ARTÍCULO EN ESPAÑOL

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

TEMA:

"La Programación Extrema aplicada al desarrollo del Sistema Informático para la Gestión de Fondos de la Asociación de Profesores de la FICA utilizando MVC"

Autor:

GUZMÁN ANGULO LORENA MAGALI

Director: Ing. Miguel Orquera

Ibarra, 2012

La Programación Extrema aplicada al desarrollo del Sistema Informático para la Gestión de Fondos de la Asociación de Profesores de la FICA utilizando MVC

ENERO 2012

Ibarra - Ecuador

1. METODOLOGÍAS ÁGILES DE DESARROLLO DE SOFTWARE

1.1. INTRODUCCIÓN

Según el Instituto Nacional de Tecnologías de la Comunicación de España, el software juega un papel significativo en la vida de las personas. Desde hace muchos años el principal objetivo en el desarrollo de software ha sido diseñarlo e implementarlo en menos tiempo y que su coste sea menor.

Existen varias metodologías para desarrollar software. Las metodologías que se han utilizado se caracterizan por ser rígidas y su enfoque va dirigido a la documentación, es así que se centran más en el control del proceso, estableciendo rigurosamente las actividades, herramientas y notaciones. A estos métodos se les suele conocer como métodos tradicionales o pesados.

La aplicación de estas metodologías no resultan ser las más adecuadas para muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad.

Es por eso que en el año 2001 un grupo de expertos en software se reunió en Utah-EEUU y formaron el término ágil aplicado al desarrollo de software, en esta reunión se planteó los objetivos y principios que debería tener el desarrollo de software de forma que se realice rápidamente y este de acuerdo a los cambios que tenga el proyecto. Estas nuevas técnicas fueron conocidas como metodologías ágiles.

Las metodologías ágiles se basan en dos aspectos fundamentales, retrasar las decisiones y la planificación adaptativa al cambio de requisitos, y se enfocan en la gente y los resultados.

Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en cortos lapsos de tiempo, llamado una iteración, la cual debe durar de una a cuatro semanas.

Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, pero la meta es tener un demo (sin errores) al final de cada iteración. Al final de cada iteración el equipo vuelve a evaluar las prioridades del proyecto.

1.2. CARACTERÍSTICAS DE LAS METODOLOGÍAS ÁGILES.

Para definir las características que tienen las metodologías ágiles de desarrollo de software se creó el documento de la filosofía ágil.

En la reunión efectuada en el 2001 se creó The Agile Alliance, la cual es una organización sin ánimo de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos.

En la reunión se encontraban 17 expertos dentro de la industria del software, entre ellos creadores e impulsores de metodologías de software, propusieron una alternativa al desarrollo de software diferente a las metodologías tradicionales, el punto de partida fue el Manifiesto Ágil, un documento que resume la filosofía ágil.

Según el Manifiesto se valora:

- *Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.*
- *Desarrollar software que funciona más que conseguir una buena documentación.*
- *La colaboración con el cliente más que la negociación de un contrato.*
- *Responder a los cambios más que seguir estrictamente un plan.*

Los valores descritos dan lugar a los principios del manifiesto los cuales son características de un proceso ágil de un tradicional.

Tenemos doce principios, en los cuales los dos primeros principios son generales y resumen gran parte del espíritu ágil. Los demás se relacionan con el proceso a seguir y con el equipo de desarrollo, en cuanto: metas a seguir y organización del mismo.

1.3. PRINCIPIOS DE LAS METODOLOGÍAS ÁGILES

A continuación se describen los siguientes principios:

1. La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
2. Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
3. Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
4. La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
5. Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
6. El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
7. El software que funciona es la medida principal de progreso.
8. Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
9. La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
10. La simplicidad es esencial.
11. Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.
12. En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

1.4. CICLO DE VIDA

La metodología ágil promueve iteraciones a lo largo del ciclo de vida del proyecto; conocidas como timeboxes, se realiza de forma colaborativa mediante la organización de los equipos que producen software de alta calidad con un coste efectivo y en el tiempo apropiado que cumple con las necesidades cambiantes de las personas involucradas en el negocio.

Las iteraciones son lapsos de tiempo que van de una a cuatros semanas. Cada iteración del ciclo de vida incluye:

- Planificación,
- Análisis de requerimientos,
- Diseño,
- Codificación,
- Revisión y
- Documentación.

1.5. TIPOS DE METODOLOGÍAS ÁGILES

Existen varias metodologías ágiles,

Ahora bien, las metodologías más populares son:

- XP.
- SCRUM.
- ASD.
- CRYSTAL CLEAR.
- AUP.
- LSD.

2. METODOLOGÍAS TRADICIONALES DE DESARROLLO DE SOFTWARE

Inicialmente el desarrollo de software no tenía un proceso formal, es así que se adaptaron metodologías existentes de otras áreas. Esta adaptación dividió el desarrollo de software en etapas secuenciales que en cierta manera solucionó la necesidad del área de software.

Así surgen las metodologías de desarrollo llamadas Tradicionales o Pesadas, las cuales tienen mayor énfasis en la planificación y control del proyecto.

Las metodologías tradicionales se centran en una disciplina de trabajo sobre el proceso de desarrollo del software, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada, para así conseguir software más eficiente. En la planificación se detalla todo lo requerido, una vez hecho esto, comienza el ciclo de desarrollo del producto de software, ya que si luego se desea implementar un cambio su coste es alto.

Las metodologías tradicionales centran su atención en llevar una documentación exhaustiva de todo el proyecto, todo esto definido en la fase inicial del desarrollo del software.

Su premisa fundamental argumenta que hay suficiente planeación y administración, por lo que el resultado puede predecirse y así mismo pueden evitarse los riesgos.

Uno de los inconvenientes de las metodologías tradicionales es que no se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden predecirse o pueden variar.

Entonces se dice que las metodologías tradicionales tienen las siguientes características:

- Se basa en documentos.
- Se elaboran definiciones Flujo Trabajo.
- Existen muchos roles diferentes.
- Muchos puntos de control.
- Tiene alto sobrecoste de gestión.
- Mucha burocracia.

2.1. METODOLOGÍA DE PROGRAMACIÓN EXTREMA XP.

Un proceso ligero puede ser definido como… *El mínimo conjunto de actividades y elementos que deben ser incluidos en el proceso de desarrollo de Software para asegurar un buen resultado para todas las partes interesadas.*

2.2. POR QUE UTILIZAR XP?

Muchas veces nos preguntaremos cuando usar XP. Los proyectos con requerimientos dinámicos son perfectos para XP. Estos proyectos experimentarán grandes éxitos y productividad del desarrollador.

XP es un nuevo concepto refrescante. XP tiene éxito porque da énfasis al involucramiento del cliente y promueve el trabajo del equipo.

El aspecto más sorprendente de XP son sus reglas simples y prácticas. Al inicio parecen torpes y quizás incluso ingenuo, pero pronto se vuelve un cambio bienvenido. A los clientes les gusta estar en el proceso de desarrollo de software y contribuir con los diseñadores activamente sin tener en cuenta el nivel de experiencia.

2.3. DEFINICIÓN DE PROGRAMACIÓN EXTREMA O EXTREME PROGRAMMING XP

A continuación tenemos algunas definiciones para la metodología XP dadas por varios autores.

Según Kent Beck quien es considerado como el padre de la metodología XP, menciona que es un proceso ligero, de

bajo riesgo, flexible, predecible, científico y divertido de desarrollar software.

Según Wikipedia.
La programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

De acuerdo a Programacionextrema.org
La Programación Extrema es uno de los llamados procesos o metodologías ágiles de desarrollo de software. Consiste en un conjunto de prácticas que a lo largo de los años han demostrado ser las *mejores* del desarrollo de software, llevadas al extremo, fundamentadas en un conjunto de valores".

Deigote's Blog menciona:
La programación extrema es una metodología de ingeniería de software para el desarrollo del mismo, que hace énfasis en los siguientes aspectos: satisfacción del cliente y trabajo en equipo.

2.4. CONTEXTO XP

La metodología XP tiene el siguiente contexto.

- Cliente bien definido y en colaboración constante.
- Los requisitos pueden y van a cambiar (volátiles).
- Reduce los tiempos de desarrollo manteniendo la calidad.
- Desarrollo incremental y continuo para responder a los cambios.
- Grupo pequeño y muy integrado

2.5. ARTEFACTOS DE LA METODOLOGÍA XP

Entre los artefactos de la metodología XP encontramos:

- Historias de Usuario.
- Tareas de Ingeniería.
- Pruebas de Aceptación.
- Tarjetas CRC.

2.6. ROLES XP

De acuerdo Beck tenemos los siguientes roles:

- Programador.
- Cliente.
- Encargado de pruebas (tester).

- Encargado de seguimiento (tracker).
- Entrenador (coach).
- Consultor.
- Gestor (bigboss).

2.7. CICLO DE VIDA DE XP

El ciclo de vida ideal de XP consiste en las siguientes fases:

1. Exploración.
2. Planificación de la entrega.
3. Iteraciones.
4. Producción.
5. Mantenimiento y
6. Muerte del proyecto.

2.8. VALORES DE LA PROGRAMACIÓN EXTREMA

Los principios originales de la programación extrema son:

- Simplicidad:
- Comunicación:
- Retroalimentación (feedback):
- Coraje o valentía.
- Respeto

5.1. BUENAS PRÁCTICAS PARA LA APLICACIÓN EXITOSA DE LA METODOLOGÍA XP

La mayoría de las prácticas propuestas por XP no son nuevas sino que ya habían sido propuestas en ingeniería del software e incluso demostrado su valor en la práctica. El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo.

XP propone las siguientes prácticas para ayudar en el desarrollo de software.

- El juego de la planificación
- Entregas pequeñas
- Metáfora.
- Diseño simple
- Pruebas.
- Refactorización (refactoring)
- Programación en parejas
- Propiedad colectiva del código
- Integración continua
- 40 horas por semana
- Cliente in-situ
- Estándares de programación.

2.9. ACTIVIDADES DE LA METODOLOGÍA XP

- Codificar
- Probar
- Escuchar
- Diseñar

2.10. FASES DE LA METODOLOGÍA XP

- Planificación
- Diseño
- Desarrollo
- Pruebas

3. ARQUITECTURA MVC

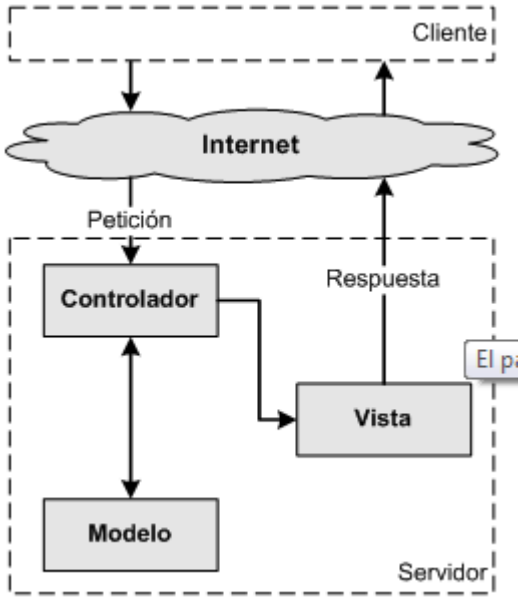
3.1. DEFINICIÓN

Es un patrón de diseño de arquitectura de software principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de conceptos para que el desarrollo esté estructurado de una mejor manera, decrementando la duplicación de código, facilitando la programación en diferentes capas de manera paralela e independiente, y permitiendo que la aplicación sea más extensible. MVC es el patrón de diseño arquitectural para la capa de presentación.

3.2. CARACTERÍSTICAS DE MVC

Es asi que tenemos: Modelo, Vista y Controlador.

- Los datos (que responden a unos modelos)
- La interfaz gráfica (la vista, como se van a representar esos datos)
- La lógica (el controlador, lo que se hace y como se hace).



4. TECNOLOGÍAS A USAR

4.1. BASE DE DATOS ORACLE.

Oracle es un sistema de gestión de base de datos objeto-relacional (o ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation.

El modelo relacional (de un modo sencillo) consiste en utilizar **tablas bidimensionales** para almacenar la información.

Consta de tres elementos básicos:

- Tablas
- Conjunto de operadores para manipular esas tablas
- Reglas de integridad

4.2. JDeveloper

JDeveloper es un entorno de desarrollo integrado para programar aplicaciones o Applets en Java, es desarrollado por Oracle Corporation para los lenguajes Java, HTML, XML, SQL, PL/SQL, Javascript, PHP, Oracle ADF, UML y otros.

JDeveloper es de la Oracle y la diferencia con otros IDEs para programar Java, es que tiene integrado comunicación con drivers nativos para Base de Datos Oracle.

JDeveloper va más allá de ser un IDE para crear aplicaciones Java y lo que promueve es el concepto de IDE empresarial.

Para el desarrollador el ambiente, los menús, etc, son los mismos; no se requiere de utilizar otras herramientas, el IDE se ajusta de acuerdo al tipo de tecnologías a utilizar.

4.3. OC4J

The Oracle Application Server Containersfor J2EE (OC4J) standalone provee una distribución completa de entorno del servidor J2EE, se encuentra distribuido como un simple archivo .zip.

Esta distribución incluye un servidor HTTPs, los servicios requeridos de J2EE, y las capacidades de Web Services, todos se ejecutan desde un proceso de Java.

5. ARQUITECTURA DEL SISTEMA DE GESTIÓN DE AHORRO PARA LA ASOCIACIÓN DE PROFESORES DE FICA.

El Sistema de Gestión de Ahorro para la Asociación de Profesores de la FICA, se encuentra desarrollado en la tecnología JSF JavaServer Faces que constituye un marco de trabajo (*framework*) de interfaces de usuario del lado de servidor para aplicaciones web basadas en tecnología Java y en el patrón MVC (Modelo Vista Controlador).

El lenguaje utilizado es Jdeveloper el cual trabaja muy bien con JSF, se utiliza páginas JSP para las funciones de la Vista, Faces Servlet para cumplir las

funciones del Controlador y para la parte del Modelo se hace uso de EJB´s (Enterprise Java Beans)
La Base de Datos utilizada es Oracle 10g, la cual se interconecta muy bien con Jdeveloper.

6. CONCLUSIONES

- La metodología XP es una muy buena alternativa para el desarrollo de software, ya que da la posibilidad de ir definiendo más requerimientos a medida que va avanzando el proyecto, permitiéndole que sea escalable.
- Los desarrolladores al tener un buen ambiente de trabajo, desarrollan software de buena calidad en menor tiempo.
- Al trabajar con el cliente durante todo el desarrollo de software, la definición de los requerimientos son más entendibles y por ende mucho más fácil el desarrollo.
- Las historias de usuario son una muy buena estrategia para definir claramente que es lo que desea que realice el sistema.
- Las pruebas de aceptación permiten afianzar lo que especificó el cliente en las user histories, ya que se escriben como debe funcionar el sistema ante el ingreso de datos erróneos o bien cuando el ingreso de los datos son exitosos.
- Existen muy pocas fuentes de información acerca de los pasos a seguir para la aplicación de la metodología XP en el desarrollo de software.
- La descripción de los artefactos de la metodología XP es corta en la mayoría de las fuentes de información en Internet, únicamente hacen mención a éstos.

- La herramienta JDeveloper utilizada para el desarrollo del sistema SIGFAP, no fue la más adecuada para aplicar la metodología XP, que propone el desarrollo en menor tiempo, ya que al no disponer de mucha documentación de su utilización, se presentaron errores que hicieron difícil el correcto avance del desarrollo del sistema.

7. RECOMENDACIONES

- Para el desarrollo de sistemas utilizando la metodología XP, se recomienda utilizar otro IDE de Java, puede utilizar Netbeans o Eclipse y si es necesario utilizar otra Base de Datos que sea mucho más compatible con estos lenguajes.
- Aplicar la metodología XP para el desarrollo de software, ya que los artefactos que utiliza para describir el funcionamiento del sistema son sencillos y fáciles de seguir.
- Se aplique el trabajo en Parejas en el desarrollo de software, ya que el conocimiento que uno de los dos tiene ayudará al otro para que vaya ganando experiencia, y el menos experimentado ayuda al otro encontrando soluciones óptimas y el proyecto avance más rápido.

8. BIBLIOGRAFÍA

Metodologías de Desarrollo de Software Ágiles.

- http://es.wikipedia.org/wiki/Desarrollo_de_software
- http://www.cenitec.com.mx/Manifiesto.pdf
- http://seccperu.org/files/Metodologias%20Agiles.pdf
- http://www.willydev.net/descargas/prev/TodoAgil.Pdf
- Metodologiasagiles.pdf

- agil.pdf.- Instituto Nacional de Tecnologías de la Comunicación
- schenone-tesisdegradoingenieriainformatica.pdf
- http://tratandodeentenderlo.blogspot.com/2010_06_01_archive.html
- http://en.wikipedia.org/wiki/Agile_software_development
- http://www.ambysoft.com/unifiedprocess/agileUP.html
- http://en.wikipedia.org/wiki/Agile_software_development
- http://everac99.spaces.live.com/blog/cns!B2296C467C188917!659.entry.htm
- http://blog.tercerplaneta.com/2007/05/lean-product-develepment.html
- Articulo Tecnico Metodologias Desarrollo.doc

Metodología de Programación Extrema

- Programacionextrema.org
- Deigote's Blog
- http://procesosdesoftware.wikispaces.com/METODOLOGIA+XP
- http://oness.sourceforge.net/proyecto/html/ch05s02.html
- ftp://190.5.199.3/jjurado/Mejora%20de%20Procesos/XP_6JAI.pdf
- http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf
- http://elblogdelfrasco.blogspot.com/2008_07_01_archive.html
- http://users.dsic.upv.es/asignaturas/facultad/lsi/ejemploxp/index.html
- http://www.ogis-ri.co.jp/otc/swec/process/am-res/am/artifacts/crcModel.html
- www.programacionextrema.org
- Critica MA.pdf
- Xtremeprogramming.org
- http://synergyca.net/index.php?p=1_9_Metodologias

Arquitectura MVC y Tecnología a utilizar

- http://en.juantxu.net/doku.php/jee/3?do=index
- capitulo2.pdf
- http://feeds.feedburner.com/librosweb_es
- http://es.wikipedia.org/w/index.php?
- http://www.orasite.com/index.php?
- http://creativecommons.org/licenses/by-nc-sa/2.0/legalcode
- http://www.oracle.com/index.html
- http://es.debugmodeon.com/debate/cuales-son-las-ventajas-de-jdeveloper
- http://www.programacion.com/articulo/integracion_de_jsf_spring_e_hibernate_para_crear_una_aplicacion_web_del_mundo_real_307/3
- Jsf.pdf

The Extreme Programming applied by development Informatics system to funds management to Teacher´s Association FICA using MVC.

1. AGILE METHODOLOGIES OF SOFTWARE DEVELOPMENT.

1.1. INTRODUCTION

According to National Institute of Communication Technologies of Spain, the software plays a significant role in the people´s live. For many years the main objective in software development has been designed and implemented in less time and less cost.

There are several methodologies to develop software. The methodologies are rigid and its focus is directed to file, so you are more focused on process control, establishing rigorously the activities, tools and notations.

These methods are generally known as traditional or heavy. The application of these methodologies are not the most appropriate for many projects where the environment changes constantly, and where required to drastically reduce development time but maintaining high quality.

In 2001 a group of software experts met in Utah, USA and formed the term applied to agile software development, in this meeting determinate the objectives and principles of the software that should have the software development, so that rapidly and this is done according to the changes of project. These new techniques were known as agile methodologies.

The Agile methodologies are based on two fundamental aspects; delay decisions and planning adapt to changing requirements, and focus on people and results.

There are many agile development methods, its minimizes the developing software risk in short time´s periods, called “iteration”, which should last from one to four weeks.

An iteration should not add much functionality to justify the product market release, but the goal is to have a demo (no errors) at the end of each iteration. At the

end of each iteration the team reevaluates projects priorities.

1.2. Characteristics of agile methodologies.

To define the characteristics that have the agile methodologies of software development so created the agile philosophy document.

In the meeting of 2001 formed the agile alliance, which is organization nonprofit spirit, dedicated to promoting the concepts of agile software development and help to organizations to adopt these concepts.

The meeting included 17 experts in the software industry, including creators and software methodologies drivers, proposed an alternative to software development different from traditional methodologies; the starting point was the Agile Manifesto, a document that summarizes agile philosophy.

The Manifest measured.

- The individual and the development team interactions, about the process and tool.
- Developed software that runs more than getting good documentation.
- Collaboration whit the client rather than negotiating a contract.
- Respond to changes most that strictly follow a plan.

The values described give rise to manifest principles of which are characteristics of a traditional agile process.

We have twelve principles, in which the first two principles are general and summarizes much of the spirit agile. The others related to the process to follow and whit the development team, as: goals to follow and organized.

1.3. Principles of agile methodologies.

Here are the following principles:

1) The priority is to satisfy the customer through early and continuous software delivery that will supply a value.
2) Welcome to change. The changes are captured so that the client has a competitive advantage.
3) Deliver software frequently that working, from a couple of weeks to a couple of months, whit the shortest possible time interval between deliveries.

4) Business people and developers must work together throughout project.
5) Building the project around motivated individuals. Give them the environment and support they need and trust them to get job finally.
6) Face to face dialogue is the most efficient and effective method for communicating information in a development team.
7) The software running is the primary measure of progress.
8) The agile processes promote a sustainable development. The promoters, developers and users should be able to maintain a constant peace.
9) The continuous attention to technical quality and good design enhances agility.
10) The simplicity is essential.
11) The best architectures, requirements and designs emerge from organized teams self.
12) At regular intervals, the team reflects on how to become more effective, and adjusts its behavior.

1.4. LIFE CYCLE

the agile methodology promotes iterations throughout the project life cycle, known as timeboxes, is done in form collaborative by organizing teams that produce software of high quality whit a cost effective and the appropriate time that meets the needs changing the people involved in the business.

The iterations are time periods ranging from one to four weeks. Each iteration of the life cycle includes:

- Planning.
- Requirements analysis.
- Design
- Review
- Documentation.

1.5. Agile methodologies types.

There are several agile methodologies. The most popular methods are:

- XP.
- SCRUM.
- ASD.
- CRYSTAL CLEAR.
- AUP.
- LSD.

2. TRADITIONAL METHODOLOGIES SOFTWARE DEVELOPMENT

Initially the software development did not have a formal process, so that existing methodologies adapted from other areas. This adaptation software development divided into sequential stages that in a way solved the need for the software area.

So arise the called traditional or heavy development methodologies, which have greater emphasis on planning and project control.

Traditional methodologies focus on a discipline of work on the software development process by a strict definition of roles, activities, artifacts, tools and notations for modeling and detailed documentation, to as to achieve more efficient software. In planning all required details, once this begins the cycle of software development product, because if you then want to implement a change its cost is high.

Traditional methodologies focus on leading a comprehensive documentation of the project, all defined in the initial phase of software development.

- The fundamental premise argues that there is sufficient planning and administration, so the result can be predicted and thus can avoid the same risks.
- Once the drawbacks of traditional methodologies do not adequately adapt to changes, so methods are not suitable when working in an environment, where requirements cannot be predicted or may vary.
- The traditional methods have the following characteristics:
- Based on documents.
- Work flow definitions are development.
- There are many different roles.
- Many control points.
- It has high overhead of management.
- Too much bureaucracy.

3. Extreme Programming methodology.

A light weight process can be defined as...The minimum sets of activities and elements that must be included in the software development process to ensure a

good outcome for all stakeholders.

3.1. WHY USE XP?

Often we ask when to use XP. The projects with dynamic requirements are perfect for XP. These projects will experience great success and developer productivity.

XP is a new concept refreshing. XP is successful because it emphasizes the customer involvement and promotes team work.

The most surprising aspect of XP is the simple rules and practices. At the beginning seem awkward and perhaps even naive, but soon becomes a welcome change. Customers like to be in the software development process and contribute actively designers regard less of experience level.

3.2. EXTREME PROGRAMMING DEFINITION.

There are some definitions for the XP methodology given by several authors.

Kent Beck who is regarded as the father of XP methodology, say that it is a lightweight process, low risk, flexible, predictable, scientific and fun to develop software.

Wikipedia.

Extreme programming is different from traditional methodologies in that put more emphasis on adaptability than on predictability. The extreme programming can be considered as adoption of improved methodologies of development according to what is intended to execute the project, and apply it dynamically during the software cycle life.

Programacionextrema.org

Extreme programming is called methodologies and processes agile software development. It consist of a set of practices over the years have proven to be the best in the software development, taken to extremes, based on a values set.

Deigote´s Blog

Extreme programming is software engineering methodology by developments its, it emphasizes the following areas: customer satisfaction and teamwork.

3.3. XP Context.

The XP methodology has the following context:

- Customer focused and in constant collaboration.

- The requirements can and will change (volatile)
- Reduce development time while maintaining quality.
- Incremental and continue development to respond to changes.
- Small group and highly integrated.

3.4. ARTIFACTS OF THE XP METHODOLOGY

The artifacts of the XP methodology are:

- User Stories.
- Engineering task.
- Acceptance testing.
- CRC cards.

3.5. XP ROLES

According Beck there are the following roles:

- Programmer.
- Client
- Tester.
- Tracker.
- Coach.
- Consultant
- Bigboss.

3.6. XP LIFE CYCLE

The ideal XP life cycle consists of the following phases:

- Exploration
- Release planning.
- Iterations
- Production
- Maintenance
- Project death.

3.7. XTREME PROGRAMMING VALUES

The original principles of extreme programming are:

- Simplicity
- Communication
- Feedback
- Courage and bravery
- Respect.

3.8. GOOD PRACTICES TO THE SUCCESSFUL APLICATION OF XP METHODOLOGY.

The practices proposals for XP are not new this had been proposed in software engineering and even proved the value in practice. The merit of XP is to integrate them in an effective and complement

them whit other ideas from the business perspective, human values and teamwork. XP proposes the following practices to help in the software development.

- The planning game
- Deliveries small
- Metaphor
- Simple design
- Testing
- Refactoring.
- Pairs programming.
- Collective ownership of code
- 40 hours per week
- Customer in situ
- Programming standards.

3.9. ACTIVITIES OF XP METHODOLOGY

- Code
- Test
- Listen
- Design.

3.10. PHASES OF THE XP METHODOLOGY

- Planning
- Design
- Development
- Testing

4. MVC Architecture

4.1. DEFINITION

Is a pattern of software architecture design in applications that handle mainly large amount of data and complex transactions which require better separation of concerns that development is structured in a better way, decreasing code duplication, facilitating programming in different layers of parallel and independent way, allowing the application more extensible. MVC is the architectural design pattern for the presentation layer.

4.2. MVC Characteristics

Thus we have: Model, View and Controller.

- Data (responding to some models)
- The graphical interface(the view, as to render such data)
- The logic (the controller, which does and howit is done).

5. TECHNOLOGY TO USE

5.1. ORACLE DATABASE

Oracle is a management system database object-relational (ORDBMS Object

Relational Data Base Management System), developed by Oracle Corporation.

The relational model (a simple way) is to use two-dimensional tables to store information.

Oracle consists of three basic elements.

Tables

Operators set to manipulate these tables.

Integrity rules.

5.2. Jdeveloper

Jdeveloper is an integrated development environment for programming or Applets in Java, is developed by Oracle Corporation for Java, HTML, XML, SQL, PL/SQL, Javascript, PHP, Oracle ADF, UML and others.

Oracle Jdeveloper is different to others IDE´s for Java, this have integrates communication with native drivers for Oracle Database.

Jdeveloper is most that JavaIDE to create applications and promoting the concept business IDE.

For the developer environment, menus, etc, are the same, not required to use others tools, the IDE is set according to the type of technology to use.

5.3. OC4J

The Oracle Application Server Containersfor J2EE (OC4J) standalone provides a complete distribution J2EE server environment, is distributed as a single file .zip. these release includes an HTTPS server, the required services J2EE and Web Services capabilities, all execute from Java process.

6. ARCHITECTURE OF MANAGEMENT OF SAVINGS ASSOCIATION OF TEACHER OF FICA SYSTEM.

The management of savings association of teacher of FICA system. Is developed in JSF Java Server Faces technology that provides a user interfaces framework to server to applications based on Java technology and MVC pattern (Model, View, and Controller).

The language used is Jdeveloper which works fine with JSF, JSP pages are used for the functions of the View, Faces Servlet to perform the functions of the Controller and the Model makes use of EJB´s (Enterprise Java Beans).

The Database used is Oracle 10g, which connects well with Jdeveloper

7. CONCLUSIONS

- The XP methodology is very good alternative for software development because it gives de possibility to defining requirements as the project progresses, allowing it to be scalable.
- Developers to have a good working environment, develop quality software in less time.
- Working with the customer throughout the software development, the definitions of the requirements are more understandable and much easier to develop.
- User stories are a very good strategy to define clearly what you want to make the system.
- Acceptance testing allows reinforce what the client specified in the user histories, as written as the system should work with the erroneous data entry or when the data entries are successful.
- There are very few sources of information about the next steps for implementation of the XP methodology in software development.
- A description of the artifacts of the XP methodology is short on most Internet information sources only mention them.
- The Jdeveloper tool used to develop SIGFAP system was not the most appropriate to apply the XP methodology, which proposes the development in less time, since not much documentation available for use, there were errors made it difficult correct system development progress.

8. RECOMMENDATIONS

- For the development of systems using XP methodology, use the other Java IDE, NetBeans or Eclipse can be used and if necessary use another database that is much more compatible with these languages.
- Apply the XP methodology for software development, since the device he uses to describe the

operation of the system is simple and easy to follow.

- Undertaken work in pairs in software development, since the knowledge that one of them has to help the other you gain experience, and the less experienced help to other person to finding optimal solutions and the project forward faster.