Lean operations

a generic, transparent input-queue framework for reducing waste and improving flow across cross-functional teams

Having introduced a static setup of an organization, we initiated talks on the dynamic, daily execution of inside the structure.

The main points of concern were the following:

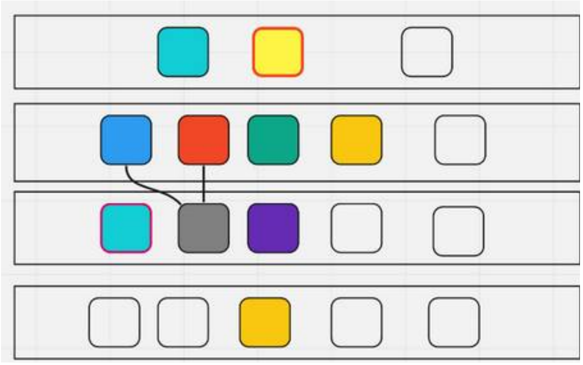
- Product Managers are involved in long, medium, and partially short term planning. They might be flooded over with different sorts of issues, questions, that are coming to their table – making the decision process slow.
- Product Owners from various development teams, may have different set of connections. Maybe another organizational unit, or a direct connection to an other team. Such set of connections, if they not defined, would mean extra, recurring time for properly maintaining them.
- A simple, automated but generically available, actionable set of metrics is needed.

Our proposed solution was built on Lean foundations, in where, we eliminate the most part of the waste, that is occuring during product development, so for easier reference in discussions we have named it as **’Lean operations’**.

The main building concept of the model is being generic - similar concepts shall be used to all teams[1], cells, layers, organizational units. These concepts are enabling constraints, giving us a foundation to build on, while maintaining team level autonomy as well.

Lets turn our attention to a particular team.

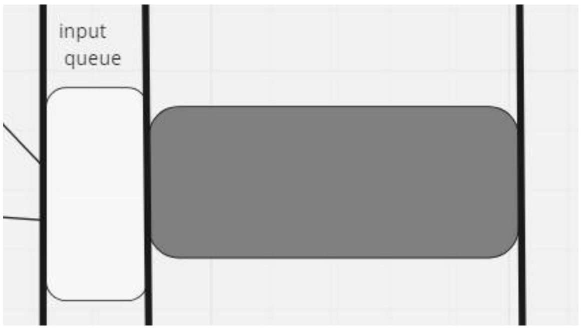
Our particular team, is <u>Team Grey</u>**.** We have identified two teams from an other unit, establishing business connections[2] with them. We need to have those business connections written down, visualized.



In order to have a closer look, we zoom in, onto the connections, and teams that are under our exploration.

Input queue

In order to establish a fully radical approach to *transparency*, we define an **input queue** to all teams.



An input queue is a designated place for all of your work items, queuing up (to be selected) for processing.

Any work item, of any source[3], of any type must be placed onto the input queue of that team.

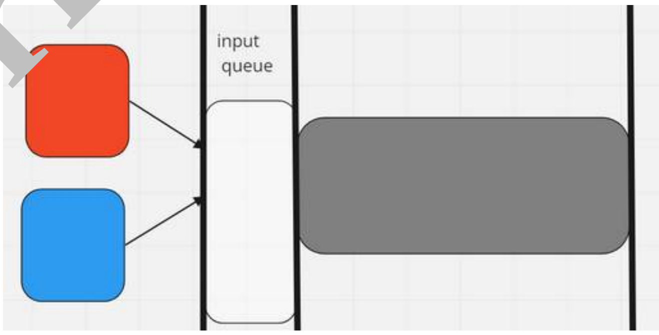
An input queue must have its rules[4] defined for entry, ordering, and pull actions. The ruleset of a particular

[1] We use ’teams’ for the entire document, but as it is being generic – lean operations are valid for any type of defined organizational unit.

[2] Team Grey is providing service to Blue and Red teams, by committing to their work items.

team’s input queue should regard the current situation of the stakeholders, Product Owner, the development team. It may vary on a large scale, from total independence of the Product Owner to a fine tuned, ratio based setup. It’s only the input queue where partners can place work items, direct calls to developers are not encouraged.

Means of limiting your work

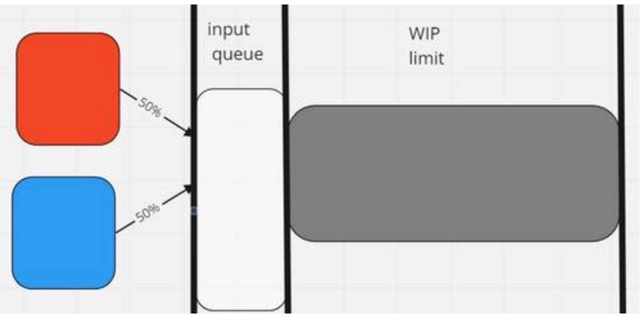


Just as in personal life, our capabilites to maintain connections are limited. Constant exploring of business partners on different development initiatives is not a direct addition to value creation. We need to be transparent on the topic of to whom we are talking to, with whom, we have **commitments of delivery**. We need to write such connections down, visualize them in an overview map of the organization. As empirical observement will tell us, such *connections are to be limited* in numbers.

This got to be a contract, where both sides of the partnership is taking on certain aspects of collaboration. E.g frequency, type of, sizes of, <insert here any properties> work items, that are being handled’[5].

You probably need to have ’virtual’ clients, other sources of work items defined as well. Examples include but not limited to guilds, legal dept., technical guidance from your cloud provider etc.

Context switching is not a direct addition to value creation. Context switch is the activity, when you are rushing to pause a current job of yours, and come to the surface, and go deep dive in an other job. Its taking up energy, time of an individual and a team as well. In Scrum we limit this by the sprint backlog, in Kanban we set and observe the **WIP limit of the team**, that is limiting the maximum workitems that a team can hold paralelly. In one way or other, *the number of workitems will be limited* – thus the overall capacity of a team[6] for a certain period of time is set.



[3] No longer saying, ’yes, but we have an other, maintainance backlog’, or we were told by the cloud provider to do this, or we have a GDPR issue, that needs to be adressed, or ’we are into our sprint, and a lot of backlog items are thrown at us, with high priority’. All above are subjects of that particular input queue.

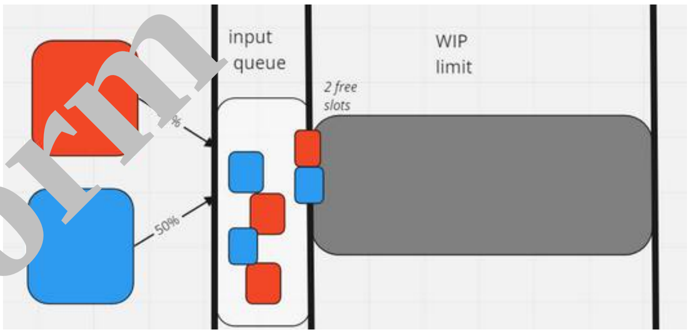
[4] such rules are defined in the team manifest of that particular team – and based on the teams’ focus, experience and maturity

[5] "In our example, Team Grey receives input from Team Red and Blue. Team Grey therefore has a commitment for delivery with both teams. This commitment can be to any usable increment of work that adds to the value required." On the other way around, Team Red and Blue knows and accepts how and what workitems can be placed onto Team Grey’s input queue.

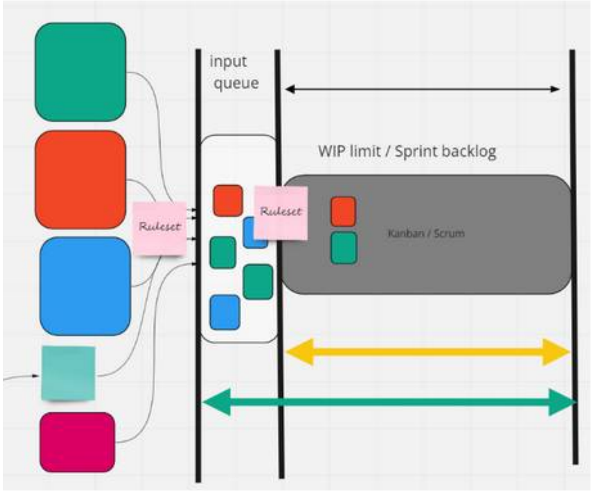
[6] How we define ’in progress’ of a team, what can be regarded as an item of work, how to set WIP limit, how to modify it, is a subject of team level implementation of lean operations modell.

In order to ease out the stress, and still making predictable commitments, a teams' upstream clients needs **to have an agreement of sharing ratios**[7] of the teams' resources.

Work items will be flowing in into our input queue. We are happy to see that, but an other responsibility is rising. Which of them **should be pulled by the team next**? Not an easy job, can be time consuming, can hurt certain interests. There are several schools of thoughts on this topic, but as always we advise teams to start simple.



Suppose, that Team Grey has 10 items capability for 10 days. In Kanban, just before you open the gate between your input queue, and processing area, check your administration – which client of yours has used to what level, their ratio[8]. Should they have a 50%-50%, and 2 work items were pulled from each client, and you have 2 slots free in your system just ask them, which item from your subset is the most important[9]? Pull those items, update your admin, and close down the gate[10]. In Scrum, setup your sprint backlog accordingly – with the accountability of your Product Owner.



Enlarging the collaboration

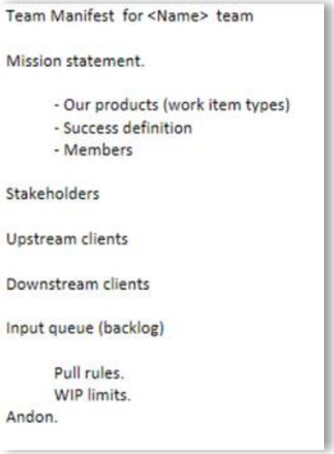
Later on, Team Grey is doing fine in the Lean operations modell. **New customers** are signalling their wish to join. We need to have a talk. As the capability of digesting work items of Team Grey hasn't grown, the previously mentioned ratios, ruleset on sharing, needs to have a review. So, with Team Green on board, we initiate a review session, and update the contracts between the teams.

[7] Special, 'virtual' clients like Legal dept, or a Guild should have special lines in the agreement, not ratios. If the team is a mature one, ratios are not needed.
[8] When in Kanban, rationing for a fix term, can have different cadences than sprints.
[9] This is just an example. Other rulesets can be set.
[10] Sure, the frequency of opening your gate is published. So all partners know the next date.

Team Manifests

Everything that’s been discussed above, forms a Team Manifest. This document is a contractual bond between the partners. A **team manifest** is valid only, if all the partners has acknowledged, and agreed upon its contents.

With simplifying how to maintain business connections, we optimize interactions by PMs, enabling them to focus more on strategic planning. With radical transparency, and simplified processing on the input queue, we help POs to focus more on value creation.



Team Manifest for <Name> team

Mission statement.

- Our products (work item types)
- Success definition
- Members

Stakeholders

Upstream clients

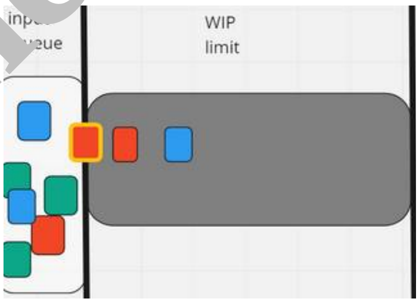
Downstream clients

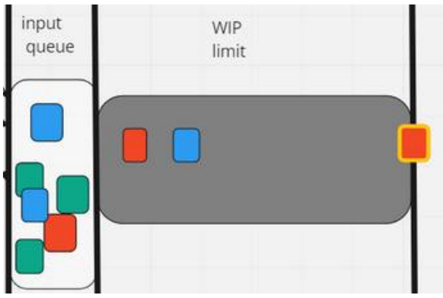
Input queue (backlog)

Pull rules.
WIP limits.

Andon.

Actionable metrics





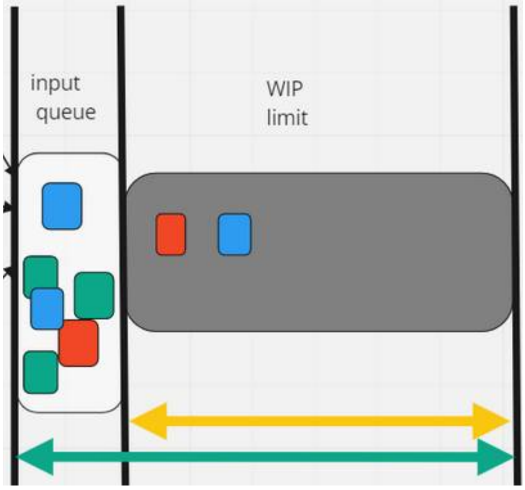
With the flow directed onto an input queue and the teams own processing area, with every work item the team needs to deal with, setting up easy and powerful metrics become a reality. We automatically time stamp a work item, when it is entering the processing area of a team. We do the timestamping of the work item, again once it leaves the processing area.

The sum of those timestamps is called **cycle time**.

We do that for all items for that team.

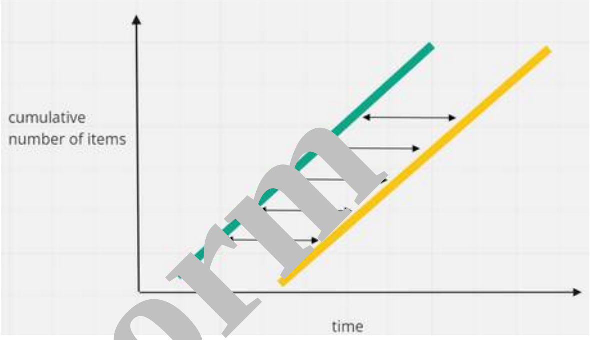
What we got back is the average cycle time (the yellow arrow).

With the same approach, lead time is the time needed for an item, from entering the input queue until leaving the teams’ processing area. (the green arrow).

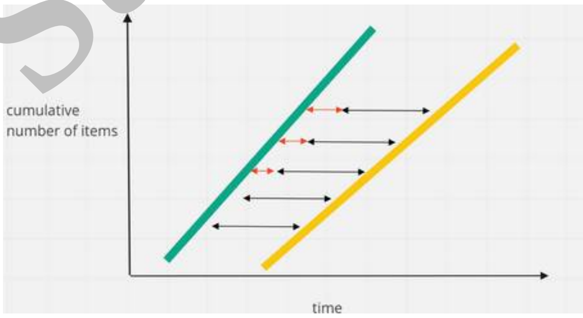


On the longer run, with a stable, mature team, stable business and technical domain, average cycle time can be regarded as a constant.

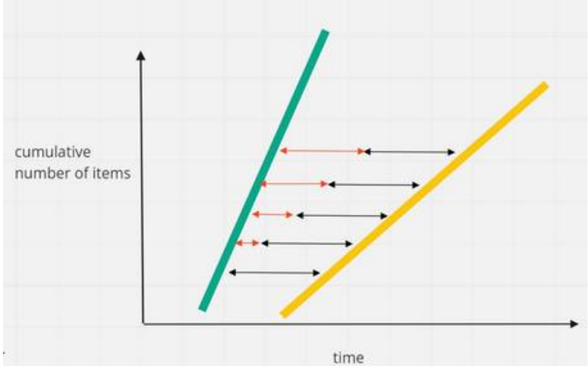
In a situation where the average ratio of new work items arriving is the same[11], the particular team is sustainable. A green light on a traffic light.



If the average lead time begin to drift away, just like pictured, your team may be getting more items, that can cover. Its a sign of caution. Setting up new business connections (allowing more workitems flowing in) are not recommended.



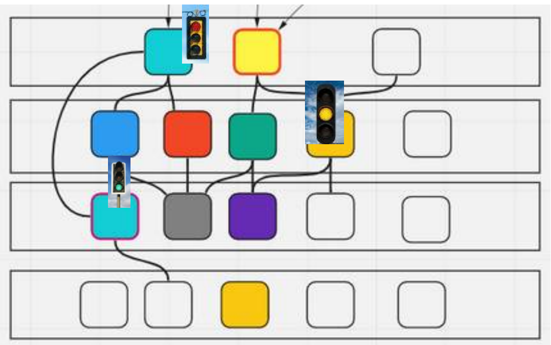
If the distance between average lead time and average cycle time has grown significantly, the team (would be overdriven, if no WIP limit, or proper sprint planning) is a blocker in the business connections – causing an overall slowness, unpredicatble deliveries.



Immediate actions are required to mitigate the situation.

With data for each team, powered by the team manifest based connections, we can setup a board where it can all be seen for the entire development organization.

Should all teams, in all units are connected in using the same enabling constraints, interconnected product deliveries are to become predictable.



[11] on the longer run. This is a game of averages and trends.