

Integrated work management

An approach to delivering services and solutions



About this white paper

This white paper presents a new concept of managing work streams from both the perspectives of Development and Support as an Integrated Work Management Model. The model presented herein is a concept that provides a flexibility lever to adopt

the model to varying environments. This model is aligned to the agile philosophy, combining the best of Scrum and Kanban models, with an underlying emphasis on Continuous Integration/Continuous Deployment (CI/CD) or DevOps environment.

Table of contents

1. Introduction	4
2. Limitations of existing models	5
2.1 Agility in application development	
2.2 Agility in application support	
2.3 Change management	
2.4 Development and support teams – Separate entities or one team?	
3. Scrumban-based integrated work management model	8
3.1 Work stream considered for the model	
3.2 Elements	
Work items	10
Team	10
Product backlog	10
Service backlog	10
Release backlog	11
Re-prioritization	11
Service portfolio Kanban board	12
4. The Model on DevOps	13
5. Conclusion	14

Introduction

In today's world of constantly evolving technologies, companies can offer a larger set of offerings and innovative solutions to customer needs. This stiff competition has intensified the pressure on organizations to resort to measures which can provide a quick turnaround on time-to-market. As business enablers, today's IT services are under immense pressure to come up with a paradigm change.

To thrive under intense competition, several changes need to be deployed in not only business-critical applications, but in any IT portfolio. The rapidly changing dynamics of application portfolios have increased the challenges in maintaining and supporting them. The velocity of changes to production environment leads to a volatile production environment. High volatility is inherently quite disruptive and increases the number of failures in a production environment.

One way to reduce these failures is to opt for more resilient and high-capacity infrastructure. Infrastructure is a huge investment and businesses would like to spend on the IT ecosystem only if they perceive a high Return on Investment (ROI) immediately. These factors cause a delay in taking the right decisions to upgrade the infrastructure.

In order to continue running the business, an IT sponsor keeps supplementing applications over the existing infrastructure, till an in-depth analysis indicates a dire need to upgrade it.

With the advancement in cloud computing, many organizations are also offloading a lot of IT components (applications and data) from their data centers to the cloud environment. However, the application density on the existing infrastructure is increasing continuously, thus making it more susceptible to failures.

In the aforesaid environment, IT owners start looking for avenues to optimize the delivery organization. An emerging trend is the need for a single team for a set of applications or a portfolio that provides enhancements, maintenance and application support. This team is responsible for both change and event management, thus diffusing the boundaries between development and support teams. As a traditional approach, both the development and support worlds are different from each other. No wonder then, evolving a single team to provide both functions is a big disruption.

Limitations of existing models

2.1. Agility in application development

User preferences, business models, technological adoption and its landscape have always evolved and continue to grow at an unprecedented rate. A quicker time-to-market rollout with an early ROI is the dire need of today's times. This has led to the evolution and adoption of agile-based methodologies. Various agile aligned delivery models have evolved, with Scrum being the most dominant of all. However, these models mostly focus on change delivery and not on managing events.

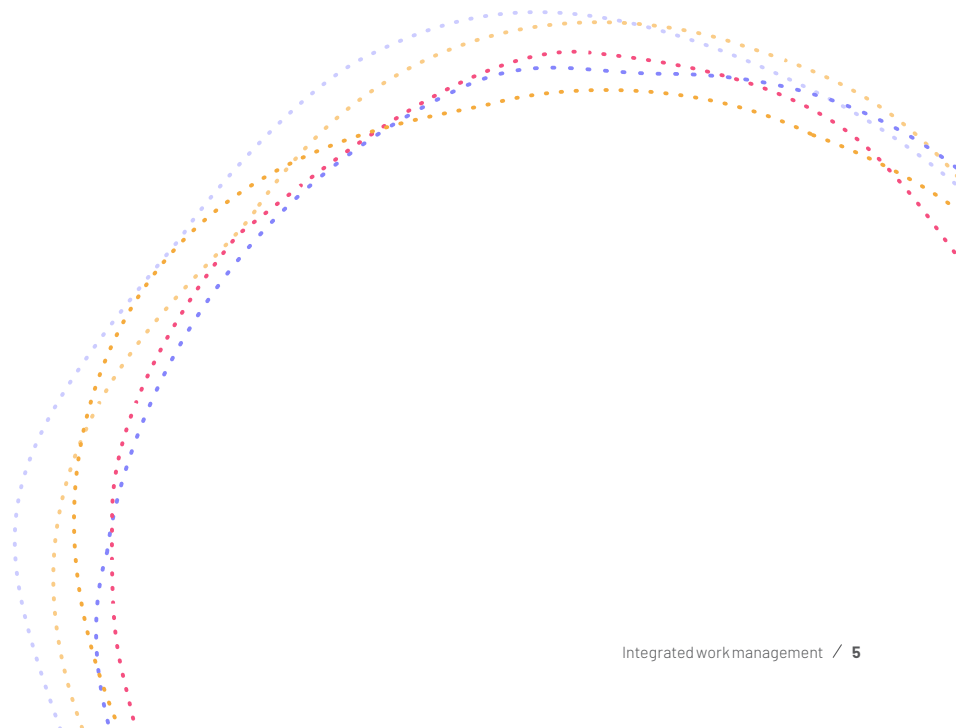
2.2. Agility in application support

Most of the application support models are based on a sequential progression of tasks. Tasks are created as an outcome of a reported event. But, they are also time-bound and are created as per their assigned priority. There are not many agile-based models which can be adopted into the Support ecosystem. However, Kanban is best-suited for all these, although it is not restricted or customized for only the Support ecosystem.

Kanban is a visualization tool and is more effective for managing a continuous flow in an agile environment. Here, work is split into smaller tasks and is displayed on a visual board, traditionally white boards. The various stages are displayed as columns on the board, representing where each item is in the workflow.

Each stage has a limited queue length and is constrained by the availability of resources to perform the respective stage work item. Hence, for better planning, each column is marked with its Work in Progress (WIP) queue depth. We should aim for process optimization to reduce the minimum time to resolve a work item. In the Kanban model, the sequence of work items can be changed at any time, which helps to manage a continuous flow.

Kanban boards are primarily followed in the manufacturing domain, though they have also been effectively employed in many IT support environments. We have successfully employed Kanban board in many support engagements. A sample Kanban board in a support engagement is shown below:



Service Backlog	Requirement Assessment	Solution Design	Development	Validate	Deploy
Incident # IN101 Requirement Description	Incident # IN101 Service Lead	Incident # IN99 Service Lead	Incident # IN97 Task 1 Support Engineer 1	Incident # IN96 Quality Analyst 1	Incident # IN93 Ready to be deployed in Minor Release
Incident # IN102 Requirement Description	Problem ticket # PR218 Service Lead	Incident # IN98 Senior Support Engineer	Incident # IN97 Task 2 Support Engineer 2	Incident # IN95 Quality Analyst 1	Incident # IN92 Ready to be deployed in Minor Release
Incident # IN103 Requirement Description	Problem ticket # PR219 Business Analyst	Incident # IN97 Senior Support Engineer	Problem ticket # PR217 Task 1 Support Engineer 2	Incident # IN94 Quality Analyst 1	Incident # IN91 Ready to be deployed in Minor Release
Problem ticket # PR442 Requirement Description		Problem ticket # PR217 Service Lead	Problem ticket # PR215 Task 2 Service Lead	Problem ticket # PR200 Quality Analyst 1	Problem ticket # PR118 Ready to be deployed in Minor Release/Major Release
Problem ticket # PR221 Requirement Description		Problem ticket # PR216 Service Lead	Problem ticket # PR214 Task 3 Support Engineer 1		
			Problem ticket # PR214 Task 3 Support Engineer 1		

To be undertaken

Ongoing

Completed

Figure 1: KANBAN Board of an AMS Engagement.

Agility is best supported if the underlying IT environment is automated to reduce manual interventions as much as possible. Continuous Integration (CI) and Continuous Deployment (CD) concepts evolved to create the DevOps

environment with synergies between the development and operations teams. The automated environment helps to manage higher workload and becomes the desired IT structure.

2.3. Change management

Any change in the status quo of the existing IT ecosystem must be managed to avoid any disruptions, such as:

- Introduction of new features.
- Change in application contour or its components
- Change in business process flow
- Introduction of new technology solutions

Every organization must control changes by pruning them from the conception stage itself. This ensures that money is rationally invested with high ROI and less disruptions to the existing ecosystem.

A change can be of two types:

- Normal change passes through multiple quality gates before being approved for further movement, eventually to production.
- Standard change is pre-approved at various stages and unlike a normal change, it is exempted from various quality gates.

2.4. Development and support teams – separate entities or one team?

In most organizations, application development is pivotal to “Change the Business”(CTB) while application maintenance and support come under “Run the Business”(RTB). The two models are managed by separate teams with their own budgets.

- CTB teams thrive on new technology incubation and solutions, based on contemporary technology stack. They are usually unaware of the IT infrastructure, various servers and system configurations.
- RTB teams primarily work on legacy applications and technology. They are usually unaware of contemporary architectural guidelines or any new age technology solutions.

There is always a critical time where the application ownership is transitioned from the CTB group to the RTB group.

A successful transition from development to support is essential for business continuity, which is an important factor for business stakeholders and also for customer satisfaction.

Both teams, when combined as one, would be the best-case scenario for business stakeholders. Around 80% of the budget is spent on business continuity while the remaining is allocated to new solutions and services. A common team that provides both application development and maintenance and support would provide higher ROIs, and would decrease the total cost of ownership over time.

Scrumban-based integrated work management model

We have developed a model which combines the best of Scrum and Kanban principles. This model retains its applicability for organizations which have (or are moving towards having) a common application support team, with responsibilities of both Run the Business and Change the Business.

Scrum is all about prioritization in advance while Kanban is all about maintaining a continuous flow. We have developed a Scrum-Ban model which is all about continuous prioritization of tasks.

Work demand is a continuous flow, with work orders initiated by business stakeholders. They can either be a change or a service request. These are known work orders and can therefore be planned. However, many uncertainties (incidents) also need to be accommodated, . Systems do face failures at times and are then reported as incidents. Support stakeholders identify repetitive failures and can raise problem tickets as well. Team can also initiate a problem ticket. Change requests, Service requests, Incidents, Problem tickets etc. create a varying demand.

In an aggressive market space like IT products, it is normal for organizations to deploy changes to the production environment quite frequently – often in weekly or even daily releases. However, the scenario is somewhat sluggish in the IT services space, where changes are promoted to a staging or user acceptance environment through sprints while being promoted to production as a single release. Releases are planned as either a major release or a minor release. Major releases deploy changes where delta functionalities are substantially large enough to cause disruption. Disruptions can either be in the IT ecosystem, process flow, data flow or even in user behaviors. Hence, major releases are few as compared to minor releases, which are generally once in each quarter. On the other hand, to match the business demand of quick turnaround, incremental changes are deployed as minor releases. Minor releases are planned either once a month, once a fortnight or

even on a weekly basis, depending on the aggressiveness of the demand. Thus, release planning plays a crucial role in addressing agility as well as return on investments.

3.1. Work stream considered for the model

For any application, work items can be initiated as planned activities or events:

- **Planned items:**
 - Change or service requests initiated by business stakeholders
- **Unplanned events:**
 - System failures reported as incidents by customers
 - Service requests reported by customers
 - Problem tickets raised by Support team after analyzing a similar pattern of customer issues
 - Problem tickets raised by the Engineering team for any functional issues

Usually, application support is provided at various levels, L1, L2, or L3. Our model is applicable for managing level 3 (L3) support and application development and maintenance, requiring similar skills.

At the application support level, incidents are classified into four classes. These levels are a product of severity and urgency:

- Priority-1(P1)
- Priority-2(P2)
- Priority-3(P3)
- Priority-4(P4)

P1 and P2 incidents require immediate attention, else may lead to a system catastrophe. The frequency of these incidents is lesser as compared to P3 and P4 incidents. The P3 and P4 incident fixes can be queued with other changes planned in the forthcoming major or minor releases.

Release planning plays a critical role in addressing agility and ROIs. The releases are categorized as:

- Major releases include the changes where the delta functionalities are large enough to impact:
 - IT ecosystem

- Process flow
- Data flow
- User behavior

These releases are usually planned on a quarter basis.

- Minor releases include minor incremental fixes and are usually planned to meet the business demand of quick turnaround. These releases can happen once in a month, a fortnight or a week.

3.2. Elements

The model combines the best of both Scrum and Kanban principles. Scrum is about prioritization beforehand while Kanban is about continuous flow. This model is Scrumban, which is about continuous prioritization of tasks. It further

describes how to manage development and support work streams together by employing a common team. To understand the model, let us study its elements.

Elements



Work Items



Teams



Product Backlog



Service Backlog



Release Backlog



Re-prioritization



Service portfolio
Kanban board

Work items

Work items represent the work that the team is expected to complete. In this model, it is a combination of both the development and support work items:

- Application enhancements
- Large-sized normal change request (normally executed as a project)
- Small-sized normal change request
- Standard change requests
- Priority-3 incidents at L3 support
- Priority-4 incidents at L4 support
- Problem tickets
- Service requests

Each work item can be mapped to a single task or broken down into multiple tasks. In consultation with stakeholders, the Work Manager prioritizes the tasks as per resource availability. If the resources are not available, the priority of an item is either upgraded or downgraded. Incidents are time-bound and hence, their sequence in the processing order is changed based on the time left to meet the SLA.

Team

The model strongly supports a one team concept, comprising cross-functional experts that includes technical architects, solution architects, business analyst, software engineers, system programmer, product specialist, data experts, system administrators, database administrators, quality analyst etc. This is a uniform team that manages a portfolio of applications, governed under a similar structure and managed through an integrated Project Management Office (PMO) set up. One-team concept is also instrumental in achieving success in a global, cross geography located teams.

Product backlog

A large-sized normal change request is generally managed as an independent project. We propose to manage not only large projects but also consider the criticality of the functionality introduced through the change, either as an incremental change or a new change. We follow scrum principles here and suggest developing product backlog and design sprints plan. The overall requirements are estimated as story points. Work items are prioritized at each sprint planning and retrospective planning. Some examples of such projects are:

- Application enhancements
- Re-engineering
- Development of new functionalities

Service backlog

A service backlog is built by accumulating incidents in the queue to be processed. All incidents such as small-sized change requests, standard change requests, incidents, problems and service requests are added to the service backlog. All the bigger events are not added to the service backlog directly. Instead, each event is broken down into individual tasks that are big enough to be completed by a team member. The reported incidents are first analyzed by the support team and a solution is subsequently proposed. The solution is subject to implementation only after approval from service owners. As soon as the P3 and P4 incidents are raised, they are added to the service backlog. As an exception, some P2 incidents may also be added to the service backlog. The resolution may be divided into multiple sub-tasks that may require a change in :

- System configuration
- Product configurations
- Application component code

Problem tickets are also divided into multiple sub-task including assessment, analysis and providing a solution.

One would suggest that all such work items should be added to the service backlog and resolved in their priority order. Every time a new incident is received, the backlog must be reprioritized. The work item and task prioritization are a continuous process, triggered by any change to the backlog. We can further classify the backlog if the volume of work items is large:

- System Service Backlogs
- Application Service Backlogs
- Service Requests Backlogs

Release backlog

Release backlog is the core of the model. Product and service backlogs provide the required data for release planning to create a release backlog. It is a representation of all the changes that need to be scheduled for a planned or an emergency release. The backlog provides the release status of work items to the release plan.

Release planning must ensure least disruption to business continuity and hence work items selected for a release are assessed to evaluate overall impact on the system. It may be renegotiated and items in release backlog may change. This would result in priority change of product or service backlog items. One may also come to a point where one needs to re-examine and re-execute sprint and service delivery planning.

Re-prioritization

As soon as a new work item is introduced, it is added to the respective product or service backlog. Once an item is included in the backlog, the release backlog is disturbed and is subject to further assessment. Based on the agreed

priority, the service backlog or respective product backlog is updated.

The core of the model is in re-prioritization. Re-prioritization means changing the sequence of tasks in the release backlog. The point is, why do we need to re-prioritize?

Changing demand

The model is agile, and it needs to react to changing demands. An inclusion to any of the backlogs will call for re-prioritization. Some examples of impact of change in business requirements are:

- Priority of the already scheduled tasks is changed
- Reschedule the fix of a P4 incident to a later date

In a service landscape, the utmost requirement is to comply with the Service Level Agreements (SLA). However, one cannot change the assigned priority of the incidents to simply meet SLAs in the form of reprioritization. We can change the current sequence of items.

Backlog work items are re-prioritized in case of limited resources. To respond to a dynamic environment, the reaction time must be quick. Teams are planned and on-boarded with a fixed capacity, in order to manage the work streams. This limitation calls for re-prioritizing the backlog. However, a scenario may arise where team expansion becomes necessary to meet the delivery commitments.

Re-prioritization base

A commitment can be achieved only when code changes are deployed to the production environment. Our model suggests that we should start re-prioritization from the release backlog. The release backlog will have a backward impact on supporting the service backlog and product backlogs. Any impact to product backlogs will impact sprint plans further.

Service portfolio Kanban board:

As with any model, it is essential that we monitor the progress of the work items, even if it is challenging. We propose to deploy a single Kanban board that represents the Release Backlog items. This board is designed as per the stages of the development and support life cycle.

Each stage of the delivery lifecycle is constrained with limited capacity of experts to complete the respective tasks. Hence, assignment is always restricted with the number of items in WIP queue depth as in the Kanban model. The WIP queue depth also applies to the work items from the product backlog list. However, the team size is subject to demand forecast and previous experience. Besides, if you visualize the model over a longer duration, you will observe that the WIP queue depth is also dynamic.

This board is known as the Portfolio Service Kanban Board.

Work assignments

On completion of each stage, the task progresses to the next phase on the Kanban board. The Service Manager then assigns the next priority task from the release backlog to the respective team member. This activity also updates the product and service backlogs. This assignment is handled separately for both product and service backlogs.

- **Product backlog:** The Service Manager assigns work items to a team member after planning and re-prioritizing the release backlog items, only during the retrospection meeting after the end of the sprint. The current sprint backlog items are not changed. Here, the work items are constrained by the sprint duration and the depth of the backlog items.
- **Service backlog:** The Service Manager assigns work items to a team member on completion of an item. The assignment of service backlog items is continuous.

The model on DevOps

There is always a continuous flow of changes to production environment in the form of work items and uncertain events. We follow the Kanban principle to manage this flow. This model is not dependent on a specific IT environment; however, we suggest employing the DevOps environment for a higher throughput from the model.

The model suggests the mantra to manage

demand by focusing on release planning. Every change or fix eventually piles up for release and results in ROI only when deployed to production. The model suggests planning items in backward progression i.e., prioritize the items in the release plan.

The following image depicts a pictorial representation of the model, capitalized on the DevOps infrastructure:

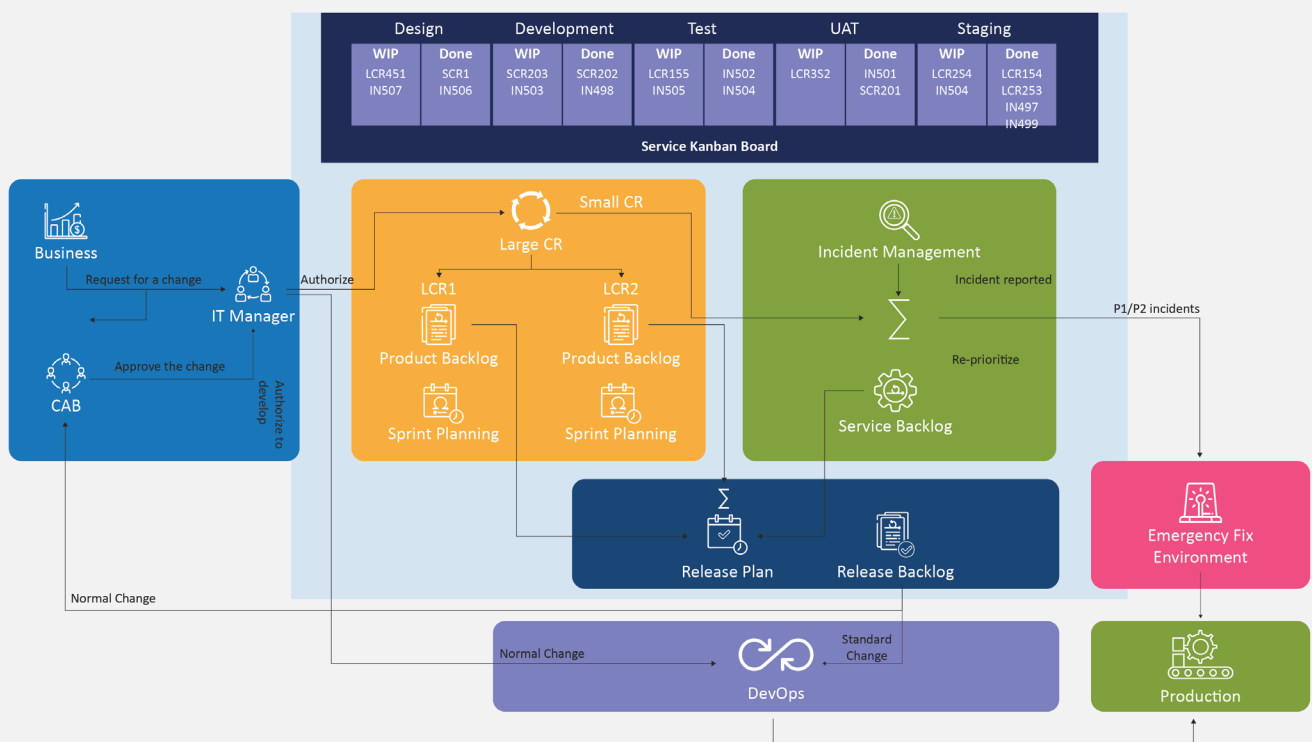


Figure 2: Integrated work management model on underlying devops infrastructure



Conclusion

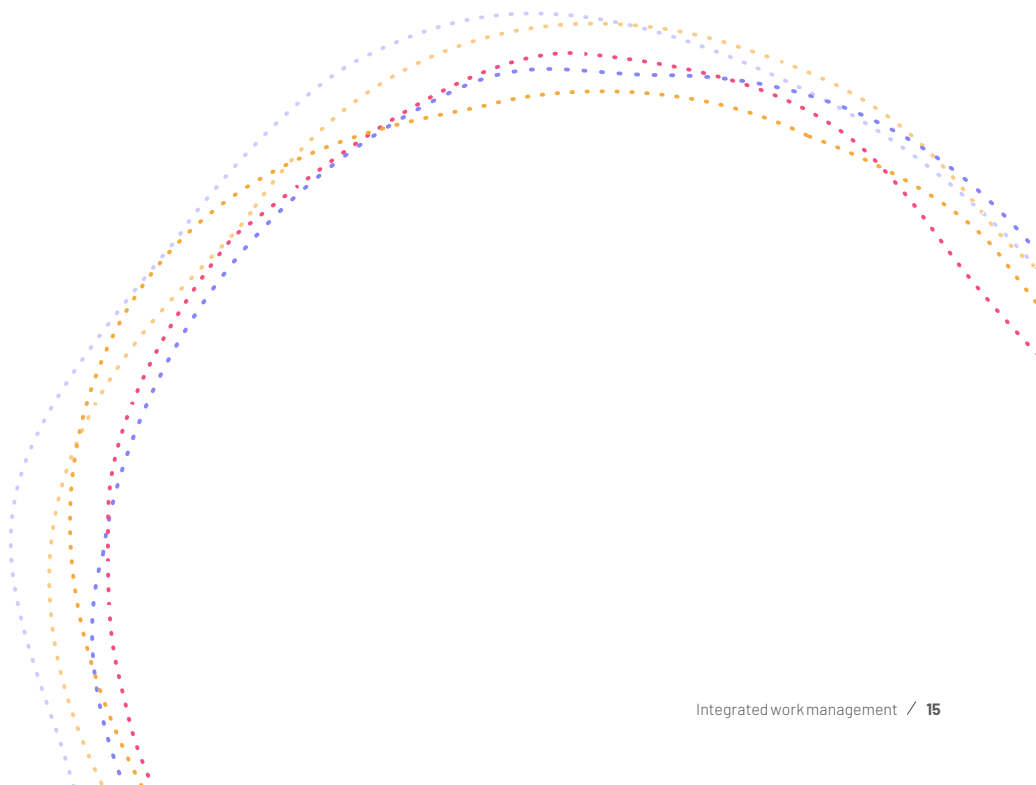
As with the onset of all new models, this model is still in its initial phase. It is an adaptable model, which can provide a base where both the Change the Business (CTB) and Run and Business (RTB) groups collaborate as one team.

We propose the following engagements as the best-fit for our model:

- High velocity of work streams
- Avail benefits of for synergies of combining enhancements, maintenance and support teams together
- Follow agile methodology for software development
- P1 and P2 incidents are not supported
- The model works best in the following conditions:
 - Employed over a CI/CD or DevOps environment
 - Sprint duration is not more than two weeks
 - A weekly sprint can be planned if work stream volume is high
- One release every month
 - Plan for biweekly releases if quick response is required

As with all new models and processes, this model also leaves you with some ideas around its implementation and acceptance:

- Change in delivery organization paradigm
 - Define Key Performance Indicators (KPIs) for the team
 - Create Governance structure
 - Design Performance monitoring structure
- Shorter or negligible transition period from the CTB to the RTB group
- Greater collaboration and handshake as both CTB and RTB groups are one team
- Better involvement of technical architects in designing the overall IT ecosystem because of insights into the on-going production system
- If allowed to evolve, this model can produce high performing and high frequency of changes to production environment
- Better resource utilization, hence cost-effective
- Lean process structure as compared to separate delivery organizations



About the author



Sumeet Popli has been providing IT services since 17 years and has extensive experience of providing technology-backed solutions to various clients. He is one of the leading ITSM and Services Process consultants at Nagarro, and is certified in ITIL Expert, PMP, CSM, LSSGB and other industry standards.

After developing various delivery and governance process models and consulting, he now develops machine learning-based solutions, which are applicable for AMS scenarios.










ABOUT NAGARRO

Nagarro drives technology-led business breakthroughs for industry leaders and challengers. When our clients want to move fast and make things, they turn to us. Working with some of our SAP clients, we continuously push the boundaries of what is possible related to innovations through process optimization and technology progress. Today, we are more than 6000 experts across 21 countries. Together, we form Nagarro, the global services division of Munich-based Allgeier SE.

©2019 Nagarro – All rights reserved

CONNECT WITH US

 info@nagarro.com  www.nagarro.com  [/nagarroinc](https://www.facebook.com/nagarroinc)  [/nagarro](https://twitter.com/nagarro)  [/company/nagarro](https://www.linkedin.com/company/nagarro)  [/user/nagarrovideos](https://www.youtube.com/user/nagarrovideos)  [/lifeatnagarro/](https://www.instagram.com/lifeatnagarro/)