WHITE PAPER

# MONITORING KANBAN EXECUTION

Infosys® Navigate your next

## Monitoring Kanban with CFD (Cumulative Flow Diagram)

In Kanban implementation, it is critical to understand the dynamics of flow, progress bottlenecks and non-value-adding tasks (wastage) that inflate the lead time. Akin to acupuncture, where applying pressure on healing points stimulates the natural abilities to effectively relieve ailments, understanding the areas to be focused upon would make the workflow optimized and predictable which can stimulate Kanban outcome in a big way. CFD charts help in analyzing the health of Kanban execution, infer the movement of work items in different stages, and understand how the work is progressing, or being stacked up in various stages. It also allows to calculate the average lead time and WIP which helps in throughput assessment and guidance for team composition. The chart will be good to infer once it has few weeks of data underneath.

Fundamentally, CFD is a visually stacked area chart that plots a cumulative number of work items on the Y-axis and period in days on the X-axis. CFD is a snapshot (point-in-time view) of the cumulative work in backlog and WIP over a time interval. While sprint burndown charts in Scrum provides the understanding of total work remaining over the sprint duration, CFD provides the visibility of flow and helps to analyze the roadblocks across all stages in a single view without restricted timeframe. Hence, sometimes CFD is used in Scrum in addition to understand bottlenecks stages.

CFD helps to understand how effectively the team is pulling work. Using CFD, we can measure lead time, WIP, and done [completed] work at any point of time. Since different teams can have many variations in work items (e.g., user stories, bugs, task, change requests), a tip from us is to segregate and have CFD represent single work item for the reliability of measures and to erroneous results. The CFD below is plotted for the example scenario of paper 2. Colors represent the different workflow stages (columns on a board) of the typical development lifecycle.

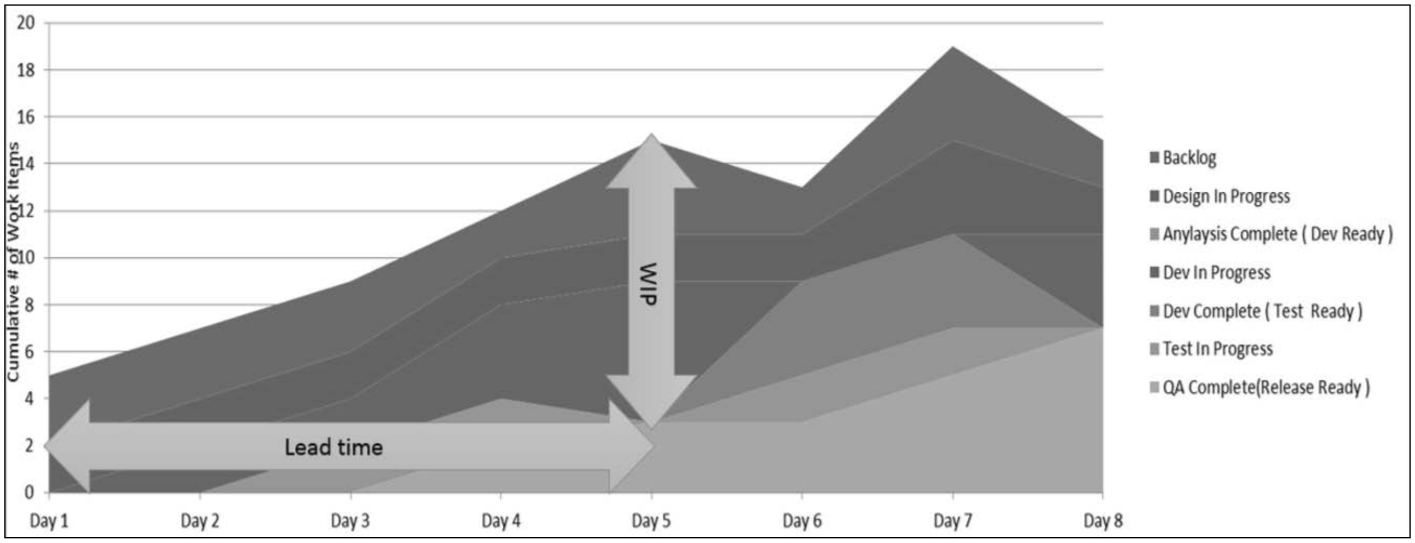


Figure 1: Cumulative flow diagram

## Cumulative flow diagram insights

- The backlog area shows the work items waiting to be picked up. It has been constantly increasing till day 5, just to fall a little before going up again from which we can infer that the scope has been changing continuously.
- The width between backlog and release ready [done stage] is the WIP. On day 5, 15 tasks are in backlog and 3 are release ready. So, the WIP is 12 (15-3).
- The wider WIP area indicates team composition across stages, which is not able to produce faster throughput
- The larger gap between stages signals:
  a. Slower transition of work items or increased bottlenecks.
  b. Either the previous stage is blocked (due to person-dependent task or environment issue) or the next stage is short of capacity and unable to pull work (task with niche skills).
  c. Increased WIP and lead time
  d. Waste can be due to high WIP, wait / blocked times or concentration of large tasks near the stages
- Consistently growing release ready area would indicate better adaption and balanced demand vs. supply whereas the steep rise or flatness means the team is working on a large chunk or throughput is not meeting expectations.

Thus, CFD can be a very effective tool to understand how long a work item is waiting in a stage before pulled in by next stage where the work items form a queue, and what is causing the avoidable delay in reaching to done stage. This can lead the team to resolve underlying causes e.g. lack of team communication / collaboration, capacity fine-tuning, faster turnaround on blockers, etc.

Apart from CFD, teams can plot control charts on understanding upper and lower limits and what type of work stay as outliers. This will help the team to focus on solutions for outliers to reduce variation.

## Kanban metrics

Fundamentally, CFD is a visually stacked area chart that plo Contrary to mistaken belief, Kanban does have metrics that teams can measure to understand the progress of workflow and wastages along with team productivity. Knowing the expected WIP and throughput, one can even calculate the ideal team composition for the successful delivery. The table below details some of the key metrics and calculations.

| **Metrics** | **Terms Definitions** |
|---|---|
| **Lead time (LT) = DD – RQD [Elapsed Time]** | RQD = Ready Queue Entry Date [Ready for Development] |
| **Throughput (TP) = WIP / average LT** | DSD = Development Start Date [Task/story picked for Development] |
| **Wastage** or idle time = DSD – RQD** | DD = Done Date [Task/Story moved to last stage i.e. Done or released] |
| | WIP = Work in progress at a particular time |

** In systems development life cycle (SDLC), wastages can be present at each stage. The table above is an example of backlog wastage i.e. time elapsed to pick the work due to capacity crunch or agreed WIP limit applied on Dev start stage.

There are often mix-ups in lead time and cycle time ending up with incorrect uses. While the lead time is elapsed time from the moment request is made until its delivered, cycle time is time duration between successive deliveries i.e. unit of work per period of time. Cycle time is used to measure throughput.

Let's take an example to illustrate the calculation of metrics. Here, there are four stories at a given point in backlog, and each was pulled for development incrementally, leading few of these being in an idle state for some period. If the team records the work start and end dates and the WIP, the productivity can be derived easily.

| Requirement | Days in Ready Queue ( Wastage) | Days taken for 'Done' [CT] | Completion Rate [ACT] | LT |
|---|---|---|---|---|
| Story 1 | 0 | 3 | 2 | 3 |
| Story 2 | 1 | 5 | | 6 |
| Story 3 | 2 | 8 | 3 | 10 |
| Story 4 | 3 | 10 | 2 | 13 |
| | | | | |
| Average | | | 2.33333333 | 8 |
| Throughput [TP] | | 0.5 | [Story/Day] | =~1/ACT |
| WIP | 4 | | | |

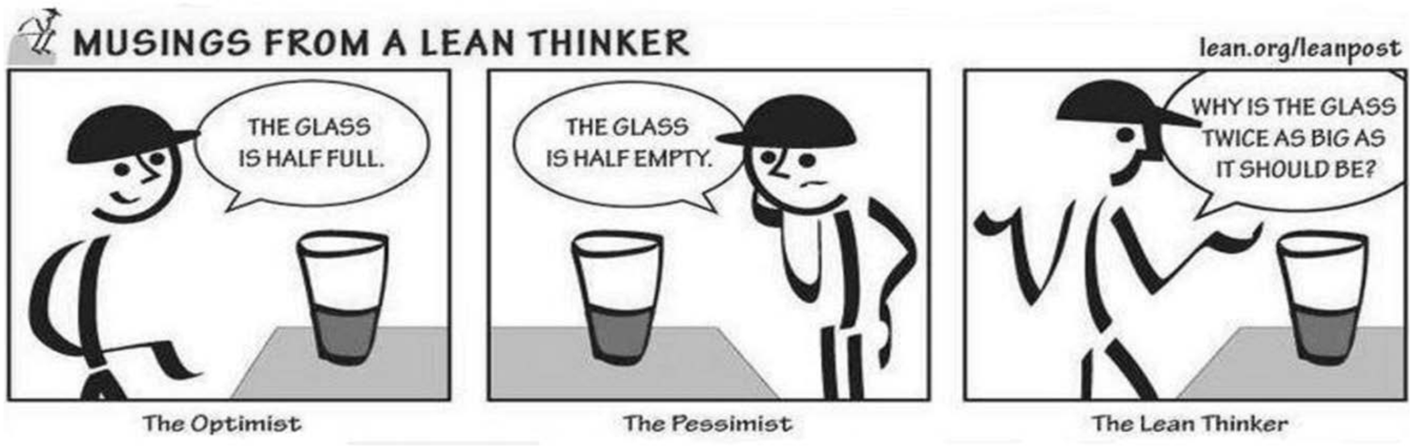
The above illustration also corroborates that LT, Throughput, and WIP are tied to each other and can also be written as Average Lead Time = Average WIP / Average Throughput

## Flow efficiency metric

What percentage of lead time does a work item actually take end-to-end on an average? That is, a work item may have spanned five days to get from the backlog to done stage, but the actual effort spent might be a lot less, the difference being waste. By comparing lead time against the touch time (aka assigned time), we get 'waste indicator' which is another useful metric.

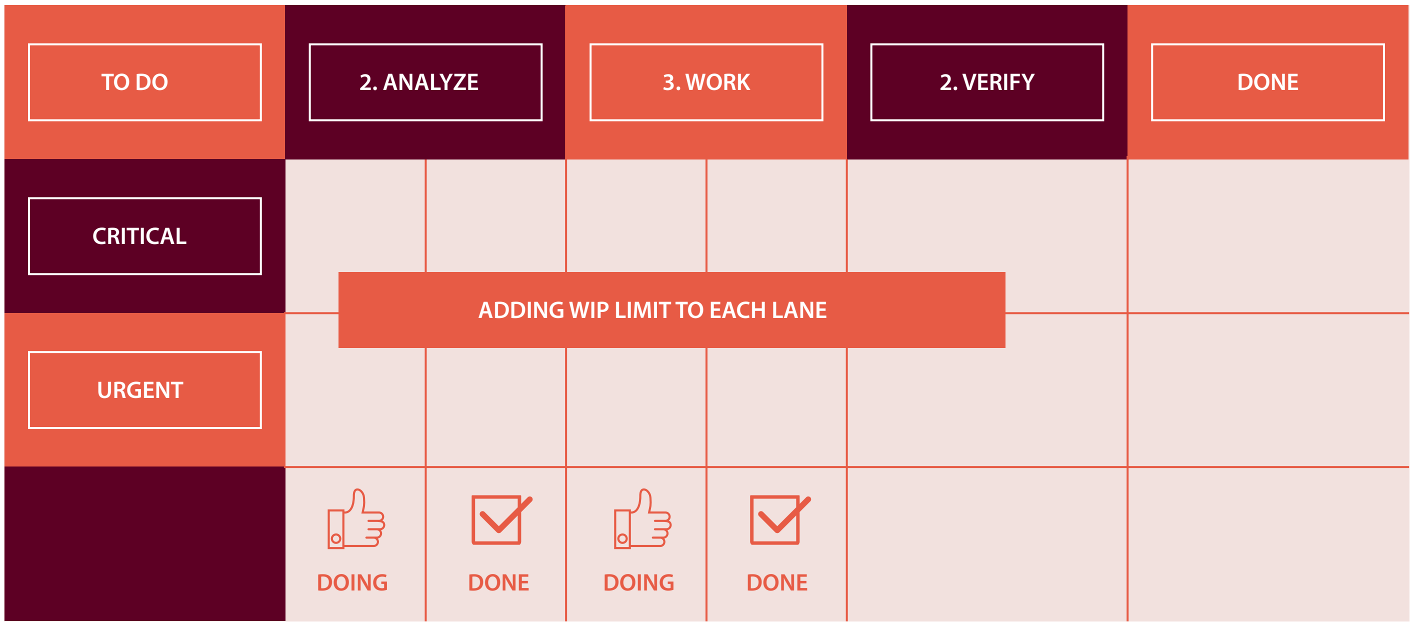
Flow Efficiency = Touch Time / Lead Time

The goal of Kanban team is to keep this efficiency ratio high. This implies that the team should focus on finishing old work before starting new and reducing the time a work item stays in backlog. To keep the ratio near to one, eliminate non-value-added tasks, processes, and wait time between stages that are increasing lead time. Optimizing the backlog management and balancing team size based on demand (seasonal spike, off-peak hours, regular periods) can also be useful here.



## Swim lanes and FAQs

Swim lanes help to organize work items in horizontal lanes on the board e.g. expedite lane to segregate blocker (P0) tickets, based on service level agreement (SLA) or severity or grouping based on work that is business as usual (BAU), on hold, removed, etc. A work item placed in an expedited lane will demand higher attention over work items placed on other lanes in the same stage. So, the team can agree on the WIP limit in this lane and suspend other work to fast-track priority work as an explicit policy. Below is an example snapshot:



The focus of Kanban daily standup is on the 'flow' and checking on the WIP limits set for queues and swim lanes. It is more than the standard three-questions-daily Scrum format. Since the work is not bound by iterations, the estimation of work items and tasks happen just in time which reduces the estimation and planning overhead. Focused synergy on flow, stock movement, and the WIP limits leads to problem-solving and fosters collaboration.

The demand or throughput could influence work completion rate, team composition, and the WIP limits as well. Adopting extreme programming practices like continuous integration (CI), test automation, pair programming does certainly help in reducing lead time by taking off execution wastages. It would be useful to understand the frequently encountered challenges and approaches to manage them.

### Fragile WIP limits

It can be difficult to set the optimum WIP at the start. So, when the flow is stuck or Kanban is not producing the intended throughput, the first thing that comes under pressure is the WIP limit. Often, teams increase the WIP and pull a new work item. The right approach is to reduce the bottlenecks, improve turnaround time (TAR), collaborate, and try pairing. The bottom-line is to refrain from changing the WIP limit on a daily basis. Over the time, as the team gains knowledge on the type of work items, arrival rate, the team's skill set and expected SLA, and such, the WIP can be controlled for an optimum flow rate.

### Work value is realized when 'live' in the production, not a 'done' state on Kanban board

In large enterprises, often the development team takes a work item only till deployment i.e. a shippable is considered 'done'. User acceptance testing (UAT) and production deployment is taken care by separate Ops teams. To ensure that done items move to live (business value), they leverage tenets of DevOps convergence to facilitate quick deployment and automation in releases.

### Ad hoc tasks pull over 'in progress' work

Kanban is often adopted in situations where the inflow is dynamic; hence, ad hoc changes should not come as a surprise. The approach we recommend is to plan the capacity for such ad hoc work by provisioning separate lanes.

### Processing bugs that come up during testing stage

Bugs should be managed similar to any other work item i.e. should be added in backlog and then prioritized based on the severity, impact on workflow, and completion of the work item. They can then be processed according to Kanban WIP limits. High severity bugs can be put across on fast track lane for priority processing.

## Conclusion

Kanban has been evolved in last few years as a method of choice for Agile adoption in the setups where the inflow of work is continuously fluctuating, and outflow is expected to be consistent with increasing productivity. Kanban also does away with or reduces ceremonial wastages of Sprint Planning, Sprint Reviews by putting just-in-time planning and review approach. Hence, Kanban is well suited for production support / ticket-based projects and where businesses are obliged to balance demand productivity with supply (capacity) balance most effectively to generate greater return on investment (ROI). Kanban helps organizations achieve predictability based on steady state throughput and average work expected to be processed. The flow should be kept moving by tuning the WIP based on average lead time, expected throughput, and team composition constraints.

Kanban is also the preferred method of Scaled Agile [SAFe] teams to track the progress at portfolio, Value Stream, and Program levels. Here, the work is pulled as the capacity becomes available i.e. WIP limit drops. This helps in the efficient movement of the work item and provides better insights into dependency and integration bottlenecks across the larger solution / value chain.

The DIY Toolkit below is specially designed for teams that want to discover, adopt or adapt Kanban to improve the workflow and deliver improved results. This takeaway will help you hit the ground running.

1. Visualize work and define states through workflows [Kanban Paper 1]
2. Create a Kanban space / board in digital Kanban tools (like Jira, Leankit Kanban, SwiftKanban, Rally) in the most visible part of project and then socialize
3. Settle collectively on the WIP limits for each state [Kanban Paper 2 can help to define and optimize the WIP]. The first and last state would usually not carry the WIP limit. You can, of course, adapt the WIP limit to improve the flow as you progress and get better insights
4. Brainstorm, establish and publish policies on managing the WIP breaches when the lane is stuck, managing work items, and such.
5. Measure the progress (CFD, lead time, throughput, lane transition wastage)
6. Retrospect and adopt Kaizen to improve continually
7. Re-emphasis on how 'Stop starting, start finishing' is the most effective mechanism for value creation.

## Way ahead

Kanban is not a silver bullet for support or operations programs, rather it is equally viable to adopt in development projects and effective also for funneling requirements. We strongly suggest move to ScrumBan model to gain the best of both Scrum and Kanban. Kanban integrated with DevOps principles of continuous delivery, seamless coordination between business, development and operations shall make tangible outcomes surface.

| | |
|---|---|
| Release Frequency | How often do we release code into production |
| Incidents | How many outages does the service experience |
| Service Availability | How available is the service |
| Mean time to Recover | How quickly do we recover from a service failure |
| Quality | How many defects are found during development |
| Lead time to Deploy | How long is the development cycle |

### Related Reading:

Assessing Kanban fitment in the fluid and Fast-paced world of software development
Kanban execution: Optimizing work-in-progress (WIP) towards achieving a shorter lead time and better flow rate

## Authors

**Vikram Abrol,** Principal Consultant, has been in IT for 16+ years and holds project management professional (PMP) as well as certified scrummaster (CSM) certifications. He is also affiliated with a leadership program from IIM Calcutta. He has been a Process Consultant and Quality Manager with seven years of experience at client sites in the US. He has been an Agile Coach with Agile transformations for various clients in the UK, US, and Europe. He actively blogs his experiences in Agile on LinkedIn (Blogs on Agile).

**Ketan Shah** is a Principal Consultant and an Agile Coach. He holds SAFe Agilist (from Scaled Agile); CSM, certified scrum product owner (CSPO) from Scrum Alliance and six sigma green belt certifications. He has over 17 years of IT experience including more than nine years in E2E Agile execution (specialized in Scaling Scrum Of Scrum, Kanban and XP Practices) from inception to implementation for several large enterprise engagements in various roles like Agile Coach, Scrum Master / Mentor, Delivery Manager, Agile consultant. He likes to spread Agile awareness via interactive training, mailers, community forum discussions, Agile center of excellence (COE), etc. He has contributed thought papers in Infosys Opinions Journal, a portal on Global Agile and also wrote papers for reputed international agile publications like DZone on DevOps.

The first two papers of the Kanban trilogy involved assessing Kanban fitment and focus on optimizing WIP (Work in progress) to influence the lead time and flow cadence. The last part of this series is to sort out the execution riddle and take an in-depth look at tracking measures, metrics, building an understanding of Kanban charts like Cumulative Flow Diagram (CFD) and do it yourself (DIY) Kanban kit. It is highly recommended to go through the earlier papers to build a solid understanding.

Infosys®
Navigate your next

For more information, contact askus@infosys.com

Infosys.com | NYSE: INFY

Stay Connected SlideShare