Procesos de desarrollo empíricos y definidos

Carrión Nicolás, Demarchi Leandro, Enrici Lisandro, Guzmán Federico, Schnell Lara y Villard Facundo

Universidad Tecnológica Nacional - Facultad Regional Córdoba
Ingeniería de Sistemas de Información - Ingeniería de Software

Abstract - El punto central de este reporte es sobre los procesos de desarrollos empíricos y definidos.

Los procesos definidos tradicionales suelen ser más burocráticos y llevan a cabo una gestión más estructurada del proyecto. Esto se da porque los procesos definidos en sus etapas iniciales definen a modo de contrato el producto final junto con sus requerimientos y necesita que el cliente tenga una idea detallada de lo que va a querer como resultado del proceso. Mientras que en procesos empíricos el producto final se va desarrollando a medida que el cliente va presentando sus requerimientos. Al cliente le resulta más sencillo ir entendiendo el producto que necesita conforme se va desarrollando.
Se comienza con una breve introducción, siguiendo con una explicación detallada de ambos tipos de procesos de desarrollo junto a qué beneficios tiene aplicar cada uno de ellos y se explica cómo se aplica el proceso empírico en Scrum. Por otro lado, también es necesario desarrollar la contracara de ambos procesos de desarrollo, mostrando cuáles son sus desventajas.

Palabras Clave:
Procesos - Definido - Empírico - Ágil- Planificación - Etapas - Software

I. Introducción

El proceso de desarrollo definido y el proceso de desarrollo empírico son diferentes enfoques para la gestión de la complejidad de proyectos. La complejidad puede depender de varios factores como las nuevas tecnologías, que son inestables, poseen muchos elementos diferentes a integrar y que se van actualizando para ir atendiendo las nuevas solicitudes que pide el mercado. Otro factor son los requisitos poco definidos, ambiguos, incompletos, poco maduros o cambiantes que puede llegar a tener nuestro cliente. Este factor es fundamental para decidir qué proceso de desarrollo utilizar, ya que una mala elección del mismo puede significar un costo alto en la elaboración del producto.

Uno de los factores más importantes son las personas del equipo de trabajo los cuales poseen diferentes aptitudes técnicas, experiencia, formación, valores, motivaciones, inteligencia, habilidades sociales, etc. y pueden llegar a tener una mejor adaptación a un proceso o al otro.[1]

II. Procesos de Software

Un proceso se define como un conjunto de actividades enlazadas entre sí que, partiendo de uno o más *inputs* (entradas) se genera una transformación, produciendo un *output* (resultado).

Un proceso del software es un conjunto de pasos que conducen a la creación de un producto de software; este no es una disposición exacta de cómo se debe elaborar un software. Por el contrario, es un enfoque adaptable que permite que las personas que hacen el trabajo (el equipo de software) busquen y elijan el conjunto apropiado de acciones y tareas para el trabajo. Se busca siempre entregar el software en forma oportuna y con calidad suficiente para satisfacer a quienes patrocinaron su creación y a aquellos que lo usarán.

La estructura del proceso establece el fundamento para el proceso completo de la ingeniería de software por medio de la identificación de un número pequeño de actividades estructurales que sean aplicables a todos los proyectos de software, sin importar su tamaño o complejidad.

Además, la estructura del proceso incluye un conjunto de actividades sombrilla que son aplicables a través de todo el proceso del software. [8]

Los procesos del software son complejos y como todos los procesos intelectuales y creativos, dependen de las personas que toman decisiones y juicios.

Aunque existen muchos procesos diferentes de software, algunas actividades fundamentales son comunes para todos ellos:

1. Especificación del software: Se debe definir la funcionalidad del software y las restricciones en su operación.

2. Diseño e implementación del software: Se debe producir software que cumpla su especificación.

3. Validación del software: Se debe validar el software para asegurar que hace lo que el cliente desea.

4. Evolución del software: El software debe evolucionar para cubrir las necesidades cambiantes del cliente.

Aunque no existe un proceso del software «ideal», en las organizaciones existen enfoques para poder mejorarlos.[7]

A. Procesos de desarrollo definidos

Los procesos definidos son aquellos que están basados en una rigurosa planificación para el cumplimiento de fases, con sus correspondientes actividades.

Hace referencia a un proceso en el cual se documentan las actividades de las fases, y el personal que la realiza para poder cumplir con ciertas expectativas de burocracia. Estos procesos se pueden

utilizar con uno o varios ciclos de vida de procesos, entre los cuales se puede mencionar, el modelo en cascada, el iterativo y el basado en prototipos.

Están basados en las líneas de producción de las industrias porque sigue siempre los mismos pasos de manera consistente, y ante las mismas entradas siempre obtiene las mismas salidas.

La administración y el control provienen de la predictibilidad de este modelo.

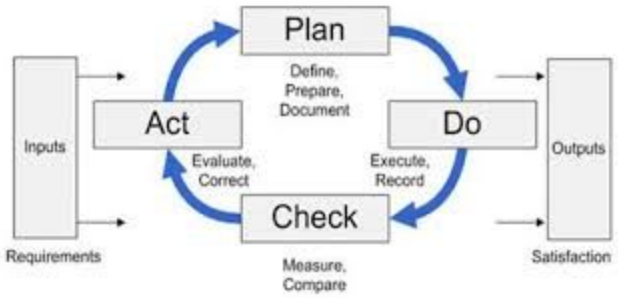

Figura 1: Ciclo Plan-do-check-act [10]

Por lo tanto, podemos decir que el proceso de desarrollo definido se basa en poder predecir y detallar a largo plazo las variables del proyecto (requisitos, planificación y recursos/costo, relacionados directamente con la calidad del producto).

Se intenta solucionar la complejidad de un proyecto esforzándose en una planificación detallada.

Esta planificación emplea el uso más eficiente de los recursos para cumplir plazos, costos y calidad; en definitiva para cumplir la "conformidad con la especificación".

El objetivo de la gestión de proyectos definidos es resolver los planteamientos iniciales cumpliendo con el plazo y con el costo. [3]

Dado que se firman contratos en función de los requisitos iniciales del proyecto, el objetivo principal a lo largo de todo el proyecto será entregar esos requisitos iniciales que fueron firmados, incluso aquellos que aportan un valor ínfimo o los que apenas serán utilizados, no los que el cliente realmente necesitará una vez se le entregue el proyecto. Para no desviarse de la predicción inicial, este proceso necesita de un fuerte control de cambios de requisitos.

En estos procesos la supervisión de proyecto es una actividad continua. El gestor debe tener conocimiento del progreso del proyecto y comparar el progreso con los costos actuales y los planificados. [7]

Entonces para comenzar con el proyecto se reúnen una o varias personas a tiempo completo y se pretende que el cliente (que también se dedica a otras tareas) lo valide, con el riesgo de que esta validación se realice de manera superficial y que al final lo que se desarrolle no sea correcto.

Adicionalmente, se añade el problema de que puede ser difícil para el cliente imaginar y entender de un manera completa desde un inicio cómo será todo el sistema y que la interpretación que realice difiera de la del equipo de desarrollo.

La comunicación entre los miembros del equipo se puede ver disminuida al basarse en documentos donde se detallan los avances de cada miembro del equipo según cada actividad que se le fue asignada (un enfoque que puede resultar muy burocrático y poco

eficiente respecto a producir el valor real que necesita el cliente, o sea, el producto que finalmente se le entregará). Este detalle de los avances de cada miembro del equipo, tampoco facilita que haya una responsabilidad común del proyecto a nivel de equipo.

El cliente no puede aprovechar los resultados del proyecto (o de una etapa de varios meses) hasta que no se ha realizado toda la secuencia de actividades, con lo que después de este tiempo el contexto del proyecto puede haber cambiado. Dado que es la primera vez que el cliente ve los resultados reales del proyecto, puede ser necesario realizar cambios para poder satisfacer sus necesidades, retardando la entrega más de lo que se predijo.

El proyecto cuando está preparado para ser entregado (o al final de una etapa de varios meses), tiene la complicación de que surgen ciertos riesgos y tareas no previstas, retrasando la entrega, obligando al equipo a esforzarse de más y comprometer la calidad del producto, con la consecuente insatisfacción del cliente.

El tiempo de desarrollo para un proyecto grande de software puede ser de varios años. Durante ese tiempo los objetivos organizacionales tienden obviamente a cambiar. Estos cambios pueden significar que el software ya no se necesita o que los requerimientos originales del proyecto son inapropiados. La gestión puede decidir parar el desarrollo del software o cambiar el proyecto para adecuarlo a los cambios de los objetivos de la organización. [7]

Este proceso tiene un defecto fundamental en el que el plan, que impulsa todo, se basa en el supuesto de que los requisitos son fijos y no cambiarán. La experiencia nos ha demostrado que este nunca es el caso; Los requisitos nunca son fijos, siempre cambian. Cuando los requisitos cambian, el plan se ve afectado; y como resultado, la fecha de finalización también debe cambiar. Desafortunadamente, en muchos casos, eso es imposible, y el equipo tiene que entregar en la fecha en que se comprometieron. Esto es cuando ocurre una gran crisis y el proyecto comienza a salirse de control. [9]

1) Ventajas de los procesos de desarrollo definidos

Los procesos definidos se basan en predecir y planificar detalladamente los aspectos del proyecto. Algunas ventajas que podemos obtener gracias a esto son:

- Eliminar o reducir la incertidumbre.
- Mejorar la eficiencia de la operación.
- Obtener una mejor comprensión de los objetivos.
- Dar unas bases para el seguimiento del trabajo.
- Mejor definición de roles de trabajo.
- Cumplimiento de los plazos establecidos

Estos son algunos aspectos a partir de los cuales podemos comprender los beneficios que proporcionan realizar procesos definidos.

2) Desventajas de los procesos de desarrollo definidos

- Poca o nula utilización de evaluaciones que permitan mejorar los métodos de trabajo.
- Incumplimiento de las expectativas del cliente con respecto al producto
- Poca relación y comunicación con el cliente a lo largo del proyecto
- Grandes costos al agregar nuevos requerimientos no detectados, en etapas avanzadas del proyecto
- Metodología de trabajo con mucha burocracia y poca agilidad.
- El cliente no puede aprovechar los resultados de un proyecto hasta no haberse cumplido una secuencia de actividades.

B. Procesos de desarrollo empíricos

El desarrollo empírico, asume que hay un horizonte de predicción de las variables de proyecto dado que hay cambios en el contexto del proyecto debido a la indeterminación y complejidad del mismo. Para gestionar la complejidad y obtener el mayor beneficio posible, el proceso de control del proyecto debe ser empírico, basado en inspección y adaptación regular en función de los resultados que se van obteniendo y del propio contexto del proyecto.

Al cliente le resulta más sencillo ir entendiendo el producto que necesita conforme se va desarrollando, esto le implica un esfuerzo menor al momento de tomar decisiones.

Como ejemplo de adaptación tenemos que, los requisitos a desarrollar en la siguiente iteración dependen del conocimiento que el cliente va adquiriendo acerca del proyecto, de la velocidad de desarrollo, de las demandas del mercado, de los movimientos de los competidores, etc. Los requisitos “emergen” y es necesario adaptarse a ellos a medida que estos aparecen. [1]

1) Control de procesos empíricos

En los proyectos y/o servicios complejos e innovadores, donde hay demasiada variabilidad e incertidumbre, no es práctico trabajar con procesos definidos. Por lo que Scrum aboga el uso de control de procesos empíricos, es decir una forma de control guiada por la experiencia.

Como ya mencionamos anteriormente, el proceso definido hace referencia a un proceso que produce repetidamente una calidad aceptable, pero cuando este no se puede lograr debido a la complejidad de las actividades intermedias se emplea lo que denominamos control de procesos empíricos.

El principal atributo del mismo constituye un ciclo continuo de inspección del proceso para el correcto funcionamiento, resultado y adaptación de ser necesario. Hay tres elementos

claves para controlar el proceso empírico:

- **Visibilidad**. Aquellos aspectos del proceso que afectan el resultado deben ser visibles para quienes controlan el proceso.

- **Inspección**. Los diversos aspectos del proceso deben inspeccionarse con la frecuencia suficiente para que se puedan detectar incongruencias en el proceso. La frecuencia de la inspección debe tener en cuenta el hecho de que es probable que el proceso se modifique como resultado de la inspección. El inspector debe poseer las habilidades para evaluar lo que está inspeccionando.[4]

- **Adaptación**. Si el inspector determina a partir de la inspección que uno o más aspectos del proceso están fuera de los límites aceptables y que el producto será incorrecto, el inspector debe ajustar el proceso o el material que se está procesando. El ajuste debe hacerse lo más rápido posible para minimizar una desviación adicional.[5]

Para lograr un control de proceso empírico, Scrum establece un marco iterativo e incremental. Se divide en:

- **Trabajo**: a una lista de resultados pequeños y concretos, clasifica la lista por prioridad y estima el esfuerzo relativo de cada uno.
- **Tiempo**: en iteraciones cortas de longitud fija con código potencialmente despachable demostrado después de cada iteración.
- **Organización**: en pequeños equipos auto-organizativos multifuncionales. [4]

2) Aplicaciones de Scrum en los procesos empíricos

Es muy común la aplicación de Scrum en los procesos de desarrollo empírico, de manera frecuente mediante ciertas prácticas:

- Los miembros del equipo de desarrollo se comunican diariamente para mostrar sus avances, y poder tener en cuenta los adaptaciones necesarias a realizar, para poder conseguir el objetivo de la iteración actual y del proyecto
- El cliente proporciona una respuesta constante sobre el proyecto a partir de las entregas que se van realizando al final de cada iteración o sprint. A partir de cada iteración y del feedback del cliente se pueden realizar adaptaciones en la lista de requisitos, pudiendo o no modificar la prioridad de los ya existentes.
- Las iteraciones que va realizando el equipo permite ir mejorando la metodología de trabajo, ya que se van conociendo el ritmo y las capacidades de cada integrante de cada uno. Esto permite mejorar la

planificación de las siguientes iteraciones y el cumplimiento de las mismas. [2]

3) Ventajas de los procesos empíricos

- Es útil en proyectos complejos, donde no hay requerimiento bien definidos desde el comienzo
- Permite gestionar de mejor manera la variabilidad de requerimientos que puede llegar a tener el cliente, adaptándose a ellos.
- A través de la inspección se va realizando una mejora continua de la metodología de trabajo
- Permite una mejor comunicación con el cliente.
- Realiza retrospectivas a lo largo de todo el proyecto para aumentar la calidad y la productividad del proyecto
- Mejora la participación activa y colaborativa de todo el equipo de trabajo
- Permite una mayor flexibilidad en la planificación con el cliente.
- Se entrega software funcionando en un periodo corto tiempo, lo que le da la posibilidad al cliente de contar con éste para realizar capacitaciones de próximos usuarios, o simplemente utilizarlo.

4) Desventajas de los procesos empíricos

- Poseen una documentación que muchas veces puede ser escasa y sin tantos detalles, dificil de comprender para alguien externo al proyecto.
- En un comienzo del proyecto,puede llegar a ser más costoso que un proceso definido
- Suele ser menos útil en proyectos donde los requerimientos están claros y hay poca complejidad.
- Las iteraciones con sus adaptaciones e incrementos correspondientes requieren de la colaboración de todo el equipo de trabajo que en muchos casos no es tan fácil de conseguir.

III. Conclusión

Podemos concluir, que estos dos procesos de desarrollo, son muy útiles si se los aplica en el contexto correcto, donde se puedan explotar más sus ventajas y reducir sus complicaciones. Si se quiere aplicar un proceso a un proyecto donde hay poca incertidumbre, y el cliente tiene bien claro cómo quiere que sea su producto, el proceso definido puede ser la mejor opción. En cambio, si el cliente no tiene una certeza total de cómo quiere que sea el producto, ni la tecnología del mismo, teniendo el proyecto un cierto nivel de complejidad e incertidumbre, las características se adaptan mejor a un proceso de desarrollo empírico, donde al cliente le pueden ir surgiendo nuevos requerimientos a partir de las iteraciones que se van realizando, pudiendo ahi terminar de definir cómo quiere que sea el producto final.

IV. Referencias

[1] https://proyectosagiles.org/control-predictivo-control-empirico/
[2] https://proyectosagiles.org/base-conocimiento-agil/
[3] https://www.scrummanager.net/bok/index.php?title=Gesti%C3%B3n_predictiva
[4] https://www.infoq.com/articles/the-curse-of-the-change-control-mechanism
[5] https://www.linkedin.com/pulse/3-pilares-del-empirismo-en-scrum-adonis-ricardo-rosales-garc%C3%ADa/
[6] Estrutuctura del informe tecnico: http://www.unisecmexico.com/archivosPDF/Formato_IEEE.pdf
[7] Ingeniería del Software 7ma. Ed. - Ian Sommerville
[8] Ingeniería de Software. Enfoque Práctico, Pressman
[9] https://medium.com/@warren2lynch/why-scrum-defined-process-vs-empirical-process-927a3d29aaa
[10] https://www.bizmanualz.com/improve-business-processes/what-is-a-well-defined-process.html