Faculdade de Engenharia da Universidade Do Porto

Mestrado Integrado em Engenharia Informática e Computação

Engenharia de Software

T1 – History of Software Engineering

Alunos:

João Gama Amaral

João Nuno Ferreira

Pedro Galvão

25 de Outubro, 2018

Structured systems analysis and design method (SSADM)

Introduction:

Structured systems analysis and design method was designed by Learmonth Burchett Management Systems (LBMS) in cooperation with Central Computer Telecommunications Agency (CCTA) in 1980.

SSADM is an unambiguous approach in the form of methodology. It is composed by system surveys, structured analysis, structured designs, hardware studies, implementations and maintenance. Basically, it uses symbols instead of long descriptions, creating a graphic model of the system.

Implementation of the SSADM on software development:

It was developed as a standard for developing British database projects in order to standardise all software projects that were being designed across all government departments. It is an open methodology based on the waterfall model. It has been used by many commercial businesses, consultants, educational establishments and CASE (Computer-aided software engineering) tool developers.

The implementation of this methodology consists of five elements:

1. System Acquisition – The actual purchase of goods and services
2. Programming – Writing of the instructions that will be read and executed by the computers
3. Testing – Tests on various coded pieces of a design and correction of the errors
4. Conversion – The gentle changeover from the old system to the new one
5. Documentation – Writing the way the system and its functions were designed

Values:

- Improve project management and control
- Make more effective use of experienced and inexperienced development staff
- Develop better quality systems
- Make projects resilient to the loss of staff
- Enable projects to be supported by computer-based tools such as computer-aided software engineering systems
- Establish a framework for good communications between participants in a project

Principles:

SSADN follows the waterfall life cycle model starting from the feasibility study to the physical design stage of development.

It divides the cycle of development of the project in steps that are followed in sequence.

It is also characterized by intensive user involvement in the requirements analysis stage.

It revolves around these main modules:

- Logical Data Modelling: This involves the process of identifying, modelling and documenting data as a part of system requirements gathering. The data are classified further into entities and relationships
- Data Flow Modelling: This involves tracking the data flow in an information system. It clearly analyses the processes, data stores, external entities and data movement
- Entity Behaviour Modelling: This involves identifying and documenting the events influencing each entity and the sequence in which these events happen

Its guiding principles are five and can be described as:

1. The development of software is based on a number of phases
2. All these phases are arranged in a sequence beforehand their start
3. The sequence of phases represents the passage through time of the development of software
4. While previous phases are revisited, other phases overlap once new information becomes available
5. As the phases are being followed, the software becomes more useful, but all more complex

Practices:

- Dividing a project into small modules with well-defined objectives
- Diagrammatic representation and other useful modelling techniques
- Outline really well beforehand what is it that is to be built
- Before even starting to write code, consider and determine how to map all the requirements to a software environment
- Build the software according to the designs
- In the end prove that the software delivers exactly what was required

Tools:

SSADM is characterized by having a huge range of tools that can be utilized to make lighter the work burden of the software developers.

Most of the tools that are used are used to give the staff a higher understanding and better perception of the whole system, making it easier to find possible mistakes made during the decision module. Also, the tools must be free of unnecessary details also simplifying the overview of the software system.

Some examples of these tools are:

- The use of symbols instead of narrative descriptions
- Data Flow Diagrams (DFD)
- Data Dictionaries (collection of descriptions of the data objects or items)
- Structured English (pseudocode)
- Decision Trees
- Decision Tables
- Context Diagrams (defines the boundary between part of the system and its environment)
- Entity Relationship Diagram (for example, UML)

Activities:

There are seven core stages on the SSADM, which include.

1. Determining feasibility
2. Investigating the current environment – current systems and problems/requirements
3. Determining business systems options
4. Defining requirements - specification
5. Determining technical system options
6. Creating the logical design
7. Creating the physical design

Each of these phases applies specific techniques and analysis sequences.

Lean Software Development

Introduction

Lean software development was created by Tom and Mary Poppendieck in 2003.

The history of Lean software development methodology began in the middle of the 20th century. At that time, the famous Japanese motor corporation Toyota had big problems with product delivery. Its manufacturing chains were too long and included lots of unnecessary stages. The managers of the company found the solution for this problem, inventing a new project management system originally called Toyota production system. The main idea was to improve the terms of product delivery by waste

elimination allowing the enterprise to make its manufacturing chains shorter and to deliver final products faster. The company came out with a simple and effective definition of waste, which was: anything that didn't impact the functionality of the final product was considered a waste. Later it became popular around the world and changed its name to Lean manufacturing.

Implementation of the lean methodology on software development

In 2003, Lean Methodology was applied to software development. Tom and Mary Poppendieck published their famous book "Lean Software Development". It was a practical guide on the issue of Lean implementation on software engineering. Modern researchers define seven Lean principles, being flexibility of project tasks and respect to the team members the main ones. Lean is and agile methodology. Its projects have iterative structure.

Values

In 2011, the Lean Systems society published a set of values. Those values are:

1. **Accept the human condition:** To develop any product we need people and process. Human behaviour is led by their emotions, thoughts and traits and it is not constant. We must accept and embrace human condition rather than to denying it and expecting robot like behaviour.
2. **Accept that complexity and uncertainty are natural to knowledge work:** In most of the project initial stages, complexity and uncertainly is unknown and is known up to some extent after several studies. But variability can't be anticipated in advance.
3. **Work towards a better economic outcome:** Investors and owners of businesses deserve a ROI (Return on investment). Employees and workers deserve a fair rate of pay performing this work. Customers deserve a food product/service for a fair price.
4. **Seek, embrace and question ideas from a wide range of disciplines:** Teams need to continuously seek for new ideas, brainstorm on it and embrace them to improve quality of process and product. This kind of culture needs to be developed in all disciplines involved in software product development.
5. **A values-based community enhances the speed and depth of positive change:** Share your new ideas/innovations and discuss with a value-based community which in turn gives more improvements as many people practice your ideas in their projects.

Principles:

Eliminate waste: To reduce waste it is critical that development teams are allowed to self-organize and operate in a manner that reflects the work they're trying to accomplish. Finished product must not have waste.

Build in quality: Your process should not allow defects to occur in the first place, but when this happens you should work in such a way that you do a bit of work, validate it, fix any issues found and iterate.

Create knowledge: Planning is useful, but learning is essential. Promote strategies, such as iterative development, helping teams discover what stakeholders really want.

Defer commitment: Support the business effectively through flexible architectures that are change tolerant and by scheduling irreversible decisions to the last possible moment.

Deliver quickly: By limiting the work of a team to its capacity, which is reflected by the team's velocity (number of "points" of functionality which a team delivers each iteration), it can be established a reliable and repeatable flow of work.

Respect people: Sustainable advantage is gained from engaged, thinking people. Enable IT teams, don't control them.

Optimize the whole: Manage programs of interrelated systems so a complete product can be delivered to stakeholders. Always look at the bigger picture.

Practices:

- Cumulative Flow Diagrams
- Visual Controls
- Visual Kanban Systems
- Small Batch Sizes
- Automation
- Kaizen Events
- Daily stand up meetings
- Retrospectives
- Operations Reviews

Roles:

There has been a traditional belief in most businesses about the decision-making in the organization – the managers tell the workers how to do their own job. In this case the roles are turned – the managers are taught how to listen to the developers, so they can explain better what actions might be taken, as well as provide suggestion for improvements. The lean approach follows the Agile Principle "find good people and let them do their own job", encouraging progress, catching errors, and removing impediments, but not micro-managing.

Tools

Lean software development tools are any application that is used by Lean developers to manage their projects or simplify their work is considered a Lean tool. Lean tools are aimed at saving your time for other tasks by automating the process of project management. Lean project management tools are subdivided into two categories: paid applications and free tools. Both can visualize the workflow of Lean team in various forms like charts, tables, diagrams.

Comparison between the two methods

First of all, we can say that Lean Software Development and SSADM have their own advantages and disadvantages. Obviously, the pros outweigh the cons in both cases. In this section we are going to talk about some pros and cons of the two methods and make comparisons.

Software delivery:

Lean software development focus on the elimination of superfluous activity, therefore saving time and money. This enables more functionality to be delivered in a shorter period and empowers the development team in the decision-making process, improving motivation to do the best job possible. On the other hand, SSADM takes a huge amount of time just to analyse the whole project, creating a big gap between the creation of the idea and the delivery of the software system itself.

Perfectionism:

Lean software development motivates its teams to make every product feature perfect. Since SSADM is a user oriented and has a logical design, the probability of rejection from the hardware or from the users itself after the system being implemented is really low. Both focus on making every product perfect.

Teams:

SSADM is a complex method, therefore it may force companies to spend a considerable amount of money in the training of the employees who don’t have the proper education or skills. Lean software development, being heavily team dependent, also needs professionals with high skill level and immense knowledge on the field. Both methods need high skilled professionals to ensure that there are no drawbacks. Learning on the go will only lead to losses.

There are also two cons on the different methods that differ. On the one hand Lean software development excessive flexibility can lead developers to lose focus and procrastinate, while on the other hand SSADM can be way too rigid sometimes pressuring its workers.

Documentation:

One of the pros to the SSADM there is the good documentation provided by the fact this method is well structured, supplying both users and programmers useful information to understand the use of the software. Lean also requires excellent documentation, meaning that any area that is poorly documented can be underdeveloped or developed incorrectly since the process should not allow defects to occur, fixing any issues only after validating work.

Upgrades:

Since SSADM is built on the analysis of data, if some of it changes after the method has already been applied, the system will highly likely be incorrect. As opposed to SSADM, Lean Software Development is great when it comes to additions and upgrades

Lean Software Development Pros and cons in a nutshell:

Pros:

1. Allows for delivery of product early.
2. Lower budget and time requirements.
3. The team is motivated to make every product feature perfect.

Cons:

1. The workability of the team decides success of software development process.
2. The approach is suitable only for highly skilled developers.
3. Excessive flexibility leads developers to lose focus.

SSADM Pros and cons in a nutshell:

Pros:

4. Good Documentation
5. Standardization
6. Modularisation
7. User Oriented
8. Logical Design

Cons:

1. Sometimes can be way too rigid
2. Time-consuming
3. Possibly expensive

References:

SSADM:

https://www.techopedia.com/definition/3983/structured-systems-analysis-and-design-method-ssadm

https://searchsoftwarequality.techtarget.com/definition/SSADM

http://www.informatik.uni-bremen.de/gdpa/methods/m-ssadm.htm#TABULAR

https://www.researchgate.net/publication/266797414?channel=doi&linkId=543c6a720cf24ef33b762745&showFulltext=true

https://slideplayer.com/slide/4332856/

http://adataanalyst.com/it-strategy/structured-systems-analysis-and-design-method/

https://bizfluent.com/list-6781448-advantages-disadvantages-ssadm.html

Lean Software Development:

https://leankit.com/learn/lean/principles-of-lean-development/?fbclid=IwAR1HYMwFhNao457j5zNlTOFXaXAXMSDFXaIuKXNC_wJhZx8YGJYZOv87kiY

http://www.disciplinedagiledelivery.com/lean-principles/?fbclid=IwAR1-farQBGnqhQsUbX5X9yYBUC5DXb8VbZ0fXkkTS5Ms6ZoRoGaBlmJiNMM

https://www.linkedin.com/pulse/lean-software-development-values-principles-balaji?fbclid=IwAR3fYTcBN9uDWm4HpILtrjCc_LuAZQ5RwOaXcGwlIONcz3x05ZP9UxjCM8o

https://hygger.io/blog/lean-tools-overview/

https://bugwolf.com/blog/the-pros-and-cons-of-lean-software-development

https://medium.com/globalluxsoft/5-popular-software-development-models-with-their-pros-and-cons-12a486b569dc

https://acodez.in/12-best-software-development-methodologies-pros-cons/