## 

Universidad Tecnológica Nacional

Facultad Regional Córdoba

Ingeniería en Sistemas de Información

Cátedra: Ingeniería de Software

TRABAJO PRÁCTICO N° 14:

Comparación de Enfoques para Gestión del Desarrollo de Productos de Software

Curso: 4K1

|  |  |
| --- | --- |
| Grupo N° 1:   * Arroyo, Camila - 74460 * Azabal, Dahyana - 75827 * Chiavassa, Agustín - 76255 * Gonzalez, Lautaro - 75174 * Mok, Ivan - 75378 * Vivas Castillo, Matías - 75828 | Docentes:   * Meles, Judith * Robles, Joaquin Leonel * Crespo, María Mickaela |

# Introducción

El objetivo del siguiente informe es presentar una comparación entre tres diferentes enfoques de gestión de desarrollo de productos de software, muy utilizados a nivel global. Buscamos identificar las diferencias y similitudes entre ellos.

Los enfoques que analizaremos son: Enfoque tradicional basado en procesos definidos, Scrum y Kanban.

El análisis de los enfoques se llevará a cabo a partir de un cuadro comparativo, donde seleccionaremos puntos de convergencia y divergencia entre los enfoques para analizar los alcances de cada uno, y las fortalezas y debilidades que presentan. Este análisis nos permite entender en qué situaciones se prefiere un enfoque frente a los demás y nos ayuda a tomar una decisión acertada al momento de elegir cómo gestionaremos el desarrollo de un producto de software.

Antes de realizar la comparación, introduciremos brevemente cada uno de los enfoques de gestión a analizar.

El enfoque tradicional es una metodología donde los proyectos siguen una secuencia fija de pasos. Se pone especial énfasis en los procesos, la documentación y la planificación en detalle por adelantado. Los requerimientos son fijos y están establecidos de antemano. Además, para cada paso hay herramientas y técnicas definidas por el estándar que marca la metodología.

Scrum y Kanban son ambos empíricos respecto a que se espera que el proceso se adapte al entorno de desarrollo.

Scrum, por su lado, consiste en equipos pequeños y auto-organizados. Se divide el tiempo en iteraciones cortas de longitud fija, con versiones de producto potencialmente entregables después de cada iteración. Acto seguido, se optimiza el plan de entregas y se actualizan las prioridades en colaboración con el cliente, en base a la inspección del entregable. Es decir, tenemos un equipo menor pasando un tiempo más corto construyendo algo menor, pero integrando con regularidad para ver el conjunto.

Por otro lado, Kanban divide el trabajo en bloques. Consiste en que se escribe cada elemento en una tarjeta y se pone en un muro. Utiliza columnas con nombre para ilustrar dónde está cada elemento en el flujo de trabajo. Además, si bien se asignan límites concretos a cuántos elementos pueden estar en progreso en cada flujo de trabajo, no se prescriben iteraciones de tiempo fijo.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Tradicional** | **Scrum** | **Kanban** |
| **Proceso** | Definido | Empírico | Empírico |
| **Ciclo de vida** | Sirve para cualquier ciclo de vida. | Iterativo e incremental. | Iterativo e incremental. |
| **Centrado en** | Proceso | Cliente | Cliente |
| **Prioridad** | Cumplir el plan | Entregar valor | Entregar valor |
| **Documentación** | Exhaustiva, se documenta todo. | Mínimo, según se requiera. | No exige documentación en forma explícita. |
| **Entrega de valor** | Al final del proyecto. | A lo largo de todo el proyecto. Busca entregar incrementos al final de cada sprint con la aprobación del product owner. | Utiliza un sistema de flujo de entrega que limita la cantidad de trabajo en proceso. El flujo de trabajo en un sistema kanban debería maximizar la entrega de valor, minimizar los tiempos de entrega y ser tan predecible como sea posible. |
| **Aceptación de cambios** | Como la entrega de valor se hace al final del proyecto, el feedback del cliente se recibe al final y no durante el desarrollo. Es poco flexible a los cambios. | Los cambios se reciben en todo momento y se colocan en el Product Backlog. | Los cambios pueden ocurrir en cualquier momento y se colocan en la columna de pendientes en el tablero Kanban, en tanto la misma tenga capacidad. Si no posee capacidad se deberá quitar un elemento. |
| **Impacto de cambios** | Bajo si es en las etapas iniciales del proyecto. Conforme avanza el mismo se podría decir que aumenta exponencialmente, por el retrabajo que implica. | Medio a bajo, al trabajar con time-boxing y limitar el alcance en funcionalidad de las iteraciones un cambio provoca a lo sumo un impacto en el trabajo en curso en una iteración. | Bajo. Kanban limita la cantidad de trabajo en progreso en cada etapa, sumado a que el dinamismo que impone el flujo de items de un estado al otro provoca que el impacto de los cambios esté relativamente limitado a los trabajos que estaban en curso. |
|  | **Tradicional** | **Scrum** | **Kanban** |
| **Características / Principios** | En el Proceso Unificado de Desarrollo:   * Iterativo e incremental. * Dirigido por casos de Uso. * Centrado en la arquitectura. | * Satisfacer al cliente a través de releases tempranos y frecuentes. * Hacer proyectos con individuos motivados. * El medio de comunicación por excelencia es el cara a cara. * Equipos autoorganizados. * Releases frecuentes. * La mejor métrica de progreso es la cantidad de software funcionando. * Simplicidad Maximización del trabajo no hecho. * Técnicos y no técnicos trabajando juntos TODO el proyecto. * Recibir cambios aún en etapas finales. * El ritmo de desarrollo es sostenible en el tiempo. * A intervalos regulares, el equipo evalúa su desempeño y ajusta la manera de trabajar. * Atención continua a la excelencia técnica y al buen diseño | * Sostenibilidad. Relativo a encontrar un ritmo sostenible y foco en la mejora. * Orientación al servicio. Enfocado a conseguir rendimiento y satisfacción del cliente. * Supervivencia. Relativo al mantenimiento de la competitividad y la adaptabilidad. |
| **Participación del cliente o usuario del software** | Mínima, predominante en las primeras etapas donde se lleva a cabo la elicitación de requerimientos. | Alta pero no siempre directa. Se busca involucrar constantemente al cliente o usuarios a través del Product Owner. Sí pueden tomar parte directa en las Sprint Review. | Si bien uno de los principios de kanban es la orientación al servicio hacia el cliente, el mismo no participa de manera directa en el proyecto. |
|  | **Tradicional** | **Scrum** | **Kanban** |
| **Roles** | Define una gran cantidad de roles específicos para cada workflow. Se pueden plantear algunos subgrupos que luego se especializan en cada workflow:   * Líder de Proyecto. * Analista. * Arquitecto. * Desarrollador / Diseñador. | Plantea 3 roles definidos:   * Product Owner * Scrum Team * Scrum Master | No hay roles definidos formalmente, queda en manos del equipo definir los roles que consideran necesarios para cumplir sus objetivos. |
| **Artefactos** | Los artefactos orientados a la gestión de Producto son:   * Documento de especificación de requerimientos de software (ERS). | Define 3 artefactos:   * Product Backlog * Sprint Backlog * Incremento del producto | * Tablero Kanban |
| **Compatibilidad con los otros enfoques** | Mínima a nula, la rigidez impuesta por los enfoques basados en procesos definidos restringe integrar los otros enfoques dentro del proceso. | Siempre que se respete el concepto de time-boxing es posible integrar los conceptos que enuncia Kanban dentro de las iteraciones para visualizar el flujo  de trabajo.  Incompatible con enfoques basados en procesos definidos por el grado de libertad que provee a la hora de definir algunas cuestiones como los artefactos. | Es compatible con Scrum, respetando lo enunciado en la casilla anterior.  Dentro de un proceso definido podría llegar a adaptarse teniendo cuidado de respetar la metodología de trabajo impuesta por el proceso seleccionado. |
| **Work In Progress** | No se establece un límite de trabajo en curso de manera simultánea entre los flujos de trabajos. | No hay ninguna regla que impida que el equipo ponga en la columna en curso todos los elementos al mismo tiempo. | Define un límite de trabajo por columnas. |
|  | **Tradicional** | **Scrum** | **Kanban** |
| **Visualización del trabajo** | No plantea un método de visualización del trabajo. | Plantea la utilización de un tablero Scrum donde:   * Cuando finaliza el Sprint, se limpia el tablero. Cuando se inicia un nuevo Sprint, se crea un nuevo tablero. * El tablero Scrum pertenece solamente a un equipo. | Plantea la utilización de un tablero Kanban donde:   * El tablero es persistente durante todo el proyecto. * El tablero está relacionado con el flujo de trabajo y no necesariamente con un equipo. |
| **Estimación** | No plantea una forma de estimación específica. En cada workflow se realizan estimaciones para cada artefacto. | Los equipos deben estimar el tamaño relativo de cada elemento al que se comprometen. | Las estimaciones son opcionales. |
| **Métricas de proceso** | Las métricas por defecto para la planificación y mejora del proceso son:   * Esfuerzo * Tiempo * Costos * Riesgos | La métrica por defecto para la planificación y mejora del proceso es la velocidad. | La métrica por defecto para la planificación y mejora del proceso es el Lead Time. |

# Conclusión

Concluímos en que el tipo de gestión que se elija depende del producto que el cliente requiera. No se puede establecer que alguna de las metodologías es mejor que otra en sí misma, sólo puede ser más adecuada para el desarrollo de un producto en específico. Pretendemos finalizar este trabajo obteniendo una vista objetiva de cada una de ellas, para así poder decidir la metodología adecuada en la gestión de cualquier producto que se nos presentara.

La gestión del desarrollo de productos que provee Scrum está formado por un conjunto de prácticas que responden a los principios ágiles. Es iterativo e incremental, en donde cada iteración (Sprint) entrega una parte funcional del producto final, es decir, el incremento. Por lo que Scrum resulta adecuado en proyectos con requerimientos inciertos y/o inestables.

Lo que se diferencia de la gestión tradicional, la cual especifica que lo primero a realizarse es la especificación de requerimientos y dedica mucho tiempo a establecer requerimientos sólidos y bien definidos. Además, se planifica todo con gran detalle y se destaca un enfoque sobre los procesos a seguir, lo que favorece la anticipación a posibles hechos que pueden ocurrir.

Por otro lado, Kanban presenta una ideología bastante diferente a la de Scrum y a la Tradicional, ya que se caracteriza por el principio de “Empieza por donde estés”. Es decir, no establece ningún plan de trabajo o una manera de iniciar el trabajo. La restricción que plantea Kanban es la de limitar el trabajo, siguiendo el principio de Sostenibilidad, que se enfoca en la construcción de servicios evitando la sobrecarga de tareas. Esto implica utilizar un tablero kanban para la visualización de estas tareas limitadas, que es el mecanismo fundamental que utiliza.

Para finalizar, obtenemos que Scrum es útil cuando se busca desarrollar un producto con requerimientos inciertos y cambiantes. La metodología tradicional es óptima cuando se tienen requerimientos sólidos. Y por último, Kanban es la metodología que se enfoca en la resistencia al cambio y la adaptabilidad de equipos ya formados.

# Webgrafía

[Fair, J. (2012). Agile versus Waterfall: approach is right for my ERP project? Paper presented at PMI® Global Congress 2012—EMEA,](https://www.pmi.org/learning/library/agile-versus-waterfall-approach-erp-project-6300)

[Marsailles, France. Newtown Square, PA: Project Management Institute.](https://www.pmi.org/learning/library/agile-versus-waterfall-approach-erp-project-6300)

[Ken Schwaber y Jeff Sutherland(2013) La Guía Definitiva de Scrum - Scrum Basics.](https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-es.pdf)

Bibliografía

Henrik Kniberg y Mattias Skarin (2010) Kanban y Scrum – Obteniendo lo mejor de ambos

David J. Anderson y Andy Carmichael - Essential-Kanban-Condensed

Jacobson, Booch y Rumbaugh - “EL PROCESO UNIFICADO DE DESARROLLO” (Editorial Addison-Wesley - Año 2000 1ª edición).