Haaga-Helia
University of Applied Sciences

Lean-Agile Scheduling Application Design for Ethiopian Orthodox Tewahido Church Parish Council in Helsinki

Mahlet Adenew

Bachelor's Thesis
Degree Programme in BITe 2018

Haaga-Helia
University of Applied Sciences

Abstract

Date

Bachelor's Thesis

Author(s)

Mahlet Adenew

Degree programme Bite

Report/thesis title	**Number of pages and appendix pages**
Lean-Agile Scheduling Application Design for Ethiopian Orthodox Tewahido Church Parish Council in Helsinki	**40 + 5**

The thesis will start with describing the problem it plans to solve, i.e. the scheduling and meeting management challenges currently faced by the Ethiopian Orthodox Tewahido Parish here in Helsinki. The thesis will focus on an agile-lean driven product design and implementation.

The thesis will start describing the problem in detail which will be followed by discussing the various existing alternatives that would fit this scenario. That would be followed by the justification on the need for a new application. The thesis will further continue on describing the theoretical principles of Lean and Agile patterns in software development.

The thesis will then outline the minimum viable product (MVP), which is followed by the solution design. The solution design will outline some of the main objectives that need to be fulfilled. This will be followed by the DevOps principles and practices that are essential in implementing agile solutions.

Even though the thesis will outline some security best practices to be used, topics of web security are beyond the scope of the thesis.

Keywords

Lean. Agile, DevOps, MVP, Lean canvas, Azure DevOps, CI, CD

Table of contents

1. Introduction

Ethiopian Orthodox Tewahido Church (EOTC) is one of the oldest Church in the world. It is also the largest of the Oriental Orthodox Christian Churches. Around 44% of the Ethiopian population are believed to be members of EOTC (Refworld, 2014). EOTC has many Churches around the world. The Debre Amin Teklehaimanot Church in Helsinki of the EOTC is one of the many Churches in the Nordics region. The Churches are interconnected and work together so that they can maintain uniformity of service and adhere to a single administrational structure governed by the Holy Synod located in Ethiopia. The Churches offer various services such as liturgy, gospel, baptism, Sunday school, spiritual counseling, and many more sacramental services (Kesis Lemma, 2018).

Churches under the EOTC have their own Parish Councils to manage their affairs. They are responsible to almost all activities done around the church. Most of the activities require the parish council members decisions that has been made together. Therefore, the secretariat schedules a meeting, prepare and share agendas, meeting minutes and action plans as well as attachments as part of the meeting management responsibilities. Also, there are other Churches belonging to EOTC spread across the world, it's a common workflow to have meeting scheduled regularly beyond borders. Since the council has limited resources, licensed high-end products are beyond its means. It had to depend on a combination of freely available solutions which have not resolved the problem fully (Kesis Lemma, 2018).

The planned thesis work targets to analyze and find a solution for a problem domain around online meeting scheduling and management. Problem at hand will be examined in more detail. This includes assessment of the current situation as well as possible freely available solutions and their short comings. In discussing the problem domain, this project will present a viable solution that will suffice the need of the organization also considering the problems ranging from the use of a common interface to the low bandwidth connection in Ethiopia (Internet world stats, 2018).

2 Goal

This main goal is to help Ethiopian Orthodox Tewahedo Church in Finland reach its maximum efficiency and be more organized. In order to do so, this project would follow currently popular principles of Lean and agile to design both the product and flow of implementation. The principles are chosen to guarantee successful product implementation with minimal investment both in time and resource. In addition to that, the projects intend to share with the different stakeholders of the project on agile-lean product design and implementation.

2 Problem description

Currently, the organization is using a combination of freely solutions. The secretariat uses emails, skype, Viber group messaging and google drive to do all the work daily. Clearly, this way of managing meetings is very inefficient. For instance, If the secretary wants to schedule a meeting then he/she needs to send the agenda of the meeting including place and time to all the members by email or Viber group message. After that the secretary checks all the responses. If only the proposed meeting has more than 60 percent confirmed attendance, the meeting will be set. If not, the secretary will do the same activity repeatedly until the meeting is set. This way of working is very time consuming and exhausting for the members, especially since most of the members are donating their free time for the organization. In addition, when trying to arrange a meeting with the other EOTC branches across the world, the time difference makes it harder and more time consuming to set a meeting date. (Kesis Lemma, R. 02.02.2018)

The second problem is data sharing. After all the bureaucracy, when a meeting is held, there will be documentation. The meeting agenda, list of attendees and absentees, meeting minutes, and additional attachments will be documented. All the documents are sent by email to all the members for confirmation. This involves to-and-fro emails of comments and correction suggestions. After all comments and suggestions are accommodated the secretary sends the final meeting minutes by email. Using email to share the files is not only insecure but also takes patience. In addition, reference documents related to the meeting are also sent by email before the meeting is held (Kesis Lemma, R. 02.02.2018)

The third problem is finding data. When a member of the council wants to search for information about past meetings, they must look through their email, Viber group message or google drive. The older the meeting the harder it is to retrieve any information. This way of working will make the organization

inconsistent, less productive and could also lead to a data loss easily (Kesis Lemma, R. 02.02.2018)

When an old member replaced by a new member, he/she has no means of following up on previous meetings. In addition to that, departing members of the council also leave with all the meeting minutes and attachments, some of which are confidential. An additional problem is security. Currently, the secretariat is using one Gmail account with a shared password (Kesis Lemma, R. 02.02.2018).

The fourth problem is language of the toolsets. The official language of the Ethiopian Orthodox Tewahedo Church is Amharic. However, the toolset that the secretariat is using are all in English. Not to be able to use own language has a major setback in the daily activity. In addition to that, not all the members are tech savvy or have the appropriate level of language skill to use these systems. Every time a council member is changed, he/she requires a training (Kesis Lemma, R. 02.02.2018).

2.1 Existing Alternatives

> *"Unless you are solving a brand-new problem(unlikely), most problems have existing solutions. Many times, these solutions may not be from an obvious competitor."*
>
> (Ash Maurya 2012, Lean canvas, 6)

After a detailed examination of the requirements with the product owner, the required product needs to include enterprise control, shared calendar, meeting management, Amharic language, and needs to be free of charge. (Kesis Lemma, R. 02.02.2018)

Enterprise control is when all the communication of its members is under the organizations control. When a member leaves the organization, he/she will no longer own the privilege of accessing any of the organization communication system or documents. This enables the organization to

maintain high level of security over its documents and confidential information. Shared calendar with the addition of a meeting manager would also be required to enable council members to follow-up on past meetings as well as schedule and plan future meetings with ease. (Kesis Lemma, R. 02.02.2018).

One of the key requirements of the system is that it must be freely available. Taking into consideration the dynamic nature of the numbers and the most importantly the current financial situation the Church is currently, it was underscored boldly that the Church was not in the position to accommodate any non-free products or services at least for the foreseeable future. in addition to that, since the working language of the EOTC is Amharic the Parish council greatly favored an app/service which has Amharic language support. (Kesis Lemma, R. 02.02.2018)

There are several existing alternatives, below listed are some would be candidates in relation to the above-mentioned requirements. In the section below, the list of products and an analysis of their pros and cons is presented as follows:

1. Personal calendars and enterprise suite
    - Google calendar/ G Suite, iCal, Office 365,

Google calendar /G Suite

Google calendar is used to create and manage personal type of calendars. Using G Suite several calendars can be created and shared. Creating an event, sending invitations, meeting room reservation and resources, attachments, check accepted and declined parties and deletion of an event is included. Google calendar has no enterprise control but it’s free. G Suite has enterprise control and can benefit the organization partially. Pricing is involved and do not support Amharic language. (GSuite, google)

iCal

It has personal calendar management system and it is integrated with iCloud. Creating calendars, making invitations, sharing calendars, alert system is some of the features. It is free but has no enterprise control and meeting management system. Also, do not support Amharic language. (IMore, Ritchie; 2016)

Outlook /Office 365

Integrated with other Microsoft tools outlook calendar has several features. Meeting management, calendar sharing, sending an invitation, multiple calendar views are some of them. It has enterprise control and can partially help the organization solve its problem. But there is pricing and don't have Amharic support. (Outlook)

2. Pro meeting management software

- Boardable, ContractZEN, twelve Directors' Portal, Azeus Convene

Boardable

It is a software made for meeting and board management. It provides meeting management with secured documentation system. Arranging a meeting, uploading meeting related documents, and after meeting documentation are included. Accessing past meeting documents is very easy. Having integrated calendar with google calendar, iCal and outlook, it is easier to creating any new schedules or any updates. Its pricing system is dependent on number of the users. It does not support Amharic language. (Boardable)

ContractZEN

Using ContractZen users can arrange a meeting easily. Invitations are automated. Meeting related documentation is secured. It has many more features. Pricing is involved and doesn't support Amharic language. (ContractZen)

Twelve Directors' Portal

It works online and offline. It has meeting management with secured and centralized documentation system. Past meetings can be displayed with meeting agenda, minutes and all related documents. It syncs meeting schedules with personal calendars. Pricing is per user. It doesn't support Amharic language. (Twelve directors' portal)

Product	**Enterprise Control**	**Shared calendar**	**Meeting management**	**Pricing**	**Amharic Language Support**
Google calendar/ G Suite	No	Yes	Partial	Yes	No
iCal	No	Yes	No	No	No
Outlook/ Office 365	Yes	Yes	Partial	Yes	No
Boardable	Yes	Yes	Yes	Yes	No
ContactZEN	Yes	Yes	Yes	Yes	No
twelve directors portal	Yes	Yes	Yes	Yes	No
Azeus Convene	Yes	Yes	Yes	Yes	No

Fig 1 Comparative analysis of existing alternatives

Azeus Convene

The meeting management system support all before and after meeting process. Documentation is supported with high security and can be accessed from anywhere. It is a platform friendly, works on any device. Amharic language is not supported and there is pricing. (Azeus Convene)

Figure 1 shows why the existing alternatives do not fulfil the requirements. As presented in the table some of the products can benefit the organization but they all have a price and no language support.

2.2 Unique value proposition

A product or a service with no unique value proposition (UVP) is just another fish swimming in the sea. UVP, which is also referred to as unique selling proposition is the core benefit your customers get when using your product / service. Customers always need to be convinced why a product is worth buying and could benefit them like no other in the market. UVP describes how the product /service solves the customers problem and what makes it stand out in the market. UVP should be the first introduction of the product / service to a prospective customer. (Ubounce)

Steps for discovering the UVP of a product.

Defining target audience: - Instead of trying to deliver for everyone concentrate on the customers who need your product/service. Many businesses try to make everyone a customer and end up lost. Try to study your customers in detail. Who they are and what they want. (BPlans)

Defining competitors: - studying competition, what they are doing, how they are doing it, what they are missing, knowing their strength and weakness, will help you find out what they are not delivering to the customers. Ask and observe competitor customers, what they like and don't like. Knowing what is there already

and what's not assist you in designing a competent product. (Small Business BC, 2017)

What the product/service can offer: - Define how your product can solve customers' problem in a different (unique) way. Compare your product/service with your competition and find out what is unique in your product/service that differentiates you from the competition. (BPlans)

Become the myth buster: Analyse popular misconceptions and stereotypes surrounding your business and utilize it your benefit. It can either be by dispelling them or using them to your advantage, with the core idea being delivering value (BPlans)

Deliver a clear message The UVP goes beyond identifying what makes a product unique. It should match with the brands vision. Therefore, the first step is to identify the key ideals your business stands for and have your product project that. (BPlans)

Put you UVP Front page Once the mission and UVP are figured out, they should be transformed to marketing campaign. (BPlans)

3 Lean Software Development Principles

Requirements in software development are mistakenly perceived to be static and non-changing. However, in reality, they often change, and this change affects the fixed design that has been implemented with the assumption of a non-changing requirement. This has been a critical factor in failure of software projects. (Dasari. Ravi Kumar, 2005, 1)

The process of developing software is implemented in linear phases such as, the waterfall model, which involves the client or the user to involve late in the development cycle (depriving the value of customer / user feedback early in the development). In addition to that, this makes testing possible only in the later phases, making fixing bugs and/or implementing changes very difficult. Furthermore, the hierarchical nature of organizations managing software

development makes the process very rigid and inefficient. One instance is that, a request for decision has to go all the up the organizational ladder and then down. product might be obsolete by the time it is delivered. (Dasari. Ravi Kumar, 2005, 2)

Lean software development principle can be used to solve these problems (Dasari. Ravi Kumar, 2005, 2). Lean software development is about delivering most value to the customer at the right time using principles such as eliminating waste, empowering people, optimization and continuous learning curves. (Ebert, Abrahamsson & Oza; 2012, 23)

Lean software development helps minimize cost, defects and programming effort. Lean software development principle is derived from lean manufacturing system by Marry Poppendieck and Tom Poppendieck. (Planview Leankit)

3.1 Eliminate waste

In lean software development, the first step is defining value. Lean focuses on the tasks that adds value to customer. Any task that does not add any value is therefore side-lined from development. (Dasari. Ravi Kumar, 2005, 3)

Anything that comes on the way of delivering value to the customer at the right time and place is considered as waste. Like Toyota production, lean software development targets on optimizing the time frame by eliminating all the unnecessary wastes. (Poppendieck & Poppendieck 2012, 24)

In lean, Inventory is considered as waste because it requires effort and resource before it can be delivered to customer successfully. The larger the inventory the more complex it gets and costly it becomes. With that context, code that is not delivered to customer is considered as inventory. Partially done work carries the investment but not the return. In addition to that, as the partially done code grows so does the risk and quality issues with the software. Lean advocates as less work-in-progress as possible.

The biggest waste of all is extra features. All features that are not adding value to the customer at the time of their release are considered as an extra feature. Usually two-third of the features in a software are used seldomly. In addition, extra features add more complexity to the code, unnecessary expense and testing. Since the code considerably grows in complexity, the cost of maintaining it becomes quite high and they are also one of the damaging factors to software efficiency.Lean always focuses on finding a way to deliver 80% of the value by only writing 20% of the code and continue in developing the next most valuable feature. (Poppendieck & Poppendieck 2012, 23-25)

3.2 Build quality with-in

Building quality focuses on eliminating defects from the beginning rather than testing them later. By testing more frequently, defects should be fixed as soon as they are detected. New phase shouldn't start before fixing the detected defects. Usually when on process or partially done works are in line waiting to be tested, Lean considers them as wastes. The main target is to transform these wastes to delivered value. (Poppendieck & Poppendieck 2012, 25-29)

One popular technique of building quality with-in is by using test-driven-development (TDD). TDD is done by writing a failing test before writing enough code to pass the failing test followed by refactoring the implementation. This process continues in a cyclical manner until the user story is complete. With TDD it is easy to detect defects early and write quality code [a]. Because of this, productivity increased as the developers spent less time tracking bugs. Mary and Tom Poppendieck mention an instance where a team was three times more productive with far lesser defects. (Poppendieck & Poppendieck 2012, 25-29)

According to Marry and Tom Poppendieck "Do it right the first time" slogan should be interpreted in a such a way that developers use principles like TDD and continuous integration to create code that behaves predictably in that time. It does not however, mean that once a code is written, it should not have to change. Such an interpretation has led developers adopt disastrous patterns which ended in failure. (Poppendieck & Poppendieck 2012, 25-29)

3.3 Creating knowledge

Creating collective Knowledge is also one of the key goals of a lean oriented development. Lean encourages teams to share and retain knowledge that was learnt during project. (Poppendieck & Poppendieck 2012, 29-32)

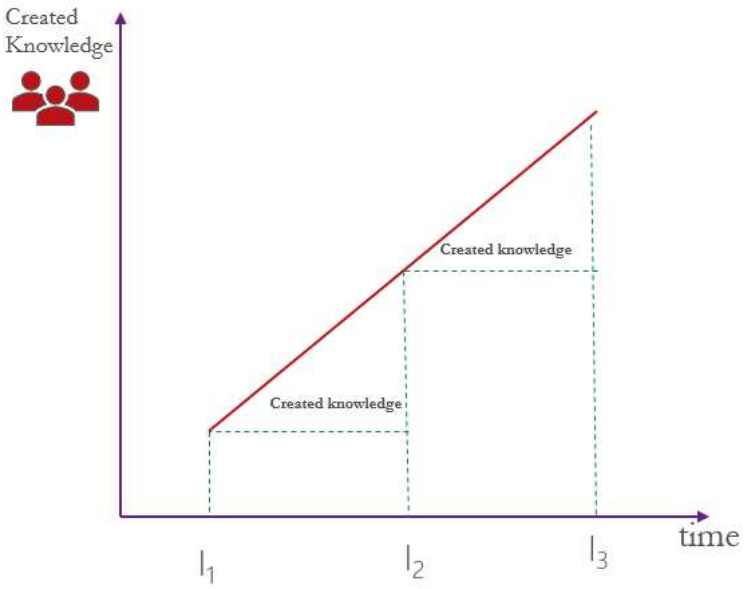


Fig 2. Knowledge creation with successive iterations

This can be achieved by using several techniques such as programming in pair, making documents, by reviewing code more frequently and comment, making the development team to share knowledge, arranging trainings for the development teams as needed. (Planview Leankit)

3.4 Defer commitment

Defer commitment should not be misunderstood for laziness, but on the contrary. It is a process of making decision as late as possible with as much information acquired. This practice decreases assumptions and encourages the team to decide based on validated learning. Additionally, it gives the team the opportunity to check other possible options. (Planview Leankit)

To better explain the concept, Mary and Tom Poppendieck present an analogy of emergency responders. Emergency responders calculate the amount of time they

have before they absolutely must decide. With that in mind, they wait until they have enough information while not passing the time limit to decide. (Poppendieck & Poppendieck 2012, 32-33)

3.5 Respect

Developing successful products requires a team of highly skilled individuals. To continue the culture of success, it is important to make sure that the team is given ample time and resource for competence development. Based on an enterprise guideline and a desired goal, teams should be credible and empowered to be self-organized to accomplish the goals. Respecting people can't be practiced by making the teams to work as they told. Most successful products have a good leader. Good leaders can grow from respecting people and these leaders develop a system that encourage teams to be independent and work hard towards successful product. (Poppendieck & Poppendieck 2012, 36)

3.6 Deliver fast

In lean software development, delivering fast is not about working vigorously all the time just to deliver the value to the customer as fast as possible. It is delivering the most value in a simple solution to the customer and continue to the next iteration based on the customer feedback. Most software developments fail due to advanced detailed requirements which end up being complex applications. (Poppendieck & Poppendieck 2012, 34)

4 Implementing Lean

Albeit in a scenario of a Start-up which nonetheless upholds true in other areas as well, Eric Ries in his book lean Start-up argues the first thing to do is to find out which are the riskiest “leap-of-faith” assumptions and validate them as early as possible with the most minimum efforts. To achieve this, we would need an iterative and experimental process known as the Build-Measure-Learn feedback loop. Once the core problems have been identified, the team should enter the build phase of the loop with the Minimum viable product (MVP). (Ries, 2011, 76)

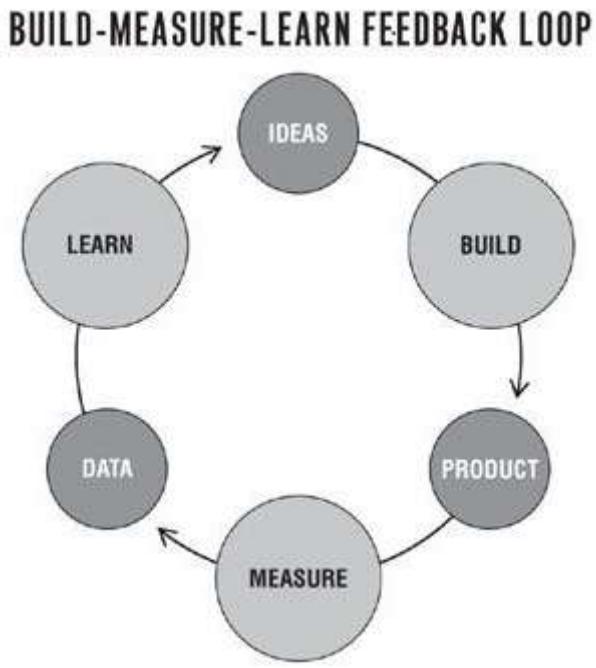


Fig.3 Build-Measure-learn loop (source. Eric Ries, The lean startup)

Eric Ries defined the MVP as

> “*A Minimum Viable Product is that version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort*”
>
> (Ries, 2011, 77)

MVP enables one full revolution of the cycle. The MVP is not a phase or a section of a product, however a version that delivers value as well as information in the form of feedback from customers. At the earliest development stages, this information is worth their weight in gold so to say. It is also important to note that the MVP misses out on several planned features of the product. (Ries, 2011, 77)

The next stage of the cycle is the measure phase. At this point, it is important to lay in place a good framework, both technical and procedural to gauge the customer reaction and seeing if the product has met its objectives. Collecting these feedbacks is important in creating what Eric Ries calls learning Milestones. These can be qualitative or quantitative feedbacks. Qualitative feedbacks are reactions from customers as the feature they like best, as the feature they most likely would like to have etc. while qualitative feedbacks could be the number of customers using the product, conversion rates, session duration (for instance if it is a web product), how often do they revisit etc. In order to measure the quantitative feedbacks, we would need to have a good automated insight behind our applications. (Ries, 2011, 77-79)

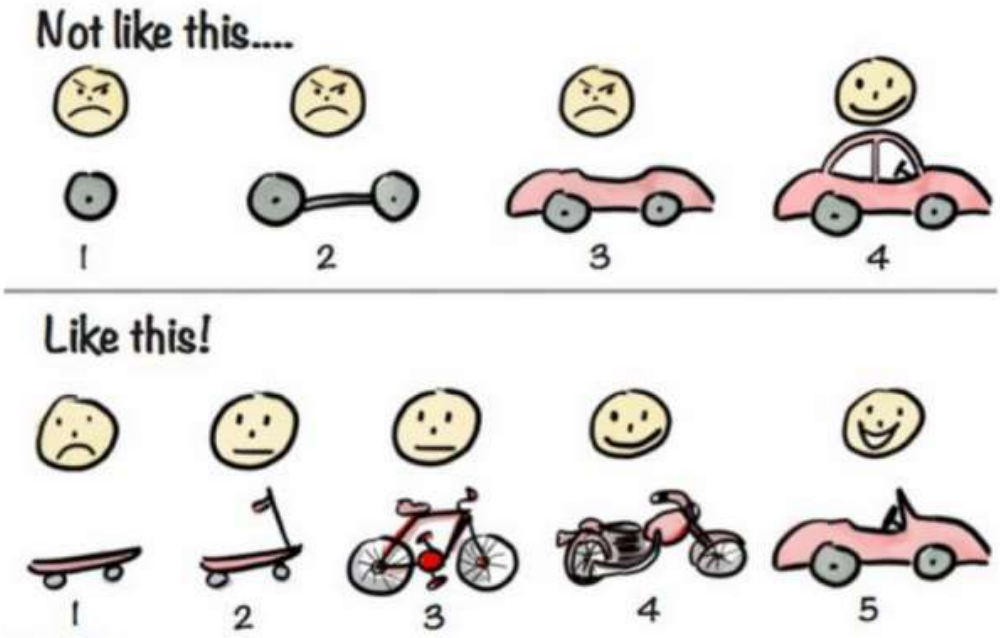


Fig.4 An example of progression of development (Agile incremental delivery visualized, Willem-Jan Ageling)

the last phase of the cycle is to analyse the information and decide whether the course set for the product is paying dividends or if it is time to *pivot.* Decision is made at this stage, which is based on validated learning as to whether to continue the path or rearrange the features if newer insights are found. (Ries, 2011, 77)

Interpretation of measurements is as important if not more as efforts to implement further releases of the product version across the cycle. In this phase, it is very important to grasp the technique Eric Ries calls innovation accounting. The first thing is to clearly set the baseline MVP from which to properly collect measurement. (Ries, 2011, 117)

The MVP would be validating multiple burning assumptions at a time or single one, the judgement as to which is the best approach is a subjective one. There is also an old trick of pre-ordering a product that has not been developed yet. That is with 0 effort the team can gauge initial responses from prospective customers. (Ries, 2011, 118)

Once the proper baseline is set, then the next phase is to concentrate on tuning the engine which would take the product from the baseline towards the ideal. This is done through a series of changes and optimizations. Through this process, the team developing the product should decide on whether or persevere or pivot. If the feedback indicates that the direction is correct although with needed corrections, then the team engages in what Eric Ries calls tuning the engine. The team will be at work to deliver the missing pieces both in terms of product increments as well as marketing initiatives that better communicate their purpose. However, tuning the engine is not always the best answer, as there is no point optimizing or adding new increments to a product which customers do not like (Eric Ries Chapter 7). As Ash Maurya famously quoted “Life is too short to build something nobody wants”. In this case the team should analyse the learnings and should consider pivoting towards the direction which the feedback generally leads. (Ries, 2011, 117-133)

The decision to persevere or pivot is pivotal for the team and hence should be based on hard facts and untainted truths. One of the main pitfalls of this scenario is basing

judgement on what is known as vanity metrics or false positives. Sometimes, there is data that might be inferred as the product is delivering on previously held assumptions. But the reality is far from it. This kind of information creates a delusion which would lead a team to decide on perseverance towards death. This kind of metrics is known as Vanity metrics. To avoid being sucked into that pitfall it is important to lay clear and practical metrics analysis methods. One of the methods used is known as Cohort analysis. According to Merriam-Webster, a cohort is defined as

> *"A group of individuals having a statistical factor (such as age or class membership) in common in a demographic study"*
>
> *(Merriam-Webster, 2018)*

This kind of analysis deepens the analytical view towards groups / cohorts of users who are using the product/service. For instance, in his book Lean start-up, Eric Ries mentions one example of a web-based product and how the data is interpreted. The cohorts or groups for this service could be registered users, active users, logged-in users, Activated users, active users. (Ries, 2011, 117-133)

One scenario would be where the service might be having more than the target number of registered users and the registration keeps amusing as with further iterations of the product, however the cohorts of active users or converts or engagement of registered users is stagnating. (Ries, 2011, 117-133)

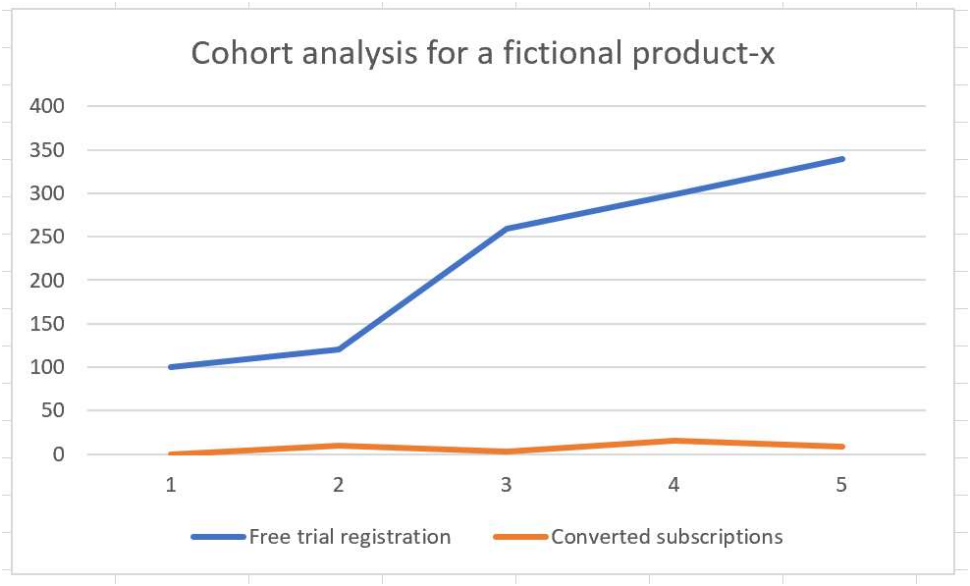


Fig.5 Cohort analysis for a fictional subscription-based product

In this scenario, a team would be duped to think the product is amazing if the team bases the judgement on the number of registered users alone. The team should be encouraged no only to see the registered users but also the failure in conversion and engagement. Such an in-depth analysis is key in deciding the future. (Ries, 2011, 117-133)

Another important methodology of acquiring feedback particularly the qualitative feedback is to do an old school interview. Eric Ries is popularly quoted as saying *"People are metrics too".* In his book, Running Lean, author Ash Maurya proposes a 20-minute interview with a hot prospect. His argument being, if one fails to convert a prospect with a 20-minute one-to-one meeting, it will be much harder to do so with 8 seconds visit to the web site. He deconstructs the interview (in case of a web product) as the following diagram. (Maurya 2012, 129-131)

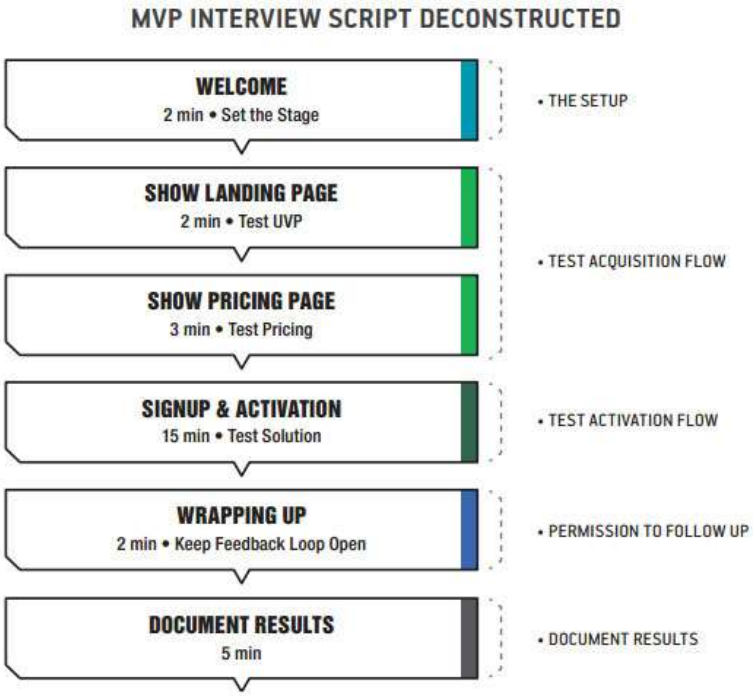


Fig.6 Deconstructing MVP interview. (Maurya 2012, 130)

The prospect would be asked how the landing page feels, does it communicate properly, does the customer make it all the way towards registration. In asking those questions, it would be important to note that if seeming obstacles are due to bad user experience design (UX) or other issues. (Maurya 2012, 129-131)

4.1 Lean Canvas

Every project starts with a plan and a lean project is no exception, although there are obvious differences on its structure and documentation. In his book, Running Lean Ash Maurya strongly argues on minimizing efforts and maximizing value and therefore takes a strong stance on a 60-page initial planning documentation which will likely undergo several changes through the process. Instead, he proposes model which he has adopted from Alex Osterwalder's Business Model Canvas **(Business model generation – Wiley).** The choice for lean canvas as a way documenting the plan has three main advantages: (Maurya 2012, 4-5)

1. Fast: It is fast to create. One can be done in a few hours instead of days
2. Concise: The canvas guides the creator to be short and descriptive.

3. Shareable: Lean canvas is easy to share and understand. Therefore, it is easily possible to get several feedbacks on the planning phase.

After the lean canvas generation, the next step is to identify the riskiest parts of the plan and to validate them. In the view of a Start-up, which Ash Maurya explains the stages, the first step is the Problem/Solution fit. Problem solution fit is the first step since it determines whether the team/person in charge of building the product has a worthy problem to solve. This phase comes down to having a proper response to three questions. (Maurya 2012, 5-13)

1. Do the customers have a demand for this?
2. Will they be willing to pay for it?
3. Is it feasible?

If the team passes the first step, the next one is known as the Product/Market fit. It is in this stage that the team builds the MVP and validate whether the team has built something that solves the problem outlined. It is in these phase that the team launches the build-measure-learn loop with the MVP. (Maurya 2012, 5-13)

In this stage, the team puts a solid base for collecting both the quantitative and qualitative metrics.

Fig.7 Three stages of a Start-up (Maurya 2012, 9)

After this stage, provided that the team has a validated product, the next stage is concerning scale. How does the product scale to cover grounds for a wider customer base? (Maurya 2012, 5-13)

4.2 Implementing lean for EOTC scheduling app

The first thing to underscore is that even though this product is not being developed for a wider paying customer consumption, there are similarities between users of this product and its development technique that warrant the usage of lean methodologies. The similarities rise from the base idea that the customer is currently facing a problem but do not have a plan for a solution. They just have a problem worth solving. However, the solution is not clear and therefore an experimental, iterative feedback-oriented and fast approach would best serve the development. Hence, implementing a lean based design approach is found to be the most optimal in this scenario. In addition to that, the users of this system would be synonymous to a would-be user of a lean product from a Start-up. With that in mind, the first thing to do was to create a lean canvas for the product.

The problem was clear as it was evident in the current working of the EOTC parish council. Hence the next step was to search for existing alternatives and create a comparative analysis and thereby decide whether it was worth the effort to build a product. As seen in section 2, the existing alternatives all failed in at least not having the Amharic interface while some failed as not having the other features while some required payment. Hence from the comparative analysis, it was clear that there was a need for the product.

The next step was to come up with a solution design that would solve the problem at hand and what would be considered a unique value proposition of the product to be built. The solution would be a multi-lingual interface with calendar at the core of the application. The present, past and future meetings would be visible in the calendar.

The details of those individual meetings would be a click away in the calendar view.

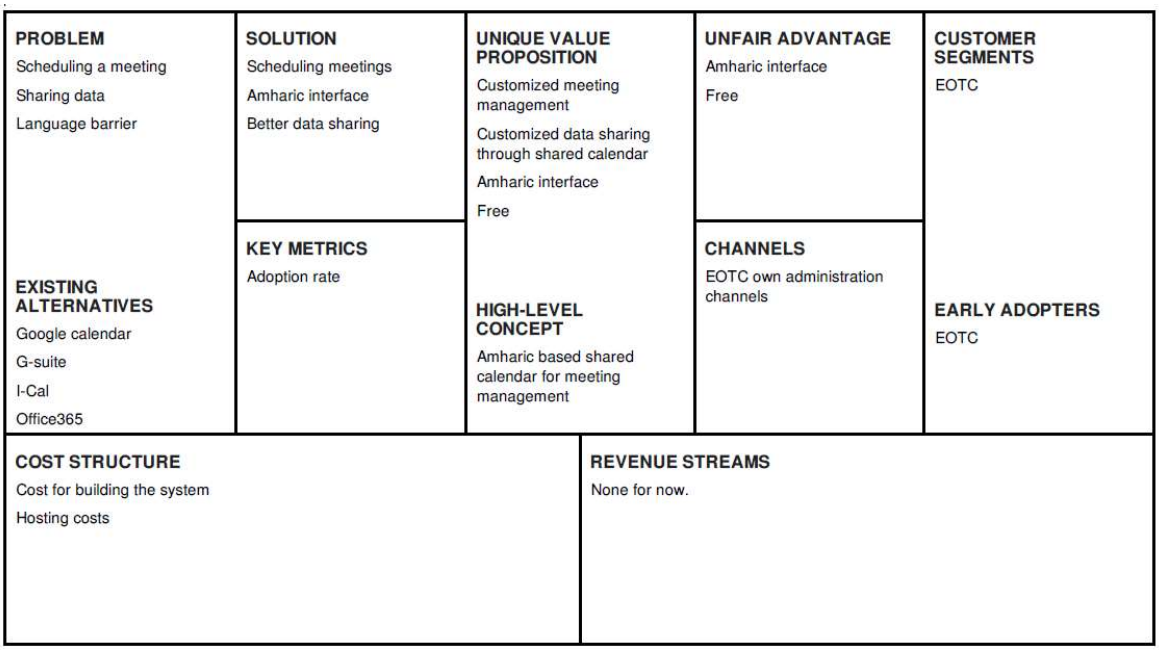


Fig.8 Lean canvas for EOTC scheduling product

As described above in the Lean canvas, the early adopters and customers was the EOTC and the key metrics to measure was the adoptability of the system by the Parish council members. The current cost structure is centred around the cost for the effort for implementation and hosting cost. However, in the scenario of scale the cost structure might differ significantly. Since this is a pro-bono project it is intended to be zero revenue, however, adoptability remains a key metric for determining whether the product has been valuable.

4.3 Minimum Viable Product (MVP) for EOTC scheduling app

As discussed earlier the MVP should be that first incremental release which would warrant a feedback from the customers. In addition to that, the MVP will contain features which are considered to be the most vital and risky. From the solution design and problem description, it has become clear the central role that the calendar plays in addition to linguistic and detail features in the of the individual meetings. Therefore, it is considered important to validate the shared calendar, meeting scheduling and Amharic interface to be part of the MVP release. In the MVP, other basic product features such as security, role and user management will

not be included in the release. These sets of features are quite common across a big range of products and their necessity needs no validation or feedback and therefore they will not be included in the MVP. Features such as role-based security will however be included in the first version production release of the product.

The purpose of the MVP in the product scenario is to cultivate valuable feedback. And since the admin level user management features and other related meta-features will not be implemented, the test bed for feedback collection will be a test environment which would only be through active participation of the Parish council members and with test data. To this end, the challenge the current problem has brought forward has also become a motivation for the council to find a better solution and hence assures active participation.

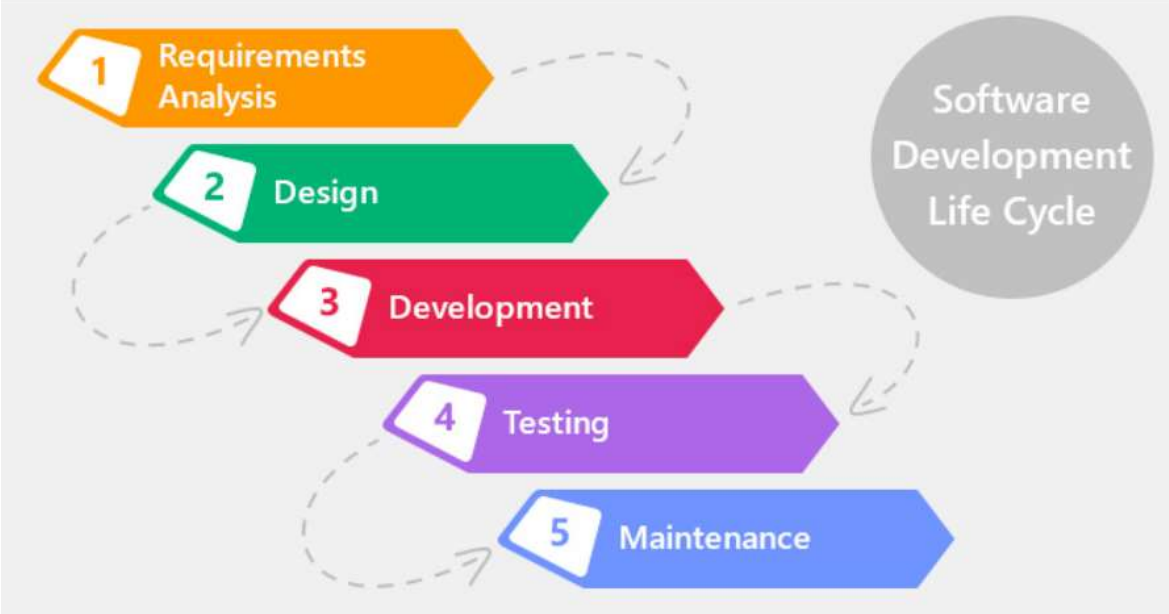
Building the MVP is an important step, however, in addition to that also there are other important technical and non-technical steps that must be included in the MVP development. The main purpose of the MVP is to get valuable feedback and therefore, we must put in place to get viable metrics while the MVP is being used which would provide enough qualitative and quantitative information for which the decision of the future can be decided. The decision on whether to pivot or persevere depends on the metrics and a meticulous translation of the facts there of. To this end, it is of utmost importance to implement an extensive automated monitoring solution into the product as well as preparing a questionnaire that will be used with ask the users regarding their impression and experience of the increment.

Even though the MVP is the first increment, it is also beneficial to plan in some what higher unrefined level what would be the features that would be validated further down the line. Provided that a good course of action could be set on the shared on the calendar, the next increment to validate would be the content management of individual meetings. These includes the meeting minutes, attached documents, attendees list and their approval / rejection with comments of the minutes. In addition to that, the next increment will release some features related to user roles and their functions as described the in section (Solution design) such as meeting scheduling can only be done by a member of the secretariat etc. After these main features are

iterated and are ready, the next would be implementing and validating non-functional requirements such as performance. With all these in check, if there is no pivoting in the process, version 1.0 of the system will be released and further iterations will continue on this released version. However, if there is information that are strong enough to warrant a pivot scenario in the development process, then team must be prepared to adopt and advance. That is also one of the reasons the further iterations after the MVP are vaguely planned in order to save resources.

4.4 Agile software development

In the early days of software development process, the waterfall model was the first process model to have been introduced. It progressed in a linear manner where there are specific phases. Each phase must be completed before the next one can begin. There is no overlap in between phases. The process begins with requirement specifications and goes down the line as seen in the figure below all the way to maintenance. (SDLC - Waterfall Model)



Fix.9 Waterfall model (source. XB software)

However, this model has had several shortcomings. One of the main disadvantages was such that it assumed requirements to be static and unchanging. This meant, when requirements actually do change in the middle of development, the waterfall

model does not have the right process to adopt to it. In addition to that, no piece of working software is provided until the very end phases which makes it quite difficult to include any change late at that phase. The waterfall model does not provide a framework where retrospection and revision could have a valuable place in the process as well as the techniques to measure the progress within stages. (SDLC - Waterfall Model)

Due to its high risk and rigid approaches, its popularity decreased over time as the new agile software development methodology kept gaining traction. Agile software development was started as a popular manifesto by some developers who were convinced to change the way software was made. According to Agile alliance, Agile is not a framework but a mindset that emphasized close collaboration between developers and business stakeholder, continuous delivery of value and teams given the freedom to self-organize. (Agile alliance)

These professionals came to what is popularly known as the agile manifesto

> *"Individuals and interactions over processes and tools*
> *Working software over comprehensive documentation*
> *Customer collaboration over contract negotiation*
> *Responding to change over following a plan"*
> *(Agile Alliance Manifesto)*

One of the popular applications of agile software development principle is Scrum. Scrum is quite efficient in delivering iteratively developed software. Scrum is inherently adaptive to the changes in software development. The theory of Scrum based on the principle of empiricism which emphasizes that knowledge comes from experiences and that decisions must be dependent on it rather than assumptions. Scrum applies iterative and incremental model to better face the unpredictability of software development process. (Schwaber, Sutherland; 2017, 3)

The three main pillars of Scrum theory are:

Transparency: This principle stipulates that the main areas of the development must be communicated across those who implement the systems and other stakeholders in a standard format so that they share a common understanding of the status. (Schwaber, Sutherland; 2017, 5)

Inspection: Scrum members regularly employ inspection on the scrum artefacts to identify any anomalies as early as possible. The inspection however, is done in a systematic manner so that it does not interfere with the actual development. (Schwaber, Sutherland; 2017, 5)

Adaptation: When it has been determined that there is a deviation and an adjustment is need to the development process, Scrum has events targeted to facilitate adaption with minimal costs. (Schwaber, Sutherland; 2017, 5)

Scrum team

A scrum team is comprised of various roles who mainly are the developer (scrum team member), a product owner and a scrum master. Scrum teams are by nature flat, cross-functional and self-organizing. The team does not have a hierarchy and decision and responsibility is collective. The team chooses what's best instead of accepting a downward stream of guidelines. The scrum team is organized as an independent unit capable of delivering value all on its own without any external dependency on other teams or departments. The team has all the skills needed. The structure and cross-functionality enable the team to be more creative, productive and adaptive. (Schwaber, Sutherland; 2017, 6)

The product owner is responsible for managing the product backlog. He/she is responsible that the product backlog is clear and up to date. In addition to that, he/she is responsible for clarifying questions and making sure that the best possible value is delivered from the backlog. There are two well known principles that must be followed by any product owner while managing the backlog. They are known as

DEEP (Detailed-Estimated-Emergent-Prioritized) and INVEST (Independent – Negotiable – Valuable – Estimable – Small - Testable). DEEP principle is used in managing the structure of the backlog while INVEST is used to manage the structure of the user stories in the backlog. The backlog must act as a stack which has the highest priority items at the top. INVEST principle acts as a check-list for any user story. If a user story fails to meet these criteria, a team may reword it, break it into smaller parts or completely change it. Each user story must reflect INVEST in their structure. Using DEEP, the user stories that will be implemented soon should have the appropriate details needed while the ones which will not be needed soon could be more generic. Product backlog is not just a list of backlog items but rather it is a planning platform and hence requires that the backlog items are estimated properly. Consistent with agile principle of adapting to change, the backlog is assumed to be ever-changing while the items in the backlog are properly prioritized.
(Schwaber, Sutherland; 2017, 15)
(Agile alliance, INVEST)
(Mike Cohn, 2010)

The scrum master is one of the key roles in a scrum team. He/she is responsible for making sure that scrum principles and guidelines are followed by the team, he/she also takes care that the team have all their need to do their work. In addition, to that the scrum master facilitates scrum events. The scrum master is also main conduit between the product owner and the team. Among the many functions of the scrum master, one of the main ones is to make sure that the team members understand the domain and specification properly. He/she also investigates and optimizes product backlog management technique. The scrum master also acts as a coach to the team in helping them better achieve under scrum while also making sure that obstacles are removed from developers.

Sprints

The core of scrum is the sprint. The sprint is a timeboxed period where the team will work to deliver a potentially shippable product increment. The sprint has a pre-defined goal which does not change once the sprint begins. The scope, however, may be negotiated with the product owner. (Schwaber, Sutherland; 2017, 9)

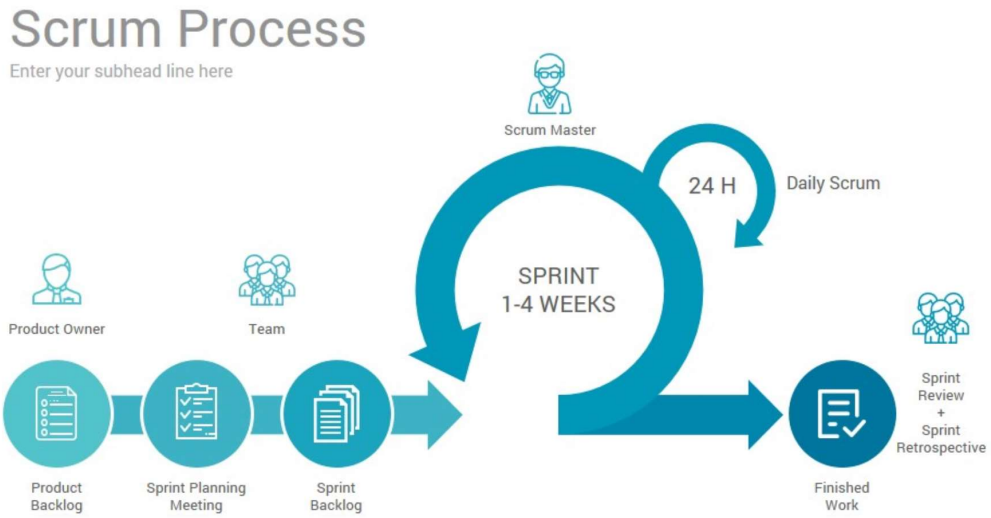


Fig.11 Scrum (source. Brainhub)

Each sprint can be considered a mini project with a potential of shipping an increment to the production. Every release of the sprint enables the stakeholders to give feedback which will be usable in successive sprints.
Before a sprint is started, it is preceded by a sprint planning. During sprint planning the development team estimates and plans what could be incorporated in the net sprint. The process and decision is collaborative in nature, while the scrum master facilitates the planning and estimation the product owner gives his input on prioritization and clarification. (Schwaber, Sutherland; 2017, 9)

Scrum events

The main principle behind scrum events is to minimize meetings and increase a regular status check on progress. Events are timeboxed and each event has a

specific objective. The scrum master enforces efficient time usage and that the team adheres to the culture. (Schwaber, Sutherland; 2017, 8-14)

1. Daily scrum: A daily scrum is a short daily meeting where every team member shares their daily status, obstacles and plans until the next daily. The daily scrum is quite short, issues require longer discussion are deferred. (Schwaber, Sutherland; 2017, 8-14)

2. Sprint review: A sprint review is an event that gets held at the end of each sprint and its main purpose is to inspect the latest product increment, review the current state of the backlog. The sprint review also analyses other influences that affect the product development and also gives valuable input for the next sprint planning. (Schwaber, Sutherland; 2017, 8-14)

3. Sprint retrospective: Another important event which is at the heart of scrum is the retrospective. Every sprint will have this event where all the stake holders reflect on what was good, bad and needs an improvement. The event is facilitated carefully by the scrum master as its intention is to progress positively rather than being a stage of criticism. (Schwaber, Sutherland; 2017, 8-14)

Scrum of one in this project

In this project for the EOTC, scrum would be ideal agile development method because of the dynamic nature of the specification and the need to validate the assumptions as the work is going along. For that kind of development, an agile approach would best serve the purpose. However, adopting scrum in this kind of setting has its challenges. For one, the stakeholders are new to this kind of development pattern and hence training the owners will take some time. In addition to that, most of the members of the Parish council who would benefit from the product have quite a hectic schedule which makes an active participation a challenge.

5 DevOps

According to Microsoft Docs, DevOps is defined as the

> "The union of people, process, and products to enable continuous delivery of value to our end users."
>
> (Microsoft docs, what is DevOps, 2017)

DevOps is a process of integration of development and operation process into a multi-disciplinary team responsible for not only the development but also the deployment and monitoring of the application. (Microsoft docs, what is DevOps, 2017)

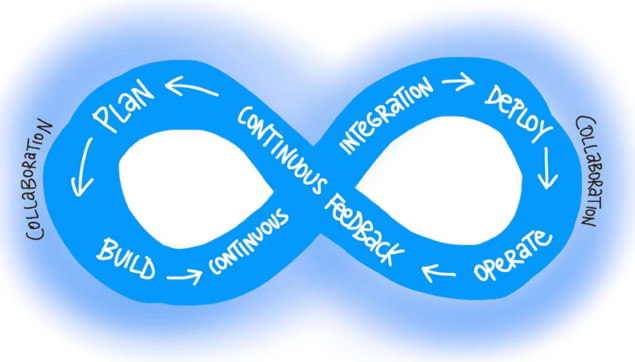


Fig.12 DevOps (Microsoft docs, what is DevOps)

As we have seen before, lean principle is about pursuing development based on validated learning. To achieve this, a team designs a build-measure-learn cycle to experiment quickly and get feedback and iterate further. When adopting Lean, it is quite critical to shorten the build-learn-measure cycle. With DevOps, a team can shorten the cycle by automating more steps to improve the release pipeline as well as telemetry reception. (Microsoft docs, what is DevOps, 2017)

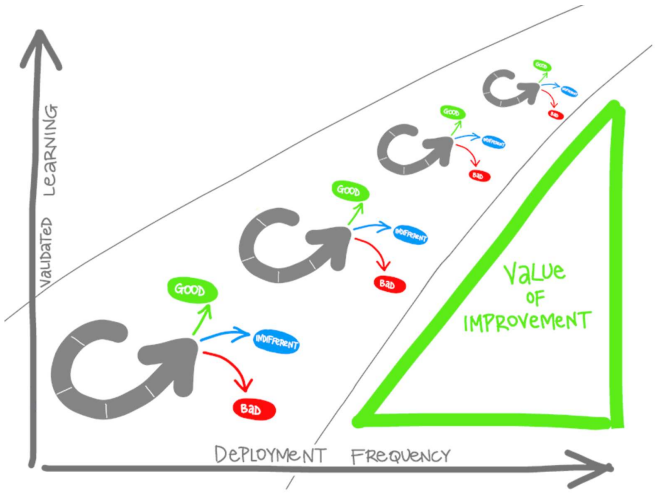


Fig.13 Improvement through retrospectives (Microsoft docs, what is DevOps, 2017)

The more frequently a deployment, the more a team can experiment and the more the chances to learn and improve. With each increment, a team gets to improve and avoid its short comings. This means, the easier the team can deploy and the shorter the team make the cycle and the greater the chance of success as the team spends less time and resource per cycle. (Microsoft docs, what is DevOps, 2017)

5.1 Continuous integration

Continuous integration (CI) refers to a process integrating and testing code of developers in a team to a shared code repository in an automated flow. CI ensures that the integration is successful after a successful build which gets triggered when developers commit a piece of code to the repository. The build fires off an automated trigger on every commit. CI is considered a best practice and a critical part of a DevOps flow as it enables developers work in isolation by enabling swifter merging of conflicts, bugs and decreasing resources in fixing bugs. (Microsoft docs, Continuous integration, 2017)

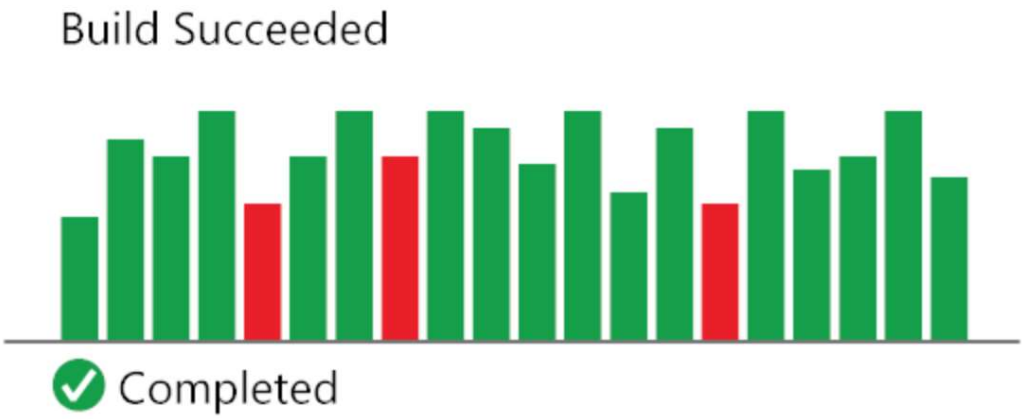


Fig.14 Continuous integration build report widget (Microsoft docs, Continuous integration, 2017)

5.2 Continuous Deployment

In traditional software development, release cycles were a bottle neck. It long a time to deliver a ready code to production. The process had several manual processes that resulted in erroneous deployments. To mitigate this, DevOps strongly advocates the use of continuous delivery (CD). CD is a lean practice with the aim of deploying ready and tested code to production with a strong automated process to production. The automation minimizes errors in releases, exponentially decreases release times and increases Time-to-Market and decreases Time-to-mitigate issues that might arise. (Microsoft docs, Continuous delivery, 2017)

CD is the process of building, testing and running all the needed configuration to deploy a code in a repository to a production environment. CD is triggered after a successful CI on a repository. CD is achieved by creating a release pipeline which, depending on every project, will have a test, quality assurance, canary and production environments. (Microsoft docs, Continuous delivery, 2017)

5.3 Continuous monitoring

Another key process both in DevOps and lean oriented development is continuous monitoring. Monitoring a deployed software gives valuable information on usage and health of an application. One of the main goals of continuous monitoring is to achieve high durable availability. This is achieved when a team can react quickly to an observed bug/issue in a system. A continuous monitoring infrastructure enables the team to be quickly informed of the issue and delivers a detailed view of the situation so that there is enough information to understand the root cause and mitigate it quickly. For instance, Let us assume our product is deployed in Azure with application insights as the monitoring application. Let's also assume we have a performance requirement to the system. In application insights the DevOps team can set an alert when a metric is below par, therefore when there is a performance issue, the team gets an alert immediately along with a wealth of telemetry information in application insights to help mitigate the issue faster. (Microsoft docs, Continuous monitoring, 2017)

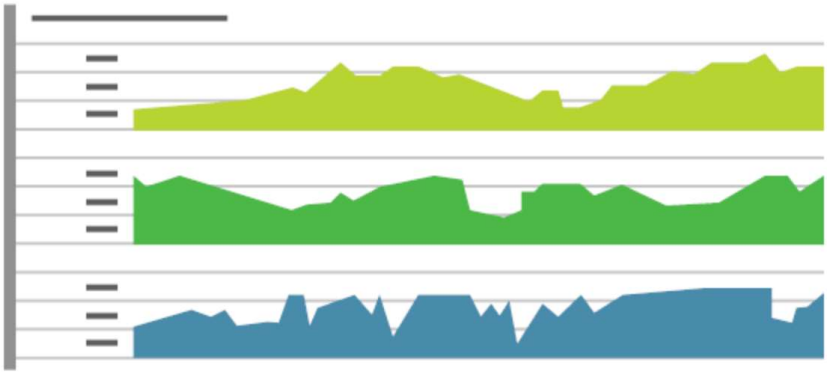


Fig.15 Continuous monitoring (Microsoft docs, Continuous monitoring)

Another usage of continuous monitoring is to enable the flow of usage pattern information of the product. As seen before, the feedback part of the build-measure-learn curve could best be served when augmented with an automated usage data so that the team can further evaluate the worthiness of a feature (s). (Microsoft docs, Continuous monitoring, 2017)

5.4 Azure DevOps

Azure DevOps is a Microsoft cloud service for managing and collaboration of software development. It has all the tool set that enables a team to manage the backlog, create a code repository and version control a repository as well as implement and run a CI and CD pipeline. (Azure DevOps Server Documentation)

In addition to the available features Azure DevOps has smooth integration to a variety of services such as Trello, Slack etc. Azure DevOps is optimal for a low cost, quick on-boarding service that enables starting the development as fast as possible. (Azure DevOps Server Documentation)

It has a free-tier that is good for organizations like EOTC. For this project, the free tier is in use for all tasks such as backlog management, CI, CD, repository and build management. In addition to that, the smooth integration to Azure also is a major advantage for using Azure DevOps. (Azure DevOps Server Documentation)

6 Solution design

An interview with the Parish administrator has revealed current pain points that the Parish is facing. In addition to that, an agreement has been reached on an abstract level what kind of a software would solve the solution. However, we have not designed the product nor the technology it is based on. This section will define the products and some of its requirements. The development uses agile principles and Agile advocates “communication over documentation”. This section, therefore, does not intend to be a detailed plan for the product but rather the initial product backlog. In addition to that, a review of some of the main objectives, requirements, actors and non-functional requirements are as follows:

1. The application is multi-role-based application. That means, the users have different roles and capabilities on the application

2. The application and the data must be hosted on the cloud and with the cheapest means possible. Since the Church does not have adequate on-premises hardware from which to host the application, it will be a web-based cloud hosted application. In addition to that, the maintenance cost of the application should be quite low and so after choosing the technology framework a cost analysis must be made in order to have the cheapest hosting costs as possible.

   The application must be performant. The application will be used from the web, but it is not a necessity that it will only be accessed from Finland alone. It’s a quite common scenario to have a member of the Parish to be in Ethiopia and will have a need to connect to the application. Since, network is not as fast in Finland as it is in Ethiopia, the application needs to perform well in lot bandwidth situations. According broadband speed checker, average results for Ethiopia is 2,8 Mb/s for download and 2,42 Mb/s in contrast to Finland where it is 49,14 Mb/s for download and 22,07Mb/s for the upload. The value in Ethiopia, from experience is quite low from the average on real situations. (Broadband speed checker)

3. The application must be usable. The application must have a good usability. It is a fact that there is not enough resource to do a detailed experience design of the product, basic principles of a good usability design should be part of the production. One of the main reasons, being that the members of the Parish might not all be tech-saavy. In addition to that, the interface should also work using Amharic interface.

4. The application should be usable from mobile. Parish members should be able to engage with the application with a wide range of devices.

   The application should use best practices for implementing security. This includes security for the data while at rest or on transit. It should also have basic guards against common attach scenarios such as SQL injection, role-elevation attacks etc. (OWASP top 10 2017, 6)

   In this regard, OWASP top-10 would serve as a design guideline for security implementation of the application. OWASP (The Open Web Application Security Project) is a

   > “worldwide not-for-profit charitable organization focused on improving the security of software. “
   >
   > (OWASP.Org)

   The OWASP top 10 is a yearly collection of security areas that application should cover in their implementation.

Fig.16 Owasp top 10 2017 (OWASP top 10 2017)

5. The application should have a good monitoring built inside it. One of the main uses in the early phases of the application implementation is to collect feedback on the usage of the application as well as errors produced. But in the long term, having a good application monitoring software is analogous to driving a car with all the gauges working. It helps to identify when problems occur in the application as well as providing insight on the usage and performance of the application.

   The application should be built using API first design mentality. API first design strategy has become a popular approach in recent days especially after the explosion of various smart devices ranging from IoT (Internet-of-things) sensors, to smart watches, to smart cars etc. Traditionally companies build a web application followed by a smart phone app. Only then, will they then build an API that exposes some of their offering to third part developers. API first however, advocates in building the API first and let both internal and

third-party apps utilize it, albeit in a different capacity. (DZone, API First design, 2016)

API first among its many advantages is ensuring parallel development. Traditional development has a dependency problem which means that front-end developers are waiting for backend before they can code against it, but with API first front end developers, by front end we mean all possible client applications, can be developed with a mock API that has the same contract as the real one, while the back-end team is working with the API. (DZone, API First design, 2016)

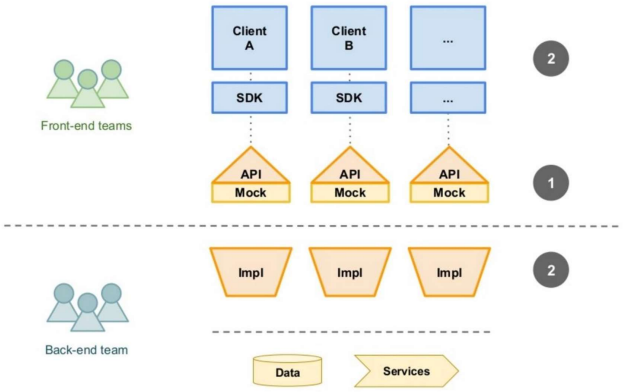


Fig. 16 API first strategy and parallel development. (DZone, API First design, 2016)

The API first design mentality advocates that every functional requirement is exposed via an API all applications are thus considered a consumer of the developed API. With API first being cloud native is a something that comes naturally and is generally good to be adopted. (DZone, API First design, 2016)

With these objectives in mind, the general solution design is an API-driven solution to be built on top of Azure cloud. The application will expose an API which a single page application as the front-end. The logic behind the single page application is due to the performance benefits they bring along to the system.

With the above objectives set, the first challenge that needed to be addressed when starting the solution was to train the Parish council on agile software development to ensure their active participation yields positive results. Additionally, we created a backlog in Azure DevOps upon which we can collaborate on.

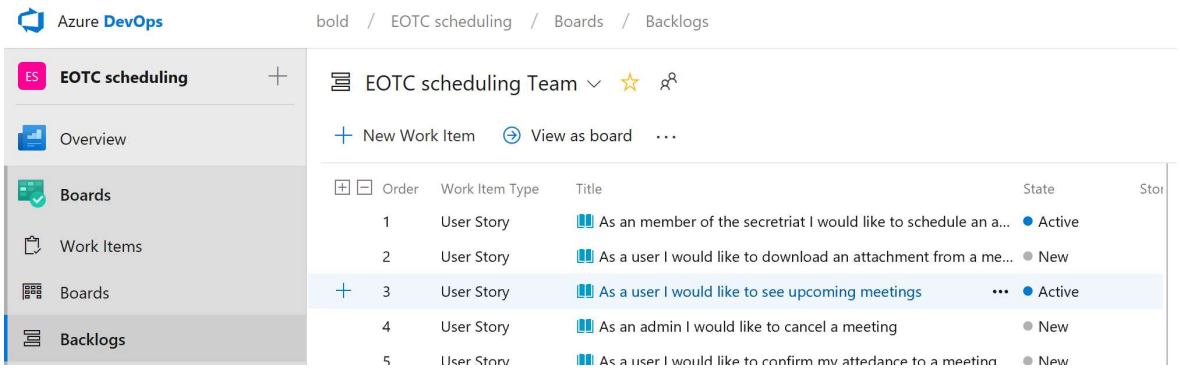


Fig.18 Product backlog

In addition to the backlog, the DevOps pipeline is also built on top of Azure DevOps. The Continuous integration and continuous delivery pipeline are quite straight forward to build in Azure DevOps while Git is the main version control for the code repository.

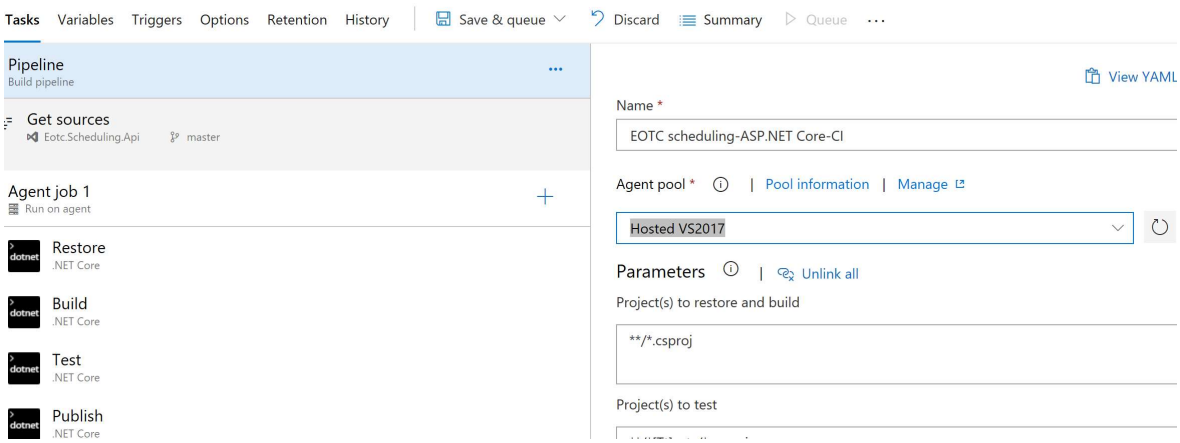


Fig.19 CI configuration for EOTC scheduling API

7 Conclusion

Lean-agile methodologies augmented with a DevOps mindset enable a team achieve success with a relatively less resource and time. Additionally, using these methodologies enforces the customer to be an active part of the development process and thereby decreasing the chance of building something that the customer did not want. The customers early participation ensures that the feedback given is positively impacting the process and therefore the product under development.

One of the things that was noticed during the project was that lean methods were good only for experimentation and that long-term product development should diverge from such an approach. However, using lean across the roadmap is beneficial as assumptions need to be validated regardless the point at which the product has reached.

In the case of the scheduling application, even though the MVP contains a small subset of the application and only to be fully used by the Parish Council in Helsinki, the general roadmap is to push adoption of the software by the general administration. To this, both the feature set and architecture would need to scale, however, the investment to do so will be driven by a validated demand and hence justify the investment. As for now, the development is underway, and soon the EOTC Parish Council will have their first increment.

References

1. Refworld, Immigration and Refugee Board of Canada, 2014: Created on August 4, 2014
   URL: http://www.refworld.org/docid/54c9f9584.html
   Accessed on 17.02.2018

2. Interview with Kesis Lemma Interview (Parish Administrator, EOTC).
   Held on 02.02.2018

3. Lean Canvas, 2012; Author: Ash Maurya
   URL: https://leanstack.com/LeanCanvas.pdf
   Accessed 04.04.2018

4. Google, GSuite
   https://gsuite.google.com/learning-center/products/calendar/get-started/#!/
   Accessed 23.03.2018

5. Calendar app: the ultimate guide; 2016; Author: Rene Ritchie, Created: 26.11.2016
   URL: https://www.imore.com/calendar
   Accessed 14.04.2018

6. Introduction to outlook calendar, Author: Microsoft
   URL:https://support.office.com/en-us/article/introduction-to-the-outlook-calendar-d94c5203-77c7-48ec-90a5-2e2bc10bd6f8
   Accessed 29.03.2018

7. Azeus convene, Author: Convene
   URL:
   https://www.azeusconvene.com/marketplace?utm_medium=BoardManagement&utm_source=capterra#about

Accessed 01.04.2018

8. Contract Zen, Author: Contract Zen
   URL: https://www.contractzen.com/en/meeting-management/
   Accessed 15.06.2018

9. Boardable, Author: Boardable
   URL:
   https://boardable.com/capterra/?utm_source=gartner&utm_medium=PPC&utm_campaign=capterra
   Accessed 07.05.2018

10. Twelve directors portal, Author: Twelve directors portal
    URL: http://unbouncepages.com/loomion-gdm/?utm_source=capterra
    Accessed: 14.04.2018

11. Ubounce. Unique value proposition. Author: Peep Laja
    URL:https://unbounce.com/conversion-glossary/definition/unique-value-proposition/
    Accessed 14.05.2018

12. BPlans. How to create a unique value proposition. Author: Lisa Furgison.
    URL: https://articles.bplans.com/create-value-proposition/
    Accessed 14.05.2018

13. Small business. How to create a unique value proposition. Created on:23.10.2017
    URL:https://smallbusinessbc.ca/article/5-tips-create-unique-value-proposition/
    Accessed 14.05.2018

14. Lean Software Development. Author: Dasari Ravi Kumar

URL:
http://projectperfect.com.au/downloads/Info/info_lean_development.pdf

15. Lean Software Development. Authors: Christof Ebert, Pekka Abrahamsson, Niay Oza. Created on: 2012
URL: http://projectperfect.com.au/downloads/Info/info_lean_development.pdf
Accessed 12.07.2018

16. Planview Leankit. 7 guiding principles of lean development.
URL: https://leankit.com/learn/lean/principles-of-lean-development/
Accessed 12.07.2018

17. Tom and Mary PoppenDieck. Implementing lean software development from concept to cash.2012. Massachusetts, USA.

18. Eric Ries. The Lean Startup. 2011, USA

19. Agile incremental delivery visualized – how to explain Agile and Incremental delivery to anyone. Author: Willem-Jan Ageling. Created. 21.09.2015

20. Merriram-Webster dictionary. Quoted on 22.10.2018
URL: https://www.merriam-webster.com/dictionary/cohort

21. Ash Maurya, 2012. Running Lean. USA

22. Tutorialspoint. SDLC- Waterfall model.
URL: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm
Accessed 12.11.2018

23. XB Software blog. Software Development Life Cycle (SDLC). Waterfall Model. Created: 03.11.2014
URL:https://xbsoftware.com/blog/software-development-life-cycle-waterfall-model/
Accessed 11.09.2018

24. Agile alliance. What is Agile?
URL: https://www.agilealliance.org/agile101/
Accessed 08.08.2018

25. Agile Alliance. Agile manifesto
URL: https://www.agilealliance.org/agile101/the-agile-manifesto/
Accessed 29.08.2018

26. Ken Schwaber and Jeff Sutherland. The Scrum Guide. 2017
URL:https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf

27. Microsoft docs. What is DevOps?
URL: https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops
Accessed: 29.09.2018

28. Microsoft docs. Continuous delivery
URL:https://docs.microsoft.com/en-us/azure/devops/learn/what-is-continuous-delivery
Accessed: 29.09.2018

29. Microsoft docs. Continuous monitoring
URL:https://docs.microsoft.com/en-us/azure/devops/learn/what-is-monitoring
Accessed: 29.09.2018

30. Internet speed checking for various countries.
URL: https://www.broadbandspeedchecker.co.uk
Accessed 24.07.2018

31. OWASP.org. OWASP top 10
URL:https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
Accessed: 22.10.2018

32. DZone. API first design. Created: 2016

URL: https://dzone.com/articles/an-api-first-development-approach-1

Accessed 10.10.2018

33. Brainhub. Differences between Lean, agile and Scrum. Author: Matt Warcholinski
URL: https://brainhub.eu/blog/differences-lean-agile-scrum/
Accessed 01.11.2018

34. Azure DevOps Server Documentation. Author: Microsoft
URL: https://docs.microsoft.com/en-us/tfs/index
Accessed 01.11.2018