing the likelihood and impact of each identified risk. In other words the probability of happening each risk and impact of it should be estimated. Normally it will be classified and categorized in several groups and can be based on: environmental impact, developing solutions, project budget, deadline and timelines, security, privacy and resources.

Measuring the risk

Once we have identified and classified our risks it is time to estimate them by some parameters. Normally the method of Impact and Probability is widely used. **Impact** – can be the

number of days to be loosed after having this risk and **Probability** – is the chance of having that risk in percentage. By using this variable we can create a risk matrix and figure out the level of that risk. Since the impact and probability are given in a different metrics, it is better to normalize them to have exact range of that variables. By having this we can calculate the severity level of the risk. Let's say divide the probability and impact into five levels:

Probability					
5	5	10	15	20	25
4	4	8	12	16	20
3	3	6	9	12	15
2	2	4	6	8	10
1	1	2	3	4	5
	1	2	3	4	5

Impact

Severity level and risk mitigation

After completing with risk matrix now we can determine the risk ratings (severity levels). For each level we can determine appropriate actions to mitigate the risk. Depending on what you have obtained from the matrix you can create your own severity levels and separate it let's say to low < 5, 6 < medium < 12, 15 < high < 20 and critical = 25.

> ***Example:***
> Risk: Dependent systems are incompatible during integration
> Impact: 12 days, nominal – Level 2
> Probability: 80% – Level 4
> ***Severity:*** 2 * 4 = 8.

Since 8 is in the middle of 6 and 12 this risk should be leveled as Medium. What actions are required to mitigate this

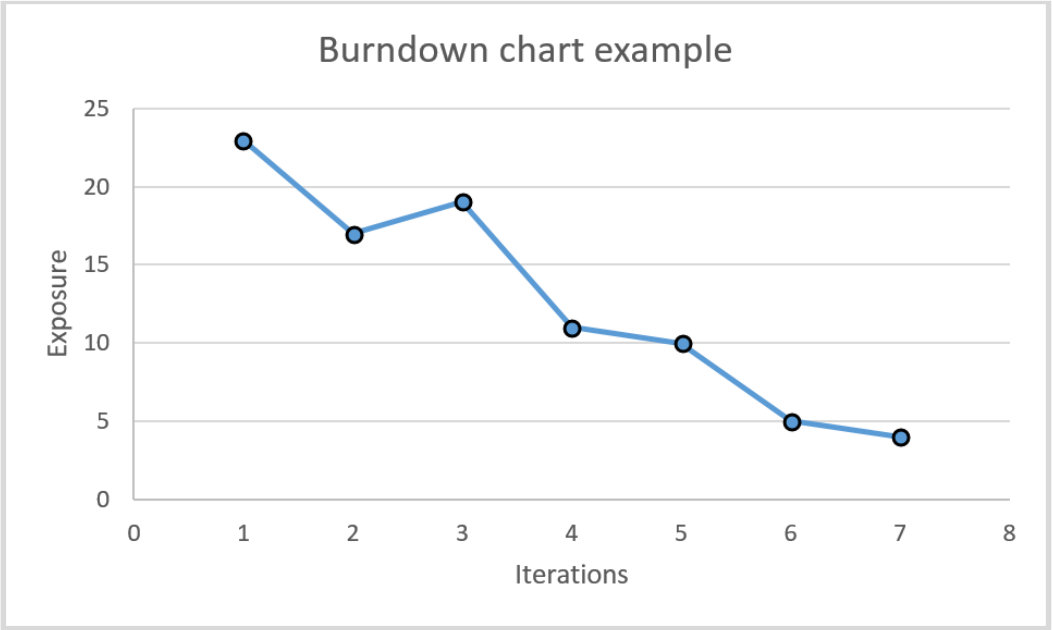
risk? You use general purpose actions from your predefined table for all the levels + some specific actions depending on this risk type, for example, search of new integration partner, research training or hiring. Tracking and monitoring all this kind of risks are called risk register.

Monitoring the risks with Burndown chart

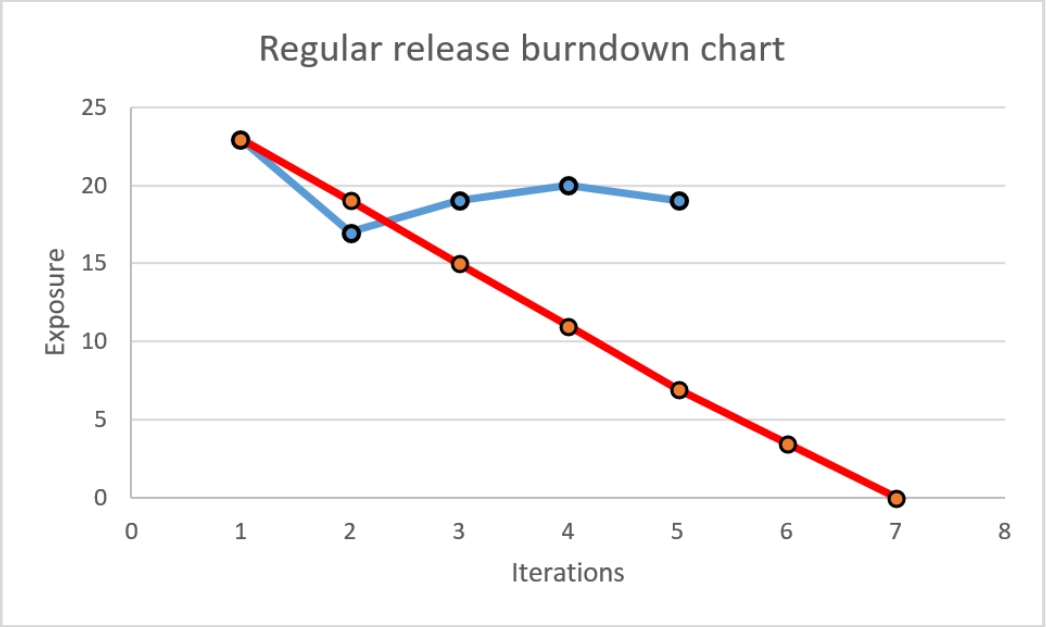
As far as the agile Kanban requires as much as possible visualize everything it is not possible to get into real risk status with traditional models. For this case an interesting technique was introduced by John Brothers in Agile Times in 2004 11. According to this method, we should still have to do something similar like traditional technique but little bit in different way. We still have gather the risk every time and identify for each of them four main attributes: detail, probability of occurrence, the amount of day will be lost if it will occur and exposure just by multiplying the probability with size of loss.

Risk register monitor has to be created in the beginning of the project. This risk register monitor should be updated and reevaluated in every iteration, since the new risk assessments can come or the probability of existing risks can change. After that, finally the risk burndown chart will be created by plotting the sum of the risk exposure values from the iterations.

According to Mice Cohn it is recommended to sum only the top ten risks even if the team has identified more. A sample risk burndown chart can look like this:



As with a regular release burndown chart, we should see a linear drop in risk over the course of the project, as shown in the red line of this next risk burndown chart.



When risk is not coming down at the appropriate rate the team and management should have to make some decisions and separate additional time in upcoming iterations in order to reduce the risk. These are some of the bare-minimum approaches that are effective in Agile. They’re simple to manage and align with the iterative process.