TALLINN UNIVERSITY OF TECHNOLOGY

DEPARTMENT OF SOFTWARE SCIENCE

Procurement Of Health Information System For 25 Primary Health Centers In Nigeria

Lab 4 in subject "Software Quality and Standards" (IDY0204)

Authors, student group:

Monika Shrestha
Kehinde Esther Ogundeyi
Chioma Jessica Nkem-Eze

Team ID: Digital Health
Supervisor: Jaak Tepandi
Presented: 12th Dec 2021

TALLINN 2021

Table of Content

10. Proposing system-related metrics and forecast

10.1 Metrics

The most relevant measure we propose for our software is production analytics, which includes mean time between failures (MTBF) and mean time to failures (MTTF). The following discusses why certain metrics were picked and why they are important to our system.

- Both measures assess how well the software performs in a production context.
- It aids in the planning of preventative maintenance.
- We can predict the number of failures that will occur during production.
- Increase pharmacy inventory levels to avoid drug stockouts.
- It allows us to assess the dependability of the computers that will be deployed at the hospital.
- It enables us to determine which computer model would be best suited for use in production when the HIS is deployed.

10.1.1 MTBF Calculation

The Mean Time Between Failures (MTBF) is the average time between repairable production failures. It aids in determining the dependability and availability of HIS software in production. We strive for a higher MTBF value. The following is the equation we're using to calculate MTBF:

MTBF = Total Uptime / # of Failures

The HIS System is operational 24 hours a day, seven days a week, with four outages per month, the first two lasting up to 0.5 hours and the last two lasting up to 1 hour.

Total Working Time: 720 hours which is 24 hours in 1 month

Number of Failures: 4 Outages

Total Failure Time: 2(0.5 Hours) + 2(1 Hours) = 3 Hours

(720 - 3) / 4 = 179

MTBF = 179

This means that when our HIS system is in production, the average time between failures is 179 hours which is a good value.

10.1.2 MTTF (Mean Time To Failure)

Mean Time To Failure (MTTF) is the average time it takes for a non-repairable device or asset, such as a computer, to fail. This enables us to predict how long a computer or asset can be relied on before completely failing, as well as when users should expect to replace or perform routine diagnostics. We want a higher MTTF value since the longer the MTTF, the less money the hospital will have to spend on replacing computers. The following is the equation we're using to calculate MTTF.

MTTF = Total Lifespan of Computers / # of Computers

If the African Development Bank Group (ADBG) delivered computer systems to each of the 25 health clinics in accordance with our goal in Lab 1, and a hospital was given ten computers that were all connected to the same HIS system. Assuming that computer one failed totally at 10,200 hours, computer two failed fully at 9,200 hours, and computer three failed completely at 11,600 hours.

Total Lifespan of the computers: 10,200 + 9,200 + 11,600 = 31,000

Number of computers: 10

31,000 / 10 = 3100

MTTF = 3100 hours

This means that the model of computers provided has an average lifespan of 3,100 hours which means that the computer provided will not last long and ADBG needs to consider providing a long-lasting computer for production.

10.2. Forecast

The significant change we are proposing to the HIS system is an implementation of an online payment platform. This change was initiated by the users to which the software was deployed, as all of the payment for medical services is done manually

10.2.1 System Properties of the Request

Properties	**Justification**
The HIS should offer an online payment platform	Adding the payment platform to the existing HIS will ease accountability of medical services provided by these health centres as well as provide the government with some useful metrics for providing health subsidies.

10.2.2 Cost Forecast

The budget for the request is determined by the procurer. Within the boundaries of the project's established requirements, the development team may not exceed this amount. In the forecast session below, the funding for the request is calculated. Salary for the development team (up to 10 individuals), travel expenditures, equipment, and training fees are all included in the budget. A financial strategy has been devised. The procurer creates an overall budget and development plan. The development team and the procurer must reach an agreement on a comprehensive development strategy that is consistent with the overall strategy. This request has a set budget. Budget modifications are frequently prohibited. All stakeholders must follow the project's budget and development plan to guarantee that the request is finished on time and within budget.

10.2.3 Effort Forecast

The table below estimates the time required to accomplish this request. The developer's team must be approved by the procurer. A project manager (1 person), a software developer (up to 5 individuals), a QA tester (2 people), and technical support (2) shall comprise the unit.

10.2.4 Forecast Method

The Planning Poker method was used to calculate the time required to perform the aforementioned additions. We chose it since it is a frequently used time estimation tool in the agile development area. This method aids in the estimation of software development time and effort. Members of the group rate the complexity of the work by giving points to each member, and the estimations are then collected anonymously so that no member of the team is swayed by the opinions of others. Once the cards are revealed, the estimates are discussed. Planning poker estimation is done using Fibonacci numbers, therefore the numbers 1, 2, 3, 5, and 8 are used. If a task or user narrative takes more than 12 hours, it will be divided into smaller portions.

The table below shows the number of points and comparable time evaluation:

Values	Time	Description
1	1-2 Hours	Very low
2	3-4 Hours	Low
3	5-7 Hours	Medium
5	8-11 Hours	High
8	12 Hours -...	Very high

The HIS's implementation of an online payment platform was divided into smaller phases and tasks, and each step was estimated separately, with the results of each phase's story estimation summed together. When it comes to development, each phase will need to be broken down into more specific

user stories and then evaluated separately. The cards' results were gathered and converted into comparable hours:

Action	Developers	Testers	Analysts	Product	Managers Client	Average (h)
Planning	29	15	19	15	10	17.5
Requirement Gathering	20	26	30	23	12	22.2
Development	300	200	323	150	50	204.6
Deployment and training	28	34	20	15	10	21.4

According to the above table, the development stage has the maximum projected time of 204 hours, with an average anticipated time of about 2 weeks. While the projected time for planning, requirement gathering, deployment, and training was 17, 22, and 21 hours, respectively. It would require around two weeks of labor in man-hours.

11. Implementing ITIL framework

11.1 Framework selection

We implement the ITIL framework because it embraces the latest trends in technologies and service management. It is also the most widely used approach to manage IT services as well as providing a flexible foundation for organizations to integrate digital technology into their business processes. This point is very crucial for us because our main goal is to digitize medical services at various health centers throughout Northern Nigeria.

11.2 Overview of implementation

We implement ITIL v4 approach because it is the most recent edition. Although it includes 34 management practices, we outline below a select number of practices that are relevant to our project:

- Measurement and reporting: To support continual improvement of the HIS, it is necessary to collect data on relevant metrics as well as provide valid assessment of this data. For our system, some of the valid metrics include number of patients, number of doctors available, hardware infrastructure, expansion of health centres etc.

- Service configuration management: The goal of this practice is to ensure that there is access to accurate and dependable information on configuration information and items when it is needed. Although the value of this practice is not direct, it is a foundational item for other practices to work as they should. Therefore, planning for configuration management needs to take place by understanding who needs the information, how it will be used, and the best way for them to obtain it. For our HIS, we find that after the initial setup of the system, the doctors will have no need for configuration. Therefore the plan as suggested by the ITIL framework is to make the information available in a configuration management system (CMS) that will be monitored and maintained by an IT team in charge of the systems

- Software development and management: The purpose of this practice is to ensure that the software meets the requirements of necessary stakeholders, in terms of functionality, reliability, maintainability, compliance, and auditability. Although ITIL 4 does not describe how to develop software, it is added as a practice because it is deemed as an important factor for value creation. Based on previous labs, we find that the software falls short of expectations and as such, would require some additional development work.

- Release management: The purpose of this practice is to plan, schedule and control the movement of releases to the production environment. ITIL defines six approaches for release and deployment management. The one most appropriate for our system is the big branch approach where a new release with instructions for setup is sent to all health centres at the same time. However, if the release has some system dependencies that require manual

checking, then we use the manual approach which would involve an IT team going round to health centres to distribute the release.

- Incident Management: The purpose of this practice is to minimize the negative impact of incidents by restoring normal service operation as quickly as possible. A possible process as described by ITIL involves:
    - Distinguishing between service interruptions and and service requisitions (like password reset)
    - Log all reports as incidents so that their status can be tracked and a historical record maintained
    - Usage of automated resolution tools and self-help information to diagnose and resolve issues
    - Escalated to an appropriate support group which in the case of the HIS will involve development team
    - For major incidents, greater urgency needs to be applied.

12. Retrospective analysis, references, authors

12.1 Retrospective analysis

Working on the labs project was a great way of cementing the knowledge we gained during the classes. It gave us a structured overview of the software engineering process. The following points outline each section and our learnings:

- Lab 1 (Procurement): As software engineers, we are frequently preoccupied with just the development of the software. This lab provided us with a different perspective of approaching the software engineering process. We had to understand the organization and the project, as well as determine the functional and non-functional requirements. We also had to decide which activities would be carried out and when they would be carried out.

- Lab 2 (Development): Although everyone on the team has some development experience, this lab allowed us to try out the test-driven development technique in an open source code base. We had the good fortune of selecting an open source project that fit most of our requirements and had extensive test coverage. We learned a lot by working on the project.

- Lab 3 (Testing): Of all of the labs completed, this was the hardest. This is especially true given that none of us had any prior testing experience. Working in the labs taught us the importance of designing proper tests based on the risk assessment and the requirements. The topics we acquired in this lab will undoubtedly be valuable in our future careers.

- Lab 4 (Maintenance): This lab brought the entire project to an interesting close. We had to think about how our system would hold up over time, and see what new features can be added based on actual need. It was also nice to look back at all the labs done, and see what we have achieved over the last couple of weeks.

One noticeable drawback is this: Although we were lucky to get a good open source project, a number of teams were not able to. This means that they had to make do with creating small one-off projects to fulfil some of the tasks. I think this takes away from the entire experience.

12.2 Presenting the team

The team members for these lab projects were:

- Olga Vovk, Role: procurer. She was responsible for setting the pace of the project by defining the organization and requirements, as well as risk assessment and activities. She left the course after the first lab.
- Chioma Nkem-Eze. Role: developer. She was responsible for implementing the chosen requirements according to the agreed development technique.
- Monika Shrestha. Role: tester. She was responsible for designing and manually implementing tests based on defined requirements and assessed risks.

- Kehinde Ogundeyi. Role: maintainer. She was responsible for determining the performance metrics for the maintenance as well as providing project forecasts.

Each member of the group had a turn at uploading the files on the Moodle forum based on the role distribution.

12.3 References

References to the previous and current lab projects are given below:

1. Organization: AFDB
2. TDD code:Lab 2 Github Repository
3. Software to be procured: Hospital Run Website
4. Software to be procured (License): MIT License
5. Software to be procured (Online Demo): Hospital Run Demo