

­

K.P.I. SHENESH PERERA

December 2018

IDM

SYSTEM ANALYSIS & DESIGN

For IFRB

RKTC

2019

STUDY

FEASIBILITY

Contents

[Feasibility Criteria 3](#_Toc11778329)

[Problem #1: Receive information of customer orders 3](#_Toc11778330)

[Problem #2: Print reports of customer orders 4](#_Toc11778331)

[Problem #3: Create sales orders and send them to suppliers in order to satisfy the customer sales orders for the coming month and create lists of items that are required to complete a particular customer sale 4](#_Toc11778332)

[Problem #4: Create a daily report of customer orders that have been completed 6](#_Toc11778333)

[Problem #5: Delete customer orders from the system once they have been completed 6](#_Toc11778334)

[Problem #6: The new workflow must have levels of access: 7](#_Toc11778335)

[Problem #7: The new work flow system should be able to print information of customer orders at a rate of 15 per hour 7](#_Toc11778336)

[The troubles of moving from traditional to agile 8](#_Toc11778337)

# Feasibility Criteria

Instead of having an overall scope, requirements, approach and evaluation, as I suggested use the scrum agile approach, I will break down each problem and then asses each problem’s scope, requirements, approach and evaluation. IFR Belts seem to solve a core set of problems which are:

* Receive information of customer orders
* Print reports of customer orders
* Create sales orders and send them to suppliers in order to satisfy the customer sales orders for the coming month
* Create lists of items that are required to complete a particular customer sale
* Create a daily report of customer orders that have been completed
* Delete customer orders from the system once they have been completed
* The new work flow system should have the following levels of access:
  + - * + Report and update - for the Assistant Stock Controller
        + Report, update and delete - for the Stock Controller
* The new work flow system should be able to print information of customer orders at a rate of 15 per hour

As such this feasibility report will take each problem, then asses its scope, requirements, approach and evaluate. By the end, the difficulty of moving from traditional to agile will be discussed.

## Problem #1: Receive information of customer orders

This task has to be carried out by the Assistant Stock Controller, and he has only report and update permissions. The approximate time taken to develop a solution for this problem should only variate from 2 to 5 working days.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This part of the system is focused on notifying assistant stock controller of customer orders, as such this part of the solution must result in a fully functional notification system. | Database put in place to store customer orders in order to retrieve them.  Appropriate permission hierarchy set in place to ensure assistant stock controller is capable of receiving information of customer orders. | Start with designing the database, then test the design with test data, then proceed to implement the database.  String or Integer based permission hierarchy can be set in place, this may be stored in memory or in the database. | This will reduce the necessity of stock controllers having to go through multiple documents, which is a strenuous task.  All information will be delivered as they happen (real-time). |

## Problem #2: Print reports of customer orders

This task is also carried out by Assistant stock controller. The approximate time taken to develop a solution for this problem should variate only from 1 to 2 working days.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This part of the system is printing reports, so the solution must be capable of syncing with printers. This is a print feature, either as soft copy or as hard copy. | Database put in place in order to retrieve customer orders. | First, the data must be formatted properly (prettified) so that it’s humanly readable,  Then use language native approach for printing soft-copies and synchronization with printers to print hard copies. | This will allow the assistant to send reports of customer orders to other parties. This is a low-priority problem. |

## Problem #3: Create sales orders and send them to suppliers in order to satisfy the customer sales orders for the coming month and create lists of items that are required to complete a particular customer sale

These 2 task are carried out by the Stock Controller and the assistant respectively, the stock controller has excessive permissions that can alter data (delete permissions), so when creating the solution make sure to provide confirmation message boxes for actions that alter data. Time taken must variate from 4 to 9 working days.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This part of the system is a creation feature. Extremely caution must be applied so as to not overwrite/delete existing data. | Database put in place in order to store customer orders.  A graphical interface to create an abstraction level over data storage, so that the stock controllers can store data easily. | Start by designing the graphical interface, then do simple tests and verify the design’s functionality.  After which it is possible to write the business logic. Precautious steps have to be taken. | This feature will completely overhaul the necessity of writing sales orders, and vastly decrease human error.  This is a high priority feature, as customer sales orders satisfaction monthly and list completion depends on this feature. |

## Problem #4: Create a daily report of customer orders that have been completed

This task is done by the assistant; appropriate caution must be taken place so as to not overwrite existing reports. Time taken must variate from 2 to 3 working days.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This is yet another creation feature, but must be done daily so use the notification system from problem #1 to notify the assistant to make a daily report. | Database put in place in order to store customer orders.  A graphical interface to create an abstraction level over data storage, so that the stock controllers can store data easily. | Start by designing the graphical interface, then do simple tests and verify the design’s functionality.  After which it is possible to write the business logic. Precautious steps have to be taken. | This feature can be automated, if the clients will cooperate it is best to automate this feature.  Daily reports don’t contain a lot of information so, it’s a low priority feature. |

## Problem #5: Delete customer orders from the system once they have been completed

This task is done by stock controller, who has delete permissions. Take appropriate precaution to receive confirmation from the user before deleting orders. Time taken should be from 1 to 2 working days.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This is a deletion feature, completed customer orders must easily be separable from those that are not. | Database put in place in order to store customer orders.  Backup system and a logging feature as failsafe procedures. | Start by ensuring that there is a distinct way to separate completed orders from incomplete orders, then implement the delete feature.  It is recommended to log each delete to a file including what was deleted and when, by whome. | This feature can only be implemented after the previous features have been implemented in order to ensure what’s necessary has already been put into place.  As such this is a low priority feature. |

## Problem #6: The new workflow must have levels of access:

This is a passive feature that is required by all other features, this is the most prioritized task after creation of the database. All caveats and pitfalls must be considered appropriately. Time taken should be from 3 to 5 working days.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This is a permission hierarchy, the foundation for all other features. | Database atleast in design stages. | Understand the 2 levels of access, assistant controller and controller, each have their own permissions.  Use bitflags to generate a permission sequence that will resolve to a particular set of permissions for a person. | This is a high priority feature, and must not be erroneous at all. Do appropriate testing to ensure that the permission hierarchy is not breachable. |

## Problem #7: The new work flow system should be able to print information of customer orders at a rate of 15 per hour

This is also a passive feature that may or may not come after the permission hierarchy. Resource allocation and memory management will be 2 pioneer points to focus on during this problem. Performance and memory profilers will be your greatest ally.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scope** | **Requirements** | **Approach** | **Evaluate** |
| This is a ratelimiting feature, it must be made sure that only 15 prints can be made per hour and that this applies to all users of the system. | Database put in place in order to test the ratelimiting feature.  A logging system. | The logging system will be used to ensure the ratelimiting. Each time a print is made, the time must be logged and a global counter must be incremented.  Once the counter has reached 15 then time must be checked in order to ensure whether 1 hr has passed or not. | This is a low priority feature and can be implemented the last if it has to be.  As it does not affect the workflow directly. |

# The troubles of moving from traditional to agile

When your projects are running behind schedule and the outcomes aren’t meeting the standards you’re hoping to reach, it’s time to take a hard look at your project management approach to see if it’s holding you back. Old traditional project management methodologies can be cumbersome and inefficient, and if you’re still using them, odds are good that adopting a newer, better approach like agile could be a game changer for your team.

Making a sudden shift from traditional to agile project management can be a troublesome event, especially if your team members are unfamiliar with it and have a whole new set of new technical terms to learn. Reconciling your old project management methods with an agile framework smoothly requires agility but it’s well worth the effort.

Although agile began as an approach to software coding, the sprint-based structure translates exceptionally to project management.

As such I believe that intensive training and proper orientation must take place during this transition, the company must invest on training their members to understanding the new concepts and handling them one by one.

This is why picking the right agile approach is important when you’re transitioning from a traditional approach. Scrum is your best bet for when you transition from a tradition to agile approach, as the biggest strength of scrum is how the emphasis is on teamwork and communication than it is about learning a bunch of heavy words or putting some new process to work.

Everyone knows how to communicate and communicating efficiently in the right way is what Scrum does. Despite whatever traditional approach that IFRB has followed previously, this transition procedure will be the most ideal as the responsibility is joined.

Why is this ideal? Because it is our human nature to never purposely jump into a pit of fire, however though it is also our human nature to jump into a pit of fire if a lot of other people are jumping in it too. What I mean by this is, that when the responsibility is shared each individual is less pressured, the expectations don’t come individually, the come as a team. Each individual will get to know their role and will contribute in adequate manners, if they mess up, the responsibility for it will be shared in the next Sprint meeting.

Moving from traditional to agile seems very difficult at first, but after you begin the transition you would never want to stop it. Once you go agile, you can never go back. It’s that wonderful.