Міністерство освіти і науки України  
Національний університет «Запорізька політехніка»

кафедра програмних засобів

**РЕФЕРАТ**

з дисципліни «Верифікація цифрових систем» на тему:  
«**Канбан»**

Виконала:  
студентка КНТ-227 О.С.Бурдакова

Прийняв:  
к.т.н., проф. Г. В. Табунщик

2019

**Канбан**  — система организации производства и снабжения, позволяющая реализовать принцип «точно в срок».

**Происхождение**

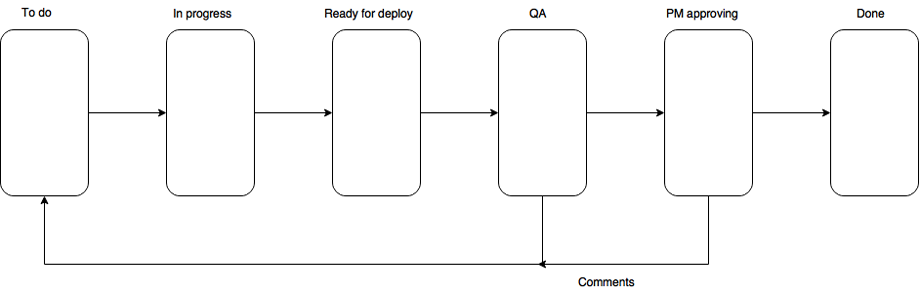
Канбан был изобретён проектными менеджерами на заводах Toyota для того, что бы процесс создания продукта (в данном случае, автомобилей) стал более прозрачным. Классическая схема, используемая в то время, предполагала производство огромного количества деталей в разных цехах или даже предприятиях, не связанных между собой. Перед финальной сборкой могли пройти дни, недели и даже месяцы. Таким образом, если один из отделов допускал ошибку, о ней узнавали только через несколько недель, и все это время выпускались бракованные комплектующие. Убытки в данном случае чудовищны: оплаченная работа, испорченный материал, транспортировка, складирование, переработка и время. Спасительной и ключевой здесь являлась идея создания единого потока.

Поток — это такой производственный процесс, где нет простоя незавершенных задач. Нет пауз, одно движение вперёд. Получается настоящая эстафета: задача, как та самая палочка, передаётся отделами четко из рук в руки. В этом случае ошибки предыдущей стадии очевидны и обнаруживают себя мгновенно. Это позволяет избежать бессмысленных трат, увеличивает качество продукта, снижает стоимость и уменьшает сроки выполнения.

## Применение канбан в IT

Система канбан, продолжая приносить пользу на производственных конвейерах, проникла в сферу программного обеспечения.

Допустим, перед нами рядовая IT-компания, которая при производстве применяет итеративную методологию, а для отслеживания оперативной ситуации внутри спринта используется Канбан доска.



To do — спринт бэклог.  
In progress — задачи, которые разрабатываются в данный момент.  
Ready for deploy — задачи, которые уже выполнены, но не представлены в тестовом окружении.  
QA — задачи, готовые к тестированию.  
PO/PM approving — готовые задачи проходят проверку project owner-ом или проектным менеджером.  
Done — выполненные (завершённые) задачи текущего спринта.

Процессы в спринте здесь проходят таким образом: приступая, разработчик перетягивает задачу из To do в In progress. В один момент времени он имеет только одну задачу, так как параллельное выполнение задач ничем хорошим не кончается. После окончания своей части работы, он перетаскивает её в колонку Ready for deploy.

Начиная с этого момента за дело берётся отдел тестирования. По достижению лимита задач в этой колонке QA инженер инициируют сборку билда, или сервера, или ещё чего-нибудь. В идеале для этих целей хорошо иметь continuous integration инструмент, такой как Jenkins, например. Также допустимо просто попросить разработчика собрать билд или обновить сервер. При успешных результатах тестирования задача отправляется в колонку PO/PM approving, где получает финальное подтверждение и перетягивается в последнюю колонку Done. Если задача не проходит тестирование, она снова попадает в колонку To do с соответствующим комментарием. Багрепорт имеет точно такой же жизненный цикл.

Для построения такого процесса необходимо, чтобы у всех членов команды были правильно настроены нотификации. В таком случае можно избежать простоя при переходе задач между исполнителями.

Также желательно осуществлять промежуточные поставки кода в колонку Ready for deploy, так как это сможет обеспечить более оперативную обратную связь. Скопление тасок в любой из колонок называется «бутылочное горлышко». Это значит, что пропускной способности на данном участке не достаточно, и нужно найти причину и решить её.

Для быстрой обратной связи QA инженер должен иметь возможность протестировать отдельные компоненты: БД, API, front-end, back-end.

## Преимущества и недостатки канбан

### Kanban имеет такие достоинства:

1. Гибкость планирования. Команда концентрируется только на текущей работе, приоритет задачи выставляется менеджером.
2. Высокое вовлечение команды в процесс разработки. Благодаря постоянным собраниям, прозрачности процессов и возможностям самоорганизации работники сплачиваются и проявляют искренний интерес.
3. Меньшая продолжительность цикла. Если несколько человек обладает схожими навыками, продолжительность сокращается, если же только один — появляется узкое место. Поэтому сотрудники должны делиться знаниями и тем самым оптимизировать продолжительность цикла. Тогда вся команда сможет взяться за работу, которую забуксовала,и восстановить плавный поток.
4. Меньше узких мест. Лимиты Работы в процессе позволяют быстро находить узкие и проблемные места, которые появились из-за дефицита внимания, людей или навыков.
5. Наглядность. Когда все исполнители имеют доступ к данным, то узкие места легче заметить. Kanban-команды, обычно используют два общих отчета: графики управления и совокупного потока.

### На практике система отлично себя показывает в сферах неосновного производства:

* группы поддержки программного обеспечения или службы поддержки.
* Канбан хорошо работает при управлении стартапами без четкого плана, но где активно продвигается разработка.

### Канбан имеет и недостатки:

1. система плохо работает с командами численностью более 5 человек
2. он не предназначен для долгосрочного планирования.

## Вывод

С таким инструментом компания имеет слаженную работу, где каждый знает свою локацию на общей карте. Перед ней открываются все преимущества Канбана, из которых следует отметить:  
 — Уменьшение времени прохождения каждой конкретной таски и, как следствие, выполнения всего проекта в целом;  
 — Быстрая обратная связь от отдела тестирования;  
 — Раннее вовлечение тестировщиков и, как следствие, никакой головной боли перед релизом;  
 — Высокое вовлечение команды в процесс разработки;  
 — Выполнение проекта в срок за счёт уменьшения времени простоев и автоматизации интеграции;  
 — Высокое качество продукта.