International Journal of Science and Research (IJSR)
ISSN: 2319-7064
ResearchGate Impact Factor (2018): 0.28 | SJIF (2018): 7.426

Model for the Management of Software Projects Between Universities and Companies Based on CMMI-DEV Ver. 1.3

Lic. Daniela Muñoz Coyotzi[1], PhD. José Juan Hernández Mora[2], Ing. Erika Gonzalez Botello[3]

[1]Tecnológico Nacional de México/ Instituto Tecnológico de Apizaco, División de Estudios de Posgrado e Investigación, Carretera Apizaco Tzompantepec, esquina con Av. Instituto Tecnológico 90300, Tlaxcala, México

[2]Tecnológico Nacional de México/ Instituto Tecnológico de Apizaco, División de Estudios de Posgrado e Investigación, Carretera Apizaco Tzompantepec, esquina con Av. Instituto Tecnológico 90300, Tlaxcala, México

[3]Tecnológico Nacional de México/ Instituto Tecnológico de Apizaco, División de Estudios de Posgrado e Investigación, Carretera Apizaco Tzompantepec, esquina con Av. Instituto Tecnológico 90300, Tlaxcala, México

Abstract: ***The following paper proposes a model for the management of a software development project, based on the maturity Model CMMI-DEV Ver 1.3. The proposed model will be applied by the Instituto Tecnológico de Apizaco which is responsible for the creation and delivery of the computer system to H. Municipality of Apizaco, Tlaxcala; Mexico. In this proposal will apply 13 processes of the 22 that make up the CMMI-DEV Ver 1.3, emphasizing in the description and explanation of the process applied, also presents the how the phases of requirements and analysis were developed. This will provide a basis for the management of future projects that the institute promotes.***

Keywords: software engineering, CMMI-DEV, project management

1. Introduction

Software project management has been one of the biggest challenges for companies engaged in this activity [1].

More and more organizations need to obtain systems to automate their process and optimize time. When there is a need to develop software development projects between universities and companies, new challenges appear; one of the most important is the management of the project because it allows the fulfillment of the objetives established and consequently injects quality in the delivered product [2].

In Mexico the linkage between the business and education sector is booming in the growth of the country, as it promotes and facilitates employment; and specifically in software it allows to implement systems that automate processes that perform manually; however, not begin entities dedicate to the development of software do not have a model to start projects whit this type linkage, on the other hand the educational institutions either. This is why this research is carried out where an existing model is adapted in the management of software projects and consequently deliver a quality product.

2. Background

The development of software in its beginnings did not have standard processes or model that allowed the organized work, to follow a single method of the creation of software systems its difficult because by nature it is a creative process [3] in which one must involve all the persons who will interact with the product obtained, being the administrator or manager of te project the one in charge of verifying that the necessary activities are carried out on the basis of the established one, because it is responsible for the deliveries of the product.

At the same time as the software development technologies advanced so did the models and standards that serve as a guide during the process allowing the final product to be useful and efficient for the people who request it.

The main objective of a project manager is to define the model the way forward, as well as to organize and to follow up the activities that are carried out during the software development; facilitating stakeholders the tools are needed to achieve objetives, optimizing resources [4], [5]; using techniques and procedures appropriate to the requirements and nature of the system.

Most of the activities carried out by the project manager have a lot of relationship whit software engineering and SEVOCAB define it as *"application of a systematic disciplined and quantifiable approach for the development, operation and maintenance of the software"* [11].

CMMI (Capability Maturity Model Integration) is a model that serves as a guide to improve the processes that companies carry out, CMMI-DEV Ver 1.3 is one of the constellations that integrates CMMI and is specifically on the improvement of business processes dedicate to software development.

CMMI-DEV Ver 1.3 contains practices that encompass project management, process management, systems engineering, hardware engineering, software engineering and other support processes used in development and maintenance [2].

Licensed Under Creative Commons Attribution CC BY

Even now CMMI-DEV Ver 1.3 is one of the most widely used models in the world counting whit thousands of companies certified at different levels according to the list that shares the CMMI Institute on its web site [12]. Although the CMMI version 2.0 already exists.

3. Justification

CMMI is not the only model there are the others global standards such as ISO/IEC 15504, 15288, 12207, 29110 and 25000 to mention some or Moprosoft models in Mexico or Competisoft (Improving Processes the Competitiveness of the Small and Medium Industry of software of Ibero-America), although as its name indicates the ISO/IEC are standards thats must be followed punctually to achieve the objetive and in the case of Moprosoft and Competisoft are norms that also must cover all the requested aspects to obtain the certification.

The norms and standards are guidelines that must be followed as stipulated, already in its application they become complex and sometimes hinder the development process. CMMI-DEV Ver. 1.3 is chosen since it is a model in which different tools can be adapted and combined to meet the objetives of the different specific practices that in proposes.

Many of the time you do not get successful results; even whit an administrator and a role model, when you lack them he results obtained can end in incomplete projects or complete disasters.

4. Defining the context

CMMI-DEV Ver 1.3 is a widely used by software development companies all over de world [8], [9] for its flexibility to adapt techniques, methodologies and tools enrich the ideas poses [6], [7]. The constrains to occupy all the processes of CMMI-DEV Ver 1.3 is, because being a linked project both organizations do not hace a workflow, as well as the minimum artifacts needed to define a quality product, which is what this model intended, it also delimits the process that should follow the development.

For the definition of the general process, the processes directly involved whit the life cycle of development are taken, which are intrinsically salted to the categories of support, project management and engineering at levels 2 and 3 of maturity (see Table 1).

CMMI-DEV in its version 1.3 describes 22 processes where each one has a series of good practices, specific and generic that the companies recommend to carry out during the development of a software project. However, in this investigation, 9 have bee ruled out, that is to say, only 13 will be used; since they are the ones that directly intervene in the life cycle of the software development and not I the organizational process that are the discarded ones.

Table 1: CMMI-DEV Ver. 1.3 Processes to use.

#	*Nombre*	*Abrv.*	*Nivel*
1	Configuration Management	CM	2
2	Process and Product Quality Assurance	PPQA	2
3	Project Planning	PP	2
4	Project Monitoring and Control	PMC	2
5	Measurement and Analysis Project	MA	2
6	Product Integration	PI	3
7	Integrated Project Management	IPM	3
8	Requirements Development	RD	3
9	Requirements Management	REQM	3
10	Risk Management	RSKM	3
11	Technical Solution	TS	3
12	Verification	VER	3
13	Validation	VAL	3

4.1. Definition of the general process

To start a general outline is made by dividing those processes that support and that will be carrie out throughout the project and those that are only carried out during the life cycle of the software Figure 1.

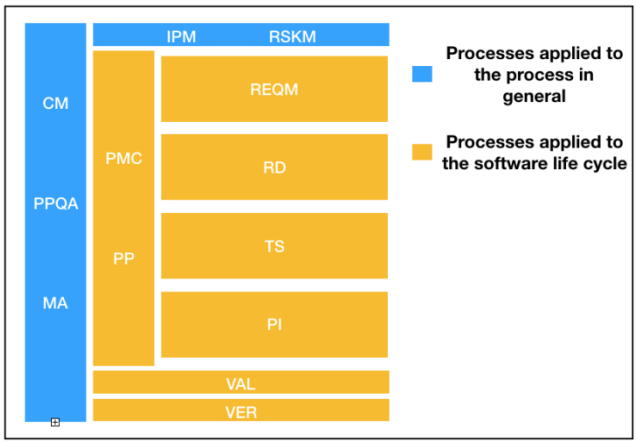

Figure 1: Division of process

The SEVOCAB defines the **software development process** as *"a process by which the needs of the user are translated in to a software product"*, mentions that *"the process implies the conversion of the needs of the user into software requirements, transforming the software requirements in designing, implementing code design, testing the code, and sometimes installing and verifying the software for operational use. These activities can overlap or perform iteratively"* [11], while the **software life cycle** is *"the project specific activity sequence that is created by mapping the activities of a standard into a life cycle model of the software"*.

The definitions may be similar, however the way forward is designed in such a way that the processes at level 2 of the Support and level 3 category in the Project Management category are the basis in the project management; while the level 2 Project Management and Engineering level 3 are the ones that will guide the software life cycle, the core for the mapping of activities to be carried out during the development of the system.

Licensed Under Creative Commons Attribution CC BY

4.2. Working workflow definition

Once defined the processes to occupy were adjusted to determine the workflow, in this adaptation you see the relationship between processes (Figure 2) that are not in the category of support.

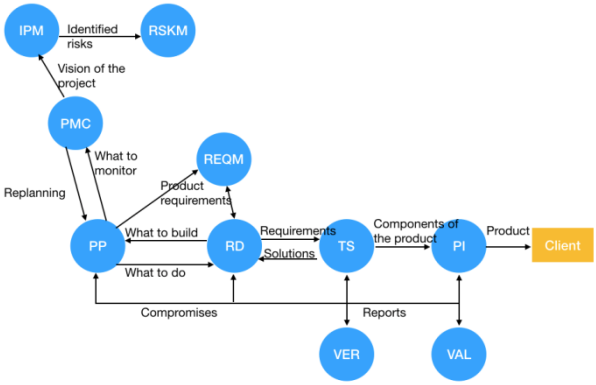

Figure 2: Workflow.

CM, PPQA y MA are part of basic processes that address fundamental support functions that are used by all areas of CMMI-DEV Ver 1.3 and do not consider them within the relationships between others.

Although all Support process areas are based on the others areas for information, Basic Support that are the aforementioned, provide functions that also help implement several generic practices [2]. However it also been attempted to comply whit the specify practices of the model.

4.3. Definition of artifacts

Below are determined the artifacts to use and deliverables that are generated during the development of the software, each artifacts is created from specific goals of the model CMMI-DEV Ver 1.3 and these in turn depart from the methodology chosen for the cycle of life the iterative or agile waterfall model allows each of the phases to be feedback from the other allowing a correlation between all these (Figure 3).

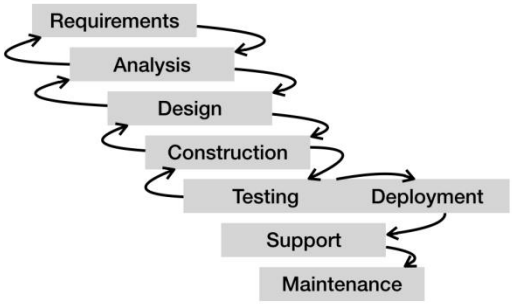

Figure 3: Life cycle phases

The waterfall model is defined as part of the IPM process where the first practice specifies is to "establish the define process of the project"; the iterative waterfall model will allow an agile workflow between the phases that are necessary in the development of the system [13].

For each phase, the minimum artifacts needed to meet the specific practices of each process were defined, Table 2 lists some of the artifacts occupied in the requirements and analysis phase.

Table 2: Artifacts for the Requirements and Analysis phase.

#	*Key and name*
1	FR_REQM_Customer interview
2	FR_IPM_Statement of work
3	FR_RSKM_Risk management
4	FA_CM_Version control
5	FA_RD_Requirements specification
6	FA_RD_Use case diagrams
7	FA_RD_Process diagrams
8	FA_PP_Critical route
9	FA_PP_Project plan

5. Occupied tools

Once the workflow and phases were determined, the methodologies and tools that supported supported each of the processes and artifacts were chosen to fulfill the specific practices proposed by the CMMI-DEV model. The tools chosen are those that are considered to serve the project better and will allow the working team to work well (Table 3).

Table 3: Processes with tools to occupy

Process	*Occupied methodologies and techniques*
CM	Version control (documents, code).
PPQA	Artifacts audits and specifications requirements.
PP	WBS (Work Breakdown Structure), CPM (Critical Path Method).
PMC	Burndown chart.
MA	Process (number of artifacts actual against planned, percentage between tasks delivered in time against delay, forward projection). Product (number of defects injected, number of improvements made).
PI	MVC pattern.
IPM	Waterfall model.
RD	UML (use case analysis, process diagrams).
	UX methodology.
REQM	Interviews, observations, meetings (documents of agreements).
	Specifications of requirements.
RSKM	FMEA technique (Failure Mode and Effect Analysis).
TS	Definition of tools (software for development, programming language, database manager, libraries).
VER	Code test plan (unitary, modular and integration).
	Prototype test plan.
VAL	Prototype validation.
	Requirements specification validation.
	System acceptance.

6. Phases development

Requirements phase

In this first phase, the research and investigation of the needs of the client and the end users was carried out, using the UX methodology [14]; the methodology occupied covers the first three phases. Having a large part of information of the client's needs is create the Statement of Work, a document that consist of the following elements:

- Introduction

Licensed Under Creative Commons Attribution CC BY

- Scope of work
- Period of performance
- Place of the performance
- Work requirements
- Acceptance criteria

Introduction
A description of the project is made, as well as the results that you want to achieve. The detail is important so that all stakeholders are aware of whats is intended to be done.

Scope of work
This section delimits the scope of the project in generals terms, highlighting what will not include the system.

Period of performance
This section defines the schedule of activities for the development of the system, and although there is a delay in the delivery of the product; this must be clear to avoid possible disagreements.

Place of te performance
This step described the place where the system is developed.

Work requirements
This part describes the requirements and deliverables in broad terms for each phase of the project. The commitments that the system development team and the client undertake to meet.

Acceptance criteria
At this point we describe who is responsible for the delivery of the product as well as the characteristics that must have the product to be acceptable.

Once drafted the report of the project is carried out the analysis of risk whit the technique FMEA (Failure Mode and Effect Analysis) that helps in the prevention of possible irrigations. FMEA is an analytical methodology used to ensure that potencial problems have been considered and addressed through the process of product development and process [15]. FMEA is applied to the software lifecycle, potential failures are divided by each phase during the development of the system, this will serve to avoid unforeseen and reduce uncertainty, helping to have a good level in the quality of the delivered product.

The generated format is a table containing the following data:

- Phase: life-cycle phase.
- Activity: action necessary for the fulfillment of the phase.
- Risk identification: the way in which the defined activity could fail.
- Potential cause: indications of weakness and consequences of how the risk can occur, which described in detail in the most concise and complete away.
- Severity: the values associated with the most serious risk effect modes are indicated (Table 4).

Table4: Severity value

#	*Type severity*	*Value*	*Description*
1	High	4	Has an impact that significantly affects the development of the project.
2	Medium	3	Cause minor damage but is important to mitigate it.
3	Low	2	Has not incorregible effect and should be consider.

- Uncertainty: it is probability that some risk occurs (Table 5).
- Result: the result is called the Priority Number at Risk PNR, which is the priority action approach.
- PNR = Severity (S)*Uncertainty (U).
- Containment action: activities currently carried out to control the possible risk.

Table 5: Uncertainty value

#	*Uncertainty*	*Calification*	*Description*
1	High	4	Its has a high probability of occurrence
2	Medi-um	3	Its has a 50% probability of occurrence
3	Low	2	Its has a low probability of occurrence

- Opportunities: describes possible actions that are not performed but can reduce risk.
- Actions to undertake: are the prevention actions to undertake, the intention is to reduce the ranges of severity and uncertainty of risk.

The implantation of RSKM process allows to know those risks that can cause a decontrol in the project and therefore that the system does not meet the objetives, on the other hand when carrying out a prevention analysis it allows the improvement of the process and the product.

The next artifact to be development is for the CM process, where the document and code version control management model is occupied. The purpose of the version control is to easily distinguish the changes between the versions, the search and the tracing, show the relationship between the different version [16].

To do so, you install the Visual SVN program as a server, and Tortoise SVN program as a client.

The structure for the base line repository consist of a folder for each phase, within them are the artifacts generated for the list that is defined at the startup but the codes will be centered in the Construction folder, they will be managed with the help of the framework that is chosen to develop the system, in this case Netbeans.

Version control will allow you to maintain an order in the changes and updates of documents and codes.

The complete phase finally schedule the preliminary dates of beginning and end of echa phase, since it is after the analysis when it comes to plan and breakdown the activities and specific managers whit the task force.

Licensed Under Creative Commons Attribution CC BY

The requirements phase emphasizes understanding and engaging whit the problem to be resolved and the needs of the costumer and the end-users.

Analysis phase
Once a large part of the information is collected, the requirements specification document is created and in parallel the use case diagrams to transform the needs of the costumer into requirements, which are the actions that the system should perform; this part of the RD process.

For the development of the specification of the requirements the format is assembled whit the recommendations of the SDT. 830 of the IEEE in which they explain the different approaches that should consider to better establish the requirements of the software to realise. This documents defines the characteristics of the system, restrictions, the persons concerned and responsible, the functions in details of the system; I mean the functional and non-functional requirements and the required interfaces [17].

The format created consist of the following sections:

Introduction
This explains the overview the document and consists of the following elements.

- Purpose: describes the reason for making this document, delimiting the scope it has.
- Scope: gives the explanation of the system that will be developed, the benefits that will be obtained and the functions, this sections mentions the work statement the serves as a reference.
- Stakeholders: the people involved in the development of the system are explained and the roles of each one are determined, the development team and the responsibilities of the client are defined here.
- Definitions, acronyms and abbreviations: this section describes the acronyms used during the document.
- References: the related documents are written.
- Summary: the contents of the document are explained.

General description
This section describes those factors that directly affect requirements and consists of:

- Product perspective: this item gives and explanation of the what the system should do, the hardware and software needs and peripherals that are occupied, and also clarifies the use that will give the product delivered. You write the tools to occupy for the development as well as the specifications that the hardware must have for the proper functioning of the system.
- Product functions: although the specific requirements are describe in detail in section three at this point the functions are listed in such a way that the stakeholders understand them without being so detailed.
- User characteristics: this section defines the system user profiles.
- Restrictions: refers to those limiting to the development of the system.
- Assumptions and dependencies: those factors that are not limiting but which may affect development are described.

Specifics software requirements
This sections contains all the requirements in sufficient detail in such a way that the development can satisfy them.

- Functionals requirements: these are the fundamental actions that the system will have to make, inputs and outputs and the expected responses.
- Non-functionals requirements: they are those that will allow the system a greater benefit of the resources, and to the users more useful because here describes the characteristics of performance, security, design, usability, and portability with which will count the system.

The a specification of requirements will serve programmers and systems designers to clearly and concisely understand the customer's needs.

In parallel, the use case diagrams that are representations of the behavior of a system or subsystem are performed, this implies all the variations on the normal behavior, and all the expectations that can occur [18].

The use cases are taken from the UML, a use case is made up with the data that will allow to glimpse the operation of the system, and the actors involved in each case. Remember that an actor is the idealization of an external person, a process or something that interacts whit the system.

Table 6 show the filed that make up the description of the use case, as well as the diagrams.

The use cases allow to obtain detailed information of the possible actions of the user and the functions that must have the system, supported in the creation of the prototype as well as in the development of the system; knowing the description of the functions per user causes the programmers to understand the specific requirements; consequently when the requirements are clearer changes are reduce during development progress.

Table6: Formato para un caso de uso

Use case reference	Use case number
Level	Use case priority
Name	Use case name
Actors	The name of the actors who interact in the process
Description	Functions the system perform
Preconditions	Conditions to carry out the use case
Actor'sactions	System response
Actions performed by the actor	Expected result
Possible failure	Actions
Possible exceptions that the user can make	System recovery

The following artifact that was dealt with for analysis were the process diagrams or BPMN (Business Process Modeling Notation), which allow to understand and define the flow that has the information in the system. These diagrams meet multiple purpose in diferente areas of industry in this case

Licensed Under Creative Commons Attribution CC BY

they were used to improve the understanding of the requirements.

These diagrams identify the activities carry out by the user and the reactions that the systems should be have.

The WBS (Work Breakdown Structure) y el CPM (Critical Path Method)are part of the project planning process (PP). Both techniques allow to have a start and end date of each activities and to know the approximate time that leads the development of the project.

Once understood the requirements are broken down the activities using the technique WBS which allows us to first identify the task to be performed by each of the phases, then they are broken down the specific activities to be carried out; in order to determinate the duration of each activity one must take a lot into account the skills of those responsible, and their capacity and experience for this reason is considered of paramount importance to have in common agreement the date of delivery of each activity, for this point it is crucial to know the skills of the programmers, as well as maintaining a good working environment and communication, it is essential that the same developers of the system know their own capacities and limitations.

Whit the activities identified later the document of the work plan is created in which the task that must be executed during the project are recorded, each one with the following data:

a) No.
b) Phase
c) Activity-Task
d) Deliverable
e) Responsible
f) Planned start date
g) Planned end date
h) Planned duration
i) Planned network-days
j) Actual start date
k) Actual end date
l) Actual duration
m) Delivery status: this is defined by the delivery tolerance of the task and has three different status:
   - On time: the delivery of the activity was in the time and form.
   - Delay: the task is delivered than two days from the planned end date.
   - Late: the task is delivered more than two days from the planned end date.
n) Task status: to be aware of the progress control of the task the burn-down chart is used to obtain it, the percentage of the task with status done against backlog is calculated and the task that are in doing, ie those that are in development; those are the three different statuses of the task.

The monitoring of the burn-down chart allows us to have a more realistic idea of of when the cycle of our product will be completed [19].

To calculated the critical path you need to identify the following futures:

1) The list of activities to be fulfilled throughout the project.
2) The duration of each activities.
3) The dependency that exist between the activities.
4) The deliverables.

Using these values it is determined that activities are “critical” i.e. the longest route and which can be delayed without impacting on the project’s calendar [20].

7. Conclusions and recommendations

By following this model adapted for the management of the project we could verify that it is possible to combine various methodologies, it is also observed that having a process to achieve the results is facilitated. Primarily you choose to take the minimum necessary for the development of a software and thus define the process to be executed. As the first point decreases the uncertainty about customer needs due to the use of the UX methodology, on the other hand identified the potential risks that destabilize the project whit the method FMEA and this provides the effect of occurrence. Whit the planning of the critical route it is know which task will have impact on the calendar if it is not fulfilled in time facilitating the decision making, while the artifacts and codes of the project are kept in order whit the control of version.

Although there is no single path in the software development process, the existing models, standards and methodologies are a light to guide this type of project, however the fact of implementing it does not guarantee the success of this, therefore the fulfillment of the objetives not only depends on the implementation of a model but that the professionals in systems enforce it.

CMMI-DEV although it is a very complete model and sometimes you notice the complexity levels of maturity allow to gradually achieve the improvement of the process, thus having small achievements in short, medium and long term.

References

[1] Gacitúa Bustos, R. (2003). Métodos de desarrollo de software: El desafío pendiente de la estandarización. Software Development Methodologies: A Duel Pending for Standardization. *Theoria*, *12* (1), 23-42.
[2] Software Engineering Institute. (2010). CMMI for Development, Version 1.3. Carnegie Mellon University, (November), 482. https://doi.org/CMU/SEI-2010-TR-033 ESC-TR-2010-033
[3] Brooks, F. P., Evans, B., & Hill, A. C. (n.d.). The Mythical Man Monthpp. 3-9, 1995.
[4] Patricia, L., & Guerrero, C. (2016). Gestión en proyectos de software.
[5] Garzás, J. (2013). Gestión ágil de Proyectos de Software, 226.
[6] Rong, G., Zhang, H., & Shao, D. (2016). CMMI guided process improvement for DevOps projects. Proceedings of the International Workshop on Software and Systems

Licensed Under Creative Commons Attribution CC BY

Process - ICSSP '16, 76–85. https://doi.org/10.1145/2904354.2904372
[7] Arauz, G., Morales, M., Oktaba, H., & Ramírez, E. (2016). Integración de Metodos Agiles a una Empresa de Nivel 5 CMMI-DEV: un Caso de Estudio. IEEE Latin America Transactions, 14(3), 1440–1446. https://doi.org/10.1109/TLA.2016.7459632
[8] Jezreel, M., Marcos, G., Mirna, M., & Investigación, C. De. (2016). Organización de las Áreas de Procesos del Nivel II de Level 2 through of its dependencies. Information Systems and Technologies (CISTI), 1. Retrieved from http://ieeexplore.ieee.org/document/7975919/
[9] Nohemí, Y., Meneses, G., Yaline, N., & Padilla, L. (2014). Análisis del estado actual de certificaciones CMMI- DEVver. 1. 3 año 2013 y 2014 , a nivel Mundial y en México, 79, 121–134.
[10] Pressman, R. S. (2010). Ingeniería del software. Un enfoque práctico (Séptima ed). México: McGRAW-HILL INTERAMERICANA EDITORES, S.A. DE C.V.
[11] index @ pascal.computer.org. (n.d.). Retrieved from www.computer.org/sevocab
[12] pars @ sas.cmmiinstitute.com. (n.d.). Retrieved from https://sas.cmmiinstitute.com/pars/pars.aspx
[13] New-Report-Confirms-AgileWaterfall-Mix-Produces-Code @ www.businesswire.com. (n.d.). Retrieved from https://www.businesswire.com/news/home/20140917005036/en/New-Report-Confirms-AgileWaterfall-Mix-Produces-Code
[14] Erika, I., Botello, G., Ramos, M. C. J. R., Juan, M. C. J., Mora, H., Daniela, L. I. C., & Coyotzi, M. (2018). The Case of Use of the UX user Experience Design Methodology Applied to a Face-Recognition Attendance Control System, 7(12), 1203–1207.
[15] Chrysler, Ford Motor, C., & General Motors, C. (2008). Analisis de Modo y Efecto de Falla Potenciales.
[16] Ren, Y., Xing, T., Quan, Q., & Zhao, Y. (2010). Software configuration management of version control study based on baseline. Proceedings - 3rd International Conference on Information Management, Innovation Management and Industrial Engineering, ICIII 2010, 4, 118–121. https://doi.org/10.1109/ICIII.2010.506
[17] IEEE. (1998). IEEE 830-1998: IEEE Recommended Practice for Software Requirements Specifications. IEEE Std 830-1998 (Vol. 1998). https://doi.org/10.1109/IEEESTD.1998.88286
[18] Rumbaugh, J. jacobson, I. &Booch G. (1999). LenguajeUnificadoModelado.
[19] Sink, E. (2011). Stories from my experiences learning scrum. Proceedings - 2011 Agile Conference, Agile 2011, 216–222. https://doi.org/10.1109/AGILE.2011.18
[20] Kelley, J. E., & Rand, R. (1989). Critical path method. World, 1–5.

Author Profile

Daniela Muñoz Coyotzi, has a degree inComputer Science from the Instituto Tecnológico de Apizaco, from 2014. She is currently studying the masters in Computer Systems in Software Engineering from the Instituto Tecnológico de Apizaco.

José Juan Hernández Mora, has a degree in Computer engineer from the Universidad Autónoma de Tlaxcala, from 1994. Master in Computer science at the National Center for Research and Technological Development of the TecNM, 2003 and PhD in Teaching Excellence at the University of Los Ángeles. Research professor at the Tecnológico de Apizaco del TecNM. Teacher of the Master of computer systems of the Instituto Tecnológico de Apizaco.

Erika Gonzalez Botello, has a degree inEngineering in Information and Communication Technologiesfrom the Universidad Tecnológica de Tlaxcala, from 2014. She is currently studying the masters in Computer Systems in Software Engineering from the Instituto Tecnológico de Apizaco.

Licensed Under Creative Commons Attribution CC BY