DEFINICIÓN Y PUESTA EN PRÁCTICA DE PROCESO DE DESARROLLO DE SOFTWARE PARA TIENDA DE WINDOWS 8

Proyecto de Grado

21 de Abril de 2014
Montevideo - Uruguay

Autores:

Germán Catarino

Alexis Gonzatto

Rodrigo Machín

Agradecimientos

El proyecto y los resultados del mismo, están dedicados a todas aquellas personas que, directa o indirectamente formaron parte. Gabriel López, Álvaro Azofra y Martin Cabrera por la colaboración brindada durante el estado del arte. A Agustín Vignoli y Joaquín Menéndez por ayudar en el diseño gráfico del producto construido. A Marcelo Guerra y Jorge Triñanes por oficiar de tutores y en parte clientes.

Resumen

El presente proyecto, surge como propuesta del área: Grupo Ingeniería de Software, del instituto de Computación de la Facultad de Ingeniería, en conjunto con Marcelo Guerra: Líder de Verificación en Microsoft. Ambas partes, interesadas en procesos de desarrollo y mantenimiento de software, deciden definir un modelo de proceso específico para construir aplicaciones comercializables dentro de: Windows 8 Marketplace.

El objetivo principal de este proyecto, es definir un modelo de proceso de producción de software específico para productos comercializables en la tienda de Windows 8. Para ello el transcurso del proyecto se basa en el enfoque de Investigación-Acción (Action Research), lo que implica aproximaciones sucesivas a la definición del modelo de proceso en iteraciones de tipo análisis-definición-puesta en práctica-reflexión.

La primera etapa del trabajo, consistió en el estudio del mercado objetivo en base a mercados paralelos similares, el estudio de técnicas que ayuden a crear un negocio exitoso y en base a ambos estudios la definición de la aplicación o tipo de aplicación a construir. En la segunda etapa se investigó dentro de las metodologías agiles cuál modelo de proceso se adopta mejor al producto elegido en función a lo relevado en el mercado objetivo. Para ello se realizaron relevamientos de procesos seguidos por organizaciones exitosas en mercados análogos. Además se realizó un análisis sobre rasgos del producto que resulten relevantes para lograr un mayor impacto. En la tercera etapa se puso en práctica el modelo de proceso seleccionado, implementando la solución pactada. A partir de los resultados obtenidos, se refinó dicho proceso en sucesivas sub etapas.

Índice

1. Objetivo del Trabajo

1.1 Introducción

A continuación se presenta el alcance y los objetivos del proyecto. En la sección 1.2 se presenta la definición del proyecto y su contexto. En la sección 1.3 se describe la motivación del problema. La sección 1.4 presenta los objetivos del proyecto. La sección 1.5 muestra los resultados esperados. Por último en la sección 1.6 se describe la organización de este informe.

1.2 Contexto

Windows Marketplace es la plataforma de distribución digital de software desarrollada por Microsoft como parte de Windows 8 para aplicaciones de escritorio. Esta permite a los desarrolladores publicar sus aplicaciones y a los usuarios finales: buscar, obtener información y descargar aplicaciones publicadas por terceros.

Fue lanzada en noviembre de 2012 y cuenta hoy en día con 142.000 [27] aplicaciones categorizadas en: juegos, música y video, noticias y el tiempo, entretenimiento, libros y referencia y educación.

1.3 Motivación

En los últimos años, el comercio de aplicaciones a través de tiendas en línea se ha consolidado fuertemente. Cada plataforma implementa su propia tienda y da la posibilidad, con herramientas gratuitas y licencias de bajos costos anuales, la posibilidad de que desarrolladores expongan sus productos. Aplicaciones con crecimientos exponenciales como: Candy Crash o Angry Birds [28], que llegan a facturar millones de dólares al día, logran su éxito gracias a la masividad y accesibilidad de las tiendas en línea. Esto lleva a la necesidad de crear un proceso de desarrollo de software, que de manera óptima, se adapte a la lógica de estos mercados.

La principal motivación de este proyecto es generar un producto exitoso, minimizando los costos y maximizando su impacto en el mercado. Llegar a la consolidación de un proceso de desarrollo de software a partir del análisis al mercado al que se apunta.

1.4 Objetivos

El primer objetivo es crear un modelo de proceso para desarrollar productos para la tienda de aplicaciones de Windows 8. La definición del proceso tomará en cuenta los atributos de calidad del producto y de la tienda objetivo (que

resulten relevantes), para lograr un mayor impacto en ese mercado. Para ello se estudiarán metodologías agiles considerando los procedimientos, documentación, metodologías, etc., más apropiadas. La definición del modelo de proceso estará basada en el enfoque de Investigación Acción (Action Research), lo que implica aproximaciones sucesivas a la definición del modelo de proceso en iteraciones de tipo análisis-definición-puesta en práctica-reflexión.

El segundo objetivo del proyecto es la selección de un producto o conjunto de productos a desarrollar en base a la lógica de negocio y al público objetivo que apunta la Tienda de Windows. Esta primera etapa implica un estudio del mercado objetivo considerando las aplicaciones más descargadas por categoría, las categorías existentes y los rankings gracias a votaciones de los usuarios. También hay que considerar el público objetivo al que apunta el mercado, su rango etario, distribución geográfica, poder adquisitivo, comportamiento o apego a la tecnología, etc.

El tercer y último objetivo es poner en práctica del proceso creado, construyendo en sucesivas iteraciones el producto seleccionado. Lo que se espera es ajustar progresivamente el proceso seleccionado, de manera de lograr un mayor impacto en ese mercado objetivo.

1.5 Resultados Esperados

Realizar un estudio del mercado al que se apunta y relevamiento y análisis de procesos seguidos por organizaciones exitosas para desarrollar aplicaciones para mercados análogos.

Investigar diversas metodologías de desarrollo de software basadas en metodologías agiles. Combinar las mismas con buenas prácticas que permitan crear un producto exitoso.

Llegar a la definición de un proceso de desarrollo de software y luego ponerlo en práctica para su posterior análisis en base a los resultados. Se espera publicar la aplicación resultado y llegar a las 100 descargas de usuarios independientemente del costo de la aplicación, tipo de usuario o tiempo.

1.6 Organización del documento

El presente documento cuenta con cinco capítulos. En el capítulo 1 se presenta una introducción inicial en el cual se detallan las características del problema a resolver, su contexto, motivación, objetivos del proyecto y los resultados esperados.

El capítulo 2 contiene las definiciones y características generales. Se realiza una breve introducción a los conceptos necesarios para entender el trabajo realizado. Se hace un relevamiento de los procesos y metodologías a combinar.

En el capítulo 3 presenta el modelo de proceso a seguir. Aquí, se detallan las decisiones tomadas en el diseño de la solución, se describe la arquitectura del sistema a construir, las tecnologías, metodologías de desarrollo y atributos de calidad relevantes para tener un mayor impacto en el mercado tanto del producto como del proceso a seguir.

El capítulo 4 presenta la ejecución del proceso y las adaptaciones realizadas a lo definido anteriormente.

En el último capítulo se presentan las conclusiones y trabajo futuro.

1.7 Público objetivo

Este documento tiene como público objetivo personas relacionadas al área de Tecnología de la Información con conocimientos en ingeniería del software y programación. También es aconsejable conocimientos sobre mercados virtuales y metodologías agiles para una mayor comprensión del documento.

2. Fase 1: Análisis y Estudio de los Procesos

2.1 Introducción

En este capítulo se presenta el marco teórico utilizado para la definición del proceso. Se pone énfasis en las metodologías agiles, entre ellas se destacan: Scrum, XP, y Lean Startup. También destacaremos las normas de calidad: ISO 9126, 9000:2000. Por último comentamos como fueron las entrevistas con expertos en desarrollo ágil que nos ayudaron en la toma de decisiones a la hora de construir el nuevo proceso.

2.2 Relevamiento de Procesos Existentes

2.2.1 Metodologías de desarrollo

Una metodología se puede definir como un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores de una tarea. Una metodología de desarrollo de software es un Framework de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información.

Estos Frameworks son a menudo vinculados a algún tipo de organización, que además desarrolla, apoya el uso y promueve la metodología. La metodología es a menudo documentada en algún tipo de documentación formal.

2.2.2 Metodologías agiles

Para asegurar el éxito durante el desarrollo de software no es suficiente contar con notaciones de modelado y herramientas, hace falta un elemento importante: la metodología de desarrollo, la cual nos provee de una dirección a seguir para la correcta aplicación de los demás elementos.

Generalmente el proceso de desarrollo llevaba asociado un marcado énfasis en el control del proceso mediante una rigurosa definición de roles, actividades y artefactos, incluyendo modelado y documentación detallada. Este esquema tradicional para abordar el desarrollo del software, ha demostrado ser efectiva y necesaria en proyectos de gran tamaño (respecto a tiempo y recursos). Sin embargo, este enfoque no resulta ser el más adecuado 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.

En este contexto, las metodologías ágiles emergen como una posible respuesta para llenar ese vacío metodológico. Por estar especialmente orientadas a proyectos cambiantes, constituyen una solución a medida para ese entorno,

aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del producto.

Las metodologías ágiles surgen a principios de la década del 90'. Son basadas en el desarrollo iterativo e incremental, donde los requerimientos y soluciones evolucionan mediante la colaboración de grupos autos organizados y multidisciplinarios. Se caracterizan por minimiza riesgos desarrollando software en lapsos cortos. El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación. Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, pero la meta es tener una versión sin errores al final de cada iteración. Finalizada una iteración el equipo vuelve a evaluar las prioridades del proyecto.

Estas metodologías enfatizan las comunicaciones cara a cara en vez de la documentación. Consideran que el software funcionando es la primera medida de progreso.

Las metodologías ágiles de desarrollo están especialmente indicadas en proyectos con requisitos poco definidos o cambiantes. Estas metodologías se aplican bien en equipos pequeños que resuelven problemas concretos, lo que no está reñido con su aplicación en el desarrollo de grandes sistemas. Dividir el trabajo en módulos abordables minimiza los fallos y el coste. Es por ello que presentan diversas ventajas, entre las que podemos destacar:

- ✓ Capacidad de respuesta a cambios de requisitos a lo largo del desarrollo.
- ✓ Entrega continua y en plazos breves de software funcional.
- ✓ Trabajo conjunto entre el cliente y el equipo de desarrollo.
- ✓ Importancia de la simplicidad, eliminado el trabajo innecesario.
- ✓ Atención continua a la excelencia técnica y al buen diseño.
- ✓ Mejora continua de los procesos y el equipo de desarrollo.

2.2.3 Manifiesto ágil

El manifiesto ágil (año 2001) surge a partir de una iniciativa conjunta entre los responsables de los procesos ágiles, con el fin de lograr unificar los principios compartidos por las diversas metodologías de manera de crear un Framework de trabajo que contribuya a la mejora de desarrollo.

> "Estamos descubriendo mejores maneras de desarrollar software mediante su construcción y ayudando a que otras personas lo construyan. A través de este trabajo hemos llegado a valorar: Individuos e interacciones sobre procesos y personas, software funcionando sobre documentación comprensiva, colaboración del cliente sobre negociación de contrato, responder al cambio sobre seguir un

plan. Esto es, mientras que existe valor en los ítems de la derecha, valoramos más los ítems de la izquierda." [1]

Analizando el manifiesto ágil vemos que se realiza un especial énfasis sobre las personas, está enmarcado en la primera oración respecto a los valores de las metodologías. El énfasis en el código por sobre los documentos es una propuesta eficaz para lograr el rápido Feedback requerido por la agilidad. La participación del cliente es fundamental para el éxito del proyecto ya que ellos definen la funcionalidad a construir. La importancia de responder al cambio indica que el desarrollo de software no es más que un proceso de aprendizaje en que cada cambio nos permite conocer más en detalle el dominio de la aplicación.

2.2.4 Scrum

Scrum es una metodología ágil para gestionar proyectos de software, que emerge en base a estudios realizados sobre nuevas prácticas de producción en ambientes cambiantes e inestables. Aunque surgió como práctica en el desarrollo de productos tecnológicos, resulta válida en entornos que trabajan con requisitos inestables, con necesidades de rapidez y flexibilidad; situaciones habituales en el desarrollo de sistemas de software.

2.2.4.1 Principales Características

Es una metodología de desarrollo muy simple, que requiere trabajo duro, ya que la gestión no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto. Se caracteriza por su modo de desarrollo adaptable, donde es orientado a las personas y no a los procesos y la construcción del producto es iterativa e incremental.

A diferencia de los modelos de proceso tradicionales con su gestión predictiva, Scrum comienza el desarrollo con apenas una visión general del producto. A partir de esta, se da detalle solo a las funcionalidades que, por ser las de mayor prioridad para el negocio, son desarrolladas en primer lugar, y pueden llevarse a cabo en un periodo de tiempo breve (entre 15 y 60 días). No intentar predecir en las fases iniciales cómo será el resultado final, ya que la estructura del producto no es realista. Las circunstancias del proyecto obligarán a remodelarlo muchas veces la solución final. Scrum toma la inestabilidad como premisa y por ello es importante permitir la evolución continua sin degradar la calidad de la arquitectura, que se irá generando durante el desarrollo.

Con Scrum, el diseño y la estructura del resultado se construyen de forma evolutiva. No se considera a la descripción detallada del producto, del servicio, de la estrategia o de la arquitectura del software (según el caso) deban realizarse en la primera etapa del proyecto. El desarrollo ágil no es un desarrollo por fases. En la aplicación de Scrum para software, para evitar los problemas de degradación del sistema o de la arquitectura por la evolución continua del producto se deben incluir prácticas de refactorización en las tareas de diseño y codificación.

La construcción del producto es en base a iteraciones sucesivas, no en base a abstracciones y diseños pesados. En cada uno de estos ciclos de desarrollo (denominados: Sprint) se produce un incremento terminado y operativo del producto. Esto implica que al final de cada iteración se posee una parte de producto operativa para inspeccionar y evaluar.

Las iteraciones se gestionan diariamente a través de reuniones breves de seguimiento en las que todo el equipo revisa el trabajo realizado desde la reunión anterior y el previsto hasta la reunión siguiente. A su vez, al final de cada Sprint, se realiza una revisión con todas las personas involucradas en el proyecto ya que este es el periodo máximo que se puede tardar en reconducir una desviación del proyecto o de las circunstancias del producto.

2.2.4.2 Ejecución del proceso

En Scrum el resultado final se construye de forma iterativa e incremental. Al comenzar cada iteración se determina qué partes se van a construir, tomando como criterios la prioridad para el negocio, y la cantidad de trabajo que se podrá abordar durante la iteración. Este trabajo es extraído del Product Backlog hacia el Sprint Backlog.

2.2.4.2.1 ELEMENTOS

El *Product Backlog* es una pila donde están almacenados los requisitos del sistema. Es el inventario de características que el propietario del producto desea obtener, ordenado por prioridad. Es un documento "vivo", en constante evolución. Es accesible por todas las personas que intervienen en el desarrollo. Todos pueden contribuir y aportar sugerencias. El responsable del Product Backlog es una única persona y se le denomina: propietario del producto.

El *Sprint Backlog* es la lista de los trabajos que realizará el equipo durante un Sprint para generar el incremento previsto. Entonces, el equipo en cada interacción asume el compromiso de la ejecución.

Tanto el Product Backlog como el Sprint Backlog son el registro de los requisitos del sistema. El Product Backlog se sitúa en el área de requisitos o necesidades de negocio desde el punto de vista del cliente. Área que en la ingeniería del software tradicional cubren los requisitos del sistema o ConOps (Concept of Operations). El Sprint Backlog se sitúa en el área de especificación de los requisitos de software necesarios para dar respuesta a las funcionalidades esperadas por el cliente.

A diferencia de un documento de requisitos del sistema, el Product Backlog nunca se da por completo; está en continuo crecimiento y evolución. Habitualmente se comienza a elaborar con el resultado de una reunión de "fertilización cruzada" o brainstorming; o un proceso de "Exploración" donde colabora todo el equipo partiendo de la visión del propietario del producto. El

formato de la visión no es relevante. Según los casos, puede ser una presentación informal del responsable del producto, un informe de requisitos del departamento de marketing, etc.

El Product Backlog evolucionará de forma continua mientras el producto esté en el mercado, con el fin de otorgar valor, utilidad y competitividad de forma continua.

2.2.4.2.2 GESTIÓN

En cuanto a la gestión, cada Sprint comienza con la Planificación del Sprint (*Planning Meetings*). Esta es una jornada de trabajo previa al inicio de cada Sprint en la que se determina cuál es el trabajo y los objetivos que se deben cubrir con esa iteración. Esta reunión genera el Sprint Backlog o lista de tareas que se van a realizar, y en ella también se determina el objetivo del Sprint: lema que define la finalidad de negocio que se va a lograr (Goal). A partir de allí diariamente se realizan las reuniones de seguimiento del Sprint (*Daily Meetings*). Estas son breves donde se repasa el avance de cada tarea, y al trabajo previsto para la jornada. Sólo interviene el equipo, y cada miembro responde a tres preguntas: trabajo realizado desde la reunión anterior, trabajo que se va a realizar hasta la próxima reunión de seguimiento e impedimentos que se deben solventar para que pueda realizar el trabajo. Por último existe la revisión del Sprint. En estas revisiones se realiza un análisis del incremento generado. Esta reunión no debe tomarse como un acontecimiento especial, sino como la presentación normal de los resultados.

En detalle, en la Planificación del Sprint se toman como base las prioridades y necesidades de negocio del cliente, y en función de estas se determinan cuáles y cómo van a ser las funcionalidades que se van a incorporar al producto en el próximo Sprint. En realidad esta reunión consiste en dos etapas: en la primera, se decide qué elementos del Product Backlog se van a desarrollar. En la segunda se desglosan éstos para determinar las tareas necesarias, estimar el esfuerzo que necesita cada una y asignarlas a las personas del equipo. La planificación del Sprint no debe durar más de un día.

Para llevar a cabo la Planificación del Sprint, la organización debe tener determinados los recursos posibles para llevar a cabo el Sprint, y además debe estar preparado el Product Backlog del producto por parte del propietario. Esto significa que, con su criterio de prioridad para el negocio, el propietario debe numerar y priorizar las suficiente cantidad de historias para desarrollar en el Sprint. Además, siempre que sea posible el propietario del producto debe haber trabajado ya previamente con el equipo. De esta forma su estimación previa la cantidad de pila de product a desarrollar en el Sprint será bastante ajustada. Por otro lado, el equipo debe tener un conocimiento de las tecnologías empleadas, y del negocio del producto suficiente para realizar estimaciones basadas en juicios de expertos, y para comprender los conceptos del negocio que expone el propietario del producto.

Como resultado de la Planificación del Sprint se debe obtener: el Product Backlog del Sprint, la duración del Sprint y fecha de la reunión de revisión y los objetivos del Sprint.

Por lo general, es una reunión conducida por el responsable del funcionamiento de Scrum, a la que deben asistir el propietario del producto y el equipo completo. También pueden asistir otros implicados en el proyecto. La reunión comienza con la presentación por parte del propietario del producto del Product Backlog. Allí se exponen los resultados por orden de prioridad necesaria; especialmente los que prevén que se podrán desarrollar en el siguiente Sprint. Si el Product Backlog ha tenido cambios significativos desde la anterior reunión; explica también las causas que los han ocasionado. El objetivo es que todo el equipo conozca las razones y los detalles con el nivel necesario para poder estimar el trabajo necesario.

En cuanto a las reuniones de seguimiento del Sprint, son reuniones diarias breves, de no más de 15 minutos en la que todos los miembros del equipo dicen las tareas en las que están trabajando, si se han encontrado o prevén encontrarse con algún impedimento y actualizan sobre el Sprint Backlog las tareas ya terminadas o los tiempos de trabajo que les quedan.

En ellas asiste todo el equipo, con el Sprint Backlog actualizado en el soporte que emplee el equipo (dibujado en pizarra o sobre hoja de cálculo) y un miembro del equipo (Team Leader) conduce y garantiza el protocolo, formato y tiempos de la reunión. Se recomienda realizarla de pie y emplear un formato de Product Backlog o lista de tareas en una pizarra o en la pared, para que todo el equipo pueda verlo, anotar o mover las tareas, junto con el gráfico de avance del Sprint.

Por último, la reunión de revisión del Sprint, es realizada al final de cada Sprint en la que, con una duración máxima de 4 horas el equipo presenta al propietario del producto, clientes, usuarios y/o gestores el incremento construido en el Sprint.

En esta el propietario del producto obtiene una revisión del progreso del sistema. Esta reunión le ofrece a intervalos regulares el ritmo de construcción del sistema y la trayectoria que va tomando la visión del producto. Al ver el incremento funcionando, el propietario del producto, y el equipo en general obtienen Feedback clave para evolucionar y dar valor al Product Backlog.

2.2.4.2.3 ROLES

Los roles o responsabilidades dentro de un proyecto desarrollado con Scrum dependen directamente de tres condiciones: las características del entorno como la organización y el proyecto, el conocimiento de la metodología de trabajo y la asignación de responsabilidades del producto, del desarrollo o del funcionamiento del proceso.

Entonces, por un lado existe el propietario del producto. Esta es una única persona, conocedora del entorno de negocio del cliente y de la visión del producto. Representa a todos los interesados en el producto final y es el responsable del Product Backlog. Es responsable de decidir cómo debe ser el resultado final, del lanzamiento y del retorno de la inversión. En desarrollos internos puede ser el Product Manager, o responsable de marketing.

Por otro lado, se encuentra el responsable del desarrollo o Scrum Master. Este se encarga de garantizar el funcionamiento de los procesos y metodologías que se emplean. Por tanto, el Scrum Manager designa las responsabilidades de funcionamiento del modelo.

2.2.4.3 Herramientas

2.2.4.3.1 GRÁFICO BURN-UP

Es una herramienta de planificación y seguimiento del propietario del producto, que muestra en un gráfico muy simple el plan general de desarrollo del producto, y la traza de su evolución. Se confecciona con: la estimación de esfuerzo prevista en el Product Backlog y la velocidad del equipo.

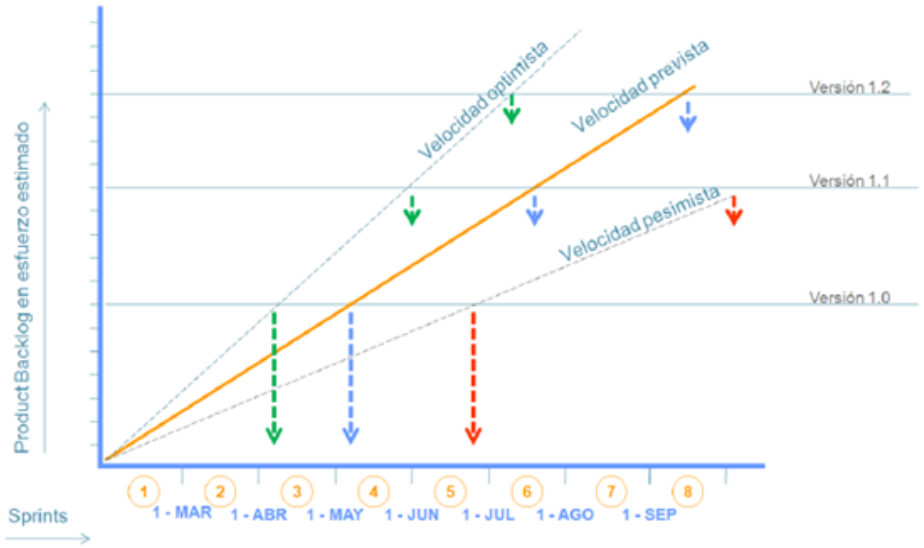


FIGURA 1

El eje Y representa el esfuerzo, y sobre él se marcan los hitos de versiones previstas en el Product Backlog. El eje X representa el tiempo de desarrollo con las fechas de los Sprints previstos.

En el área del gráfico se proyecta la línea que representa la velocidad de desarrollo del equipo. Este dato se obtiene a partir del histórico de velocidad desarrollada por el mismo equipo en proyectos o Sprints anteriores.

Si no se tiene información histórica, se puede comenzar utilizando tiempo real como unidad para el esfuerzo y la velocidad (horas o días reales) y suponer

como velocidad del equipo un tercio del tiempo disponible de trabajo. La intersección de los hitos en Y del esfuerzo previsto para una versión, con la línea de velocidad prevista, proyecta sobre X la fecha en la que previsiblemente estarán desarrolla la versión. Si las estimaciones se realizan considerando valores optimistas y pesimistas de velocidad, o de esfuerzo necesario, se pueden obtener valores probables.

2.2.4.3.2 GRÁFICO BURN-DOWN

Herramienta de seguimiento para el equipo. Muestra el avance del Sprint día a día y revela de forma temprana posibles desviaciones.

Es un gráfico cartesiano donde se representa en el eje X los días laborables del Sprint, y en el eje Y la cantidad de esfuerzo estimada. Dicho grafico al ser actualizado día a día por cada miembro del equipo con el estado de las tareas asignadas, permite determinar la cantidad de producto que falta por construir para llegar al objetivo del Sprint.

La evolución ideal del Sprint se representaría por la línea punteada en gris de la figura. La línea naranja muestra la evolución real diaria. El recorrido sobre la diagonal es síntoma de problemas o sub-estimación del Sprint Backlog. El recorrido bajo la diagonal es síntoma de sobre-estimación del Product Backlog.

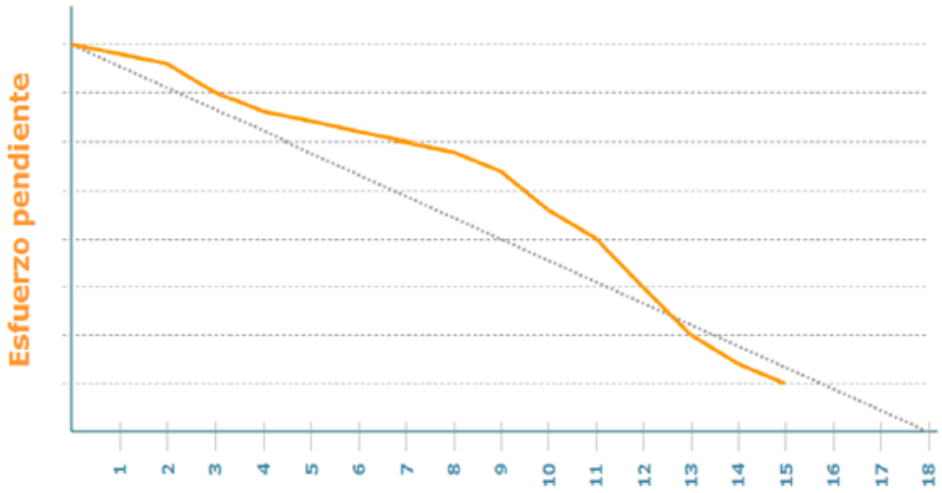


FIGURA 2

2.2.4.3.3 ESTIMACIÓN DE PÓQUER

Es un protocolo de trabajo ágil, que resulta útil en las reuniones en las que el equipo debe estimar el esfuerzo de las tareas con criterio de juicio de expertos. En estos casos es frecuente entrar en dilatadas exposiciones que no llegan a conclusiones concretas. Para evitarlas, y ayudar a conducir la reunión de

planificación de Sprint, James Grenning ideó este juego de planificación en el que cada participante dispone de 8 cartas con los números necesarios para representar la cantidad de días trabajo que puede llevar realizar una tarea.

Originalmente se ideó para realizar estimaciones utilizando Extreme Programming. Sobre este modelo ágil se utilizaba como unidad de esfuerzo: días de trabajo de cada par de programadores y tareas de una dimensión mínima de medio día, y máxima de 10. Las tareas de mayor dimensión se dividen en otras menores. Por esta razón el modelo original emplea las ocho cartas de la figura 3: con los siete números se puede representar cualquier estimación entre 10 días y medio día; y el infinito se emplea para indicar que la tarea necesitaría más de 10 días de trabajo, y por tanto debe dividirse en otras menores.

El número de cartas que debe tener cada participante, y los números que representan dependerán de la unidad de estimación empleada por el equipo: puntos de funcionalidad (Story Points), días u horas teóricas o reales.

Lo relevante no es el número de cartas, la unidad de medida que debe emplearse, o el tamaño máximo de una tarea, sino trabajar con los parámetros que resulten más apropiados para el equipo, respetando los siguientes principios:

- ✓ No definir tareas demasiado grandes. Esto dificulta la precisión de las estimaciones y la identificación de riesgos. Un criterio válido, si el equipo no dispone de experiencia previa, es descomponer en otras menores las que tengan una duración mayor de 7 días ideales de trabajo.
- ✓ No definir tareas con duraciones inferiores a medio día ideal de trabajo.
- ✓ Utilizar al misma unidad de medida

FIGURA 3

2.2.5 XP

Extreme Programming es un metodología exitosa por hacer hincapié en la satisfacción del cliente. En vez de entregar todo lo que se pueda desarrollar en una fecha futura lejana, este proceso proporciona el software que se necesite, cuando se necesite. Permite a los desarrolladores responder satisfactoriamente a los cambios en los requerimientos, incluso en etapas tardías del ciclo de vida.

Extreme Programming mejora un proyecto de software en aspectos esenciales, como son la comunicación, la simplicidad y el feedback.

- ✓ **Simplicidad**: Se desarrolla lo que sea necesario y se pidió, pero no más. Esto maximizará el valor creado por la inversión realizada hasta la fecha. Se darán pequeños pasos simples para llegar al objetivo y mitigar los fallos que se produzcan.
- ✓ **Comunicación**: Todo el mundo es parte del equipo y que se comunican cara a cara todos los días. Se trabaja en equipo en todo, desde los requerimientos de código hasta la implementación.
- ✓ **Feedback**: Luego de cada iteración, se entregará al cliente el desarrollo realizado. Se muestra el software temprano y con frecuencia y luego, se adaptarán los cambios que el cliente crea necesarios.

Los programadores se comunican constantemente con sus clientes y compañeros programadores, mantienen su diseño simple y limpio, reciben

retroalimentación probando su software a partir del primer día, entregan el sistema a los clientes tan pronto como sea posible e implementan cambios como los sugiere el cliente.

2.2.5.1 Proceso y Fases

Un proyecto XP tiene éxito cuando el cliente selecciona el valor de negocio a implementar basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración.

El ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la entrega (Release), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

2.2.5.1.1 FASE I: EXPLORACIÓN

En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.

Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

2.2.5.1.2 FASE II: PLANIFICACIÓN DE LA ENTREGA

En esta fase el cliente establece la prioridad de cada funcionalidad mientras los programadores realizan una estimación del esfuerzo necesario para cada una de ellas. Se acuerda el alcance de la primera entrega y se determina un cronograma

en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días.

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración.

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación.

2.2.5.1.3 FASE III: ITERACIONES

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede establecer la base de la arquitectura del sistema. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura.

Los elementos que deben tomarse en cuenta durante la elaboración del Plan de la Iteración son: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior.

Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores.

2.2.5.1.4 FASE IV: PRODUCCIÓN

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

2.2.5.1.5 FASE V: MANTENIMIENTO

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que se desarrollan

nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.

2.2.5.1.6 FASE VI: MUERTE DEL PROYECTO

Es cuando el cliente no tiene más historias para incluir en el sistema. Esto requiere que se satisfagan las necesidades del cliente en todos sus aspectos (considerando rendimiento y confiabilidad del sistema). Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

2.2.5.2 Características

La metodología XP tiene las siguientes características:

2.2.5.2.1 EQUIPO COMPLETO

Las partes involucradas de un proyecto XP forman un único equipo. Este está compuesto por: el cliente, quien introduce los requerimientos del proyecto, establece las prioridades del desarrollo, y guía al proyecto. También se encuentra el equipo de desarrollo, que se encarga de construir el producto, los verificadores (*testers*) que ayudan a definir las pruebas de aceptación del cliente, los analistas que relevan los requerimientos y también un *manager*, que se encarga de la comunicación externa y coordina las distintas actividades. Estos roles no son exclusivos de una persona, sino que todos en el equipo contribuyen de la manera que pueden.

2.2.5.2.2 PLANIFICACIÓN

La planificación en XP responde a dos preguntas clave del desarrollo de software: la predicción de lo que va a llevarse a cabo en la fecha prevista, y la determinación de qué hacer a continuación.

2.2.5.2.3 PLANIFICACIÓN DE LA ENTREGA

El cliente presenta los requerimientos que quiere para su producto, y el equipo de desarrollo, luego de relevarlos, estima su dificultad. Luego, cuando el cliente obtiene las estimaciones de los requerimientos, y el costo de su desarrollo, y conociendo la importancia entre cada requerimiento, establece un plan para el proyecto ordenado según las prioridades que considere más necesarias. Las primeras planificaciones de entrega son muy imprecisas, debido a que ni las prioridades del cliente, ni las estimaciones efectuadas, son sólidas. De todas maneras, el primer plan es suficiente como para tomar decisiones, y para que el equipo completo revise de forma detallada el plan.

2.2.5.2.4 PLANIFICACIÓN DE LA ITERACIÓN

Aquí es donde se establece qué es lo que se va a desarrollar en cada par de semanas. El equipo itera cada dos semanas, construyendo software útil para ser entregado al cliente. Durante esta etapa, el cliente describe los requerimientos deseados para la iteración, y luego, los programadores los dividen en tareas, y estiman el costo de desarrollarlas. Las iteraciones anteriores permiten estimar mejor el costo y el tiempo de desarrollo. El equipo se compromete a desarrollar cada tarea de la iteración.

2.2.5.2.5 PRUEBAS AUTOMATIZADAS

Para aceptar el desarrollo de un requerimiento, se realizan pruebas de aceptación automatizadas, para demostrar que el requerimiento funciona correctamente. Una vez que la prueba se ejecuta correctamente, el equipo la ejecuta permanentemente de ahí en adelante, para mostrar un avance continuo, y no un retroceso en el desarrollo.

2.2.5.2.6 ENTREGAS

El equipo de desarrollo, cuando finaliza la iteración, entrega software probado y funcionando, que a su vez, entrega el valor elegido por el Cliente, en cada iteración. El Cliente puede usar este software para cualquier propósito, sea como evaluación o incluso liberarlo para los usuarios finales.

2.2.5.2.7 PROGRAMACIÓN DE A PARES

En XP todo el software productivo se escribe en pareja, dos programadores sentados lado a lado en una misma computadora. Esta práctica asegura que todo el código productivo fue revisado por al menos otro programador, y genera mejores diseños, mejores pruebas y mejor código.

Los estudios sobre la programación de a pares muestran que las parejas producen mejor código en aproximadamente el mismo tiempo que un programador trabajando solo.

2.2.5.2.8 TDD

Los equipos de XP practican Desarrollo Guiado por Pruebas (TDD), trabajando en ciclos muy cortos que consisten en agregar una prueba, y después hacerla funcionar.

Las "pruebas del programador" o "pruebas unitarias" son una herramienta que permite al programador verificar automáticamente el código generado antes de ser administrado en el repositorio central,

2.2.5.2.9 PROPIEDAD COLECTIVA DEL CÓDIGO

Cualquier programador puede cambiar cualquier parte del código en cualquier momento. Esta práctica motiva a todos a contribuir con nuevas ideas en todos los segmentos del sistema, evitando a la vez que algún programador sea imprescindible para realizar cambios en alguna porción de código.

2.2.5.2.10 REFACTOR

El proceso de refactor se enfoca en eliminar la duplicación, y en incrementar la cohesión del código, disminuyendo el acoplamiento. De esta manera los equipos comienzan con un diseño simple y bueno, y terminan con un diseño simple y bueno para el software. Esto les permite mantener la velocidad, y en general suele acelerar la misma a medida que el proyecto avanza.

El refactor se apoya en pruebas completas que aseguran que, a medida que el diseño evoluciona, nada se rompa. Las pruebas del cliente y unitarias son críticas para permitir el refactor.

2.2.5.2.11 INTEGRACIÓN CONTINUA

Los equipos mantienen integrado al sistema todo el tiempo. La integración infrecuente lleva a problemas serios en el proyecto de software. Al momento de integrar aparecen problemas que no se detectaron en ninguna de las pruebas del software sin integrar.

2.2.5.2.12 ESTÁNDARES DE CÓDIGO

Los equipos XP usan un estándar de código en común, de manera que el código del sistema se vea como si fuera escrito por una única persona muy competente. No importa mucho el estándar en sí mismo: lo importante es que el código se vea familiar, para permitir la propiedad colectiva.

2.2.5.2.13 METÁFORA DEL SISTEMA

Los equipos XP desarrollan una visión común sobre cómo funciona el programa, que llamamos la "metáfora". La metáfora es una descripción evocativa simple sobre cómo funciona el programa.

2.2.5.3 Roles

Los roles en XP según la propuesta original de Beck son:

2.2.5.3.1.1 PROGRAMADOR

El programador escribe las pruebas unitarias y produce el código del sistema. Define las tareas que conlleva cada historia de usuario, y estima el tiempo

que requerirá cada una. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.

2.2.5.3.1.2 CLIENTE

El cliente escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.

2.2.5.3.1.3 ENCARGADO DE PRUEBAS (TESTER)

El encargado de pruebas ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

2.2.5.3.1.4 ENCARGADO DE SEGUIMIENTO (TRACKER)

El encargado de seguimiento proporciona realimentación al equipo en el proceso XP. Su responsabilidad es verificar el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, comunicando los resultados para mejorar futuras estimaciones.

También realiza el seguimiento del progreso de cada iteración y evalúa si los objetivos son alcanzables con las restricciones de tiempo y recursos presentes. Mantiene contacto directo y continuo con el equipo de desarrollo evaluando la situación. Determina cuándo es necesario realizar algún cambio para lograr los objetivos de cada iteración, involucrando al Gestor, Entrenador, o al Cliente en el caso considerarlo conveniente.

2.2.5.3.1.5 ENTRENADOR (COACH)

Es responsable del proceso global. Es necesario que conozca a fondo el proceso XP para proveer guías a los miembros del equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.

2.2.5.3.1.6 GESTOR (BIG BOSS)

Es el dueño del equipo y sus problemas. Es una persona experta en tecnología y labores de gestión. Su labor esencial es de coordinación. Agenda reuniones (planes de iteración, agenda de compromisos, etc.), verifica que se realicen de manera adecuada y registra lo referente a las mismas.

No le dice al grupo lo que tiene que hacer (el Cliente y el plan de iteración lo hacen), cuando hacerlo (la agenda de compromisos lo hace), ni verifica el avance de las tareas (Tracker).

2.2.6 Lean Startup

2.2.6.1 Definición

El método Lean Startup toma su nombre de la revolución del Lean manufacturing que Taiichi Ohno y Shigeo Shingo desarrollaron en Toyota. El pensamiento Lean altera radicalmente la forma de organizar las cadenas de oferta y los sistemas de producción de cualquier empresa tradicional. Entre sus principios están el diseño del conocimiento y la creatividad de los trabajadores, la reducción de las dimensiones de los lotes, la producción *just-in-time* y el control de inventarios, y la aceleración del tiempo del ciclo. Enseña al mundo las distintas actividades que crean valor e incorporan calidad a los productos.

El método Lean Startup aconseja a la gente que comience empiece su productividad de otra forma. Como las Startups suelen producir accidentalmente algo que nadie quiere, no importa si lo hacen dentro del tiempo pactado o ciñéndose en el presupuesto. El objetivo de una Startup es averiguar qué debe producirse, aquello que los consumidores quieren y por lo que pagarán, tan rápidamente como sea posible. El método Lean Startup es una nueva forma de ver el desarrollo de productos innovadores que enfatiza la rápida iteración y la comprensión de los consumidores, una enorme visión y una gran ambición, todo al mismo tiempo. El método está diseñado para enseñar a conducir una Startup. En lugar de hacer planes complejos basados en muchas asunciones, se pueden hacer ajustes constantes con un circuito de feedback de Crear-Medir-Aprender.

Las Startups tienen un objetivo, un destino en mente: crear un negocio próspero que cambie el mundo; esa es la visión de un Startup. Para ello, las Startups emplean una estrategia, que incluye un modelo de negocio, un mapa de productos, un enfoque relativo a los socios y los competidores e ideas sobre quiénes serán los consumidores. El producto es el resultado final de esta estrategia.

2.2.6.2 Aprendizaje

Lean Startup se basa en el concepto de aprendizaje validado. El aprendizaje validado no es una racionalización hecha a posteriori o una buena historia inventada para esconder un fracaso, sino que es un proceso para demostrar empíricamente que un equipo ha descubierto información valiosa sobre las posibilidades presentes y futuras del negocio. Es más concreto, riguroso y rápido que la previsión de mercado o la planificación clásica.

El trabajo se basa en encontrar una síntesis entre la visión y lo que los clientes aceptarían; no se trata de capitular ante lo que los consumidores creían que querían, ni de decir a los consumidores lo que deberían querer. Cuando se empieza a entender a los consumidores, se es capaz de mejorar los productos. A medida que se obtiene este feedback con los clientes, los indicadores del negocio se cambian, adaptándose a lo que realmente quiere el mercado.

Con el cambio de indicadores de negocio, se obtiene una estrategia mejor, logrando esfuerzos de desarrollo de producto más productivos, no porque se estuviera trabajando más duro sino porque se hace de una forma más inteligente, a partir de las necesidades reales de los clientes. Los cambios positivos en los indicadores se convierten en la validación cuantitativa de que el aprendizaje es real. Esto es muy importante, ya que se puede mostrar a accionistas, empleados, inversores y a nosotros mismos, que se está haciendo un progreso genuino. También es la forma correcta de ver la productividad en una Startup: no en términos de cuántas novedades se están creando sino en términos de cuánto conocimiento validado se están obteniendo con nuestros esfuerzos.

El aprendizaje es la unidad esencial para medir el progreso de una Startup. Aquel esfuerzo que no es necesario para saber qué quieren los consumidores puede eliminarse. Eso es el aprendizaje validado porque siempre se puede demostrar a través de mejoras en los principales indicadores de la Startup. Por lo tanto, el aprendizaje validado se respalda en datos empíricos que se obtienen de consumidores reales.

Una Startup es un catalizador que transforma las ideas en productos. A medida que los consumidores interaccionan con estos productos, generan feedback y datos. El feedback es tanto cualitativo (como, por ejemplo, si les gusta o no) como cuantitativo (por ejemplo, cuánta gente lo usa y lo encuentra valioso). Los productos que crean las Startups son experimentos; el aprendizaje sobre cómo crear un negocio sostenible es el resultado de dichos experimentos. Para las Startups, la información es más importante que el dinero, los premios o las menciones en la prensa, porque puede condicionar y reestructurar el siguiente conjunto de ideas.

Podemos ver este proceso de tres fases en el siguiente esquema:

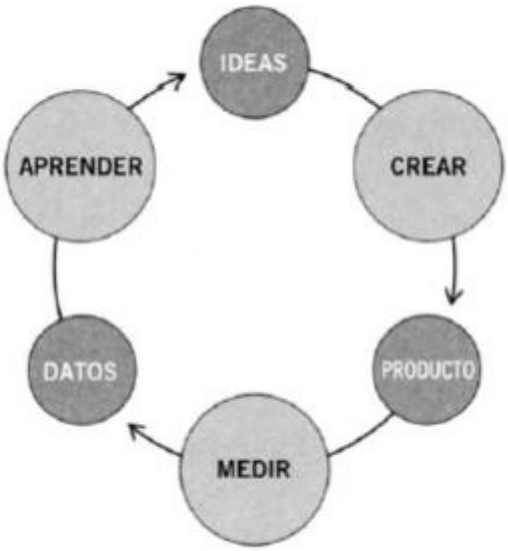


FIGURA 4

El circuito de feedback de información Crear-Medir-Aprender es el centro del modelo del método Lean Startup.

Mucha gente tiene una formación profesional que enfatiza uno de los elementos de este circuito de feedback de información. Para los ingenieros, se trata de aprender cómo construir de la forma más eficiente posible. Algunos directivos son expertos en diseñar y estudiar estrategias sobre la pizarra. Muchos emprendedores centran su energía en los elementos individuales: tener la mejor idea de producto o el producto inicial mejor diseñado, u obsesionarse sobre los datos y los indicadores.

La verdad es que, por sí sola, ninguna de estas actividades es de primordial importancia. Por el contrario, se necesita centrar las energías en minimizar el tiempo total a lo largo de este circuito de feedback de información. Ésta es la esencia de dirigir una Startup.

Centrando nuestras energías en el conocimiento validado se podrá evitar gran parte del despilfarro que asedia las Startups hoy en día. Como en el Lean manufacturing, el aprendizaje sobre dónde y cuándo invertir energía tiene como resultado el ahorro de tiempo y dinero.

Para aplicar el método científico a una Startup, se necesita identificar las hipótesis que hay que probar. Las dos asunciones más importantes son la hipótesis del valor y la hipótesis del crecimiento. Éstas dan lugar a los indicadores de ajuste que controlan el motor de crecimiento de la Startup. Cada iteración de una Startup es un intento de revolucionar el motor para ver si funciona. Cuando se comprueba que funciona, el proceso se repite, cambiando a velocidades cada vez más y más altas.

Una vez aclaradas estas asunciones el primer paso es entrar en la fase de construcción tan rápido como sea posible con un producto mínimo viable (MVP). El MVP es aquella versión del producto que permite dar una vuelta entera al circuito de Crear-Medir-Aprender con un mínimo esfuerzo y el mínimo tiempo de desarrollo. Al MVP le faltan muchos elementos que pueden ser esenciales más adelante. Sin embargo, crear un MVP requiere un trabajo extra: debemos ser capaces de medir su impacto. Por ejemplo, es inadecuado crear un prototipo que sólo se evalúa mediante controles internos de calidad realizados por ingenieros y diseñadores. Necesitamos ponerlo delante de consumidores potenciales para evaluar sus reacciones. También podemos necesitar venderles el prototipo.

Cuando se entra en la fase de Medir, el mayor reto será determinar si los esfuerzos de desarrollo del producto están produciendo un progreso real.

Si se está creando algo que nadie quiere, no importa si se hace a tiempo y dentro del presupuesto.

Finalmente, y lo más importante, está el pivote. Si se descubre que una de las hipótesis es falsa, ha llegado el momento de hacer un gran cambio hacia otra hipótesis estratégica. El método Lean Startup crea empresas eficientes en el uso del capital porque permite a las Startups reconocer pronto cuando es el momento de pivotar, consiguiendo un menor despilfarro de tiempo y dinero. A pesar de llamarse: Circuito de Feedback, a: Crear-Medir-Aprender por sus actividades, en verdad la planificación trabaja en orden inverso: nos imaginamos qué necesitamos aprender, usamos la contabilidad de la innovación para descubrir qué debemos medir para saber si estamos obteniendo aprendizaje validado, y entonces averiguamos qué producto necesitamos crear para llevar a cabo experimentos y obtener estas medidas. Todas las técnicas están diseñadas para minimizar el tiempo total del circuito de feedback de Crear-Medir-Aprender.

Un producto mínimo viable (MVP) ayuda a los emprendedores a empezar con el proceso de aprendizaje lo más rápido posible. Es necesariamente el producto más pequeño que se pueda conseguir para entrar lo más rápido posible en el circuito de feedback de Crear-Medir-Aprender con el mínimo esfuerzo. Es totalmente al contrario que cualquier tradicional desarrollo de productos (que normalmente requiere un período de incubación y de reflexión largo y se esmera en alcanzar la perfección del producto), el objetivo del MVP es empezar el proceso de aprendizaje, no terminarlo. A diferencia de un prototipo o una prueba de concepto, un MVP no sólo está diseñado para responder las cuestiones técnicas y de diseño. Su objetivo es probar las hipótesis fundamentales del negocio.

2.2.6.3 El papel de la calidad y el diseño en un MVP

Uno de los aspectos más irritantes del MVP es el reto que supone para las nociones tradicionales de calidad. Los procesos de producción modernos confían en la alta calidad como forma de ganar eficiencia. Operan usando el famoso dictado de W. Edwards Deming de que el cliente es la parte más importante del

proceso de producción. Esto significa que debemos centrar nuestras energías exclusivamente en producir algo que el consumidor perciba como valioso. Permitir un trabajo descuidado del proceso inevitablemente conduce a una variabilidad excesiva. La variabilidad en el proceso tiene como resultado productos de calidad variable a los ojos del consumidor que, en el mejor de los casos, requerirá repetir el trabajo y, en el peor, llevará a la pérdida del consumidor. Los negocios modernos y las filosofías de ingeniería se centran en producir experiencias de alta calidad para los consumidores como principio fundamental; ésta es la base del Six Sigma, el Lean manufacturing, el pensamiento de diseño, la programación extrema y el movimiento de la artesanía del software.

Estas discusiones sobre calidad presuponen que la empresa, o los desarrolladores ya saben qué atributos del producto son los que el consumidor percibirá que valen la pena. En una Startup, ésta es una asunción arriesgada de sostener. A menudo no sabemos con seguridad ni siquiera quién es el consumidor. Así, para las Startups, Eric Ries cree en el siguiente principio de calidad:

> "Si no sabemos quién es el consumidor, no sabemos qué es la calidad" [1]

Incluso un MVP de baja calidad puede ser útil para crear un producto de una enorme calidad. Aunque los MVP a veces parecen ser de baja calidad para los consumidores, se debería usar esto como oportunidad para aprender cuáles son los atributos que importan a los consumidores. Esto es infinitamente mejor que hacer meras especulaciones o diseñar estrategias sobre la pizarra, porque aporta una base teórica sólida a partir de la cual crear productos en el futuro.

A veces, sin embargo, los consumidores reaccionan de forma un poco diferente. Muchos productos famosos fueron lanzados en un estado de baja calidad y a los consumidores les encantaron.

Un MVP requiere el coraje de probar las asunciones de uno mismo. Si los consumidores reaccionan como se espera que lo hagan, se puede tomar esto como una confirmación de que nuestras asunciones eran correctas. Si se lanza un producto diseñado modestamente y los consumidores (incluso los primeros usuarios) no pueden descubrir cómo usarlo, esto confirmará que necesitamos invertir en un diseño mejor. Pero siempre hay que preguntarse: ¿qué pasa si no les importa el diseño de la misma manera que a nosotros? Así, el método Lean Startup no se opone a crear productos de alta calidad sólo con el objetivo de ganar clientes. Debemos desear dejar de lado nuestros estándares profesionales para empezar con el proceso de aprendizaje validado lo antes posible. Pero de nuevo, esto no significa trabajar de una forma chapucera o indisciplinada. Cuando vaya a crear el propio producto mínimo viable, hay que eliminar cualquier elemento, proceso o esfuerzo que no contribuya directamente al aprendizaje que se está buscando.

2.3 Complementos a Procesos Existentes

2.3.1 Normas

La ISO (International Standarization Organization) es la entidad internacional encargada de conformar normas de fabricación, comercio y comunicación en todo el mundo. Este organismo se encarga de estandariza procedimientos, técnicas, practicas, etc. útiles a la hora de construir un nuevo producto por ejemplo. En el proyecto no centraremos en la norma ISO 9126.

La norma ISO 9126 propone un modelo de calidad para medir especialmente un producto de software, teniendo en cuenta tres aspectos: calidad interna, calidad externa y calidad de uso.

En cuanto a la calidad interna, nos referimos a las mediciones que se obtienen del producto tomando en cuenta aspectos del desarrollo del producto como las definiciones de los requisitos, diseño y código, sin considerar el comportamiento externo.

En la calidad externa, nos referimos a los aspectos medibles del producto al ser ejecutado en etapa de verificación. Considerando como importante a las características y atributos que influencian al producto externamente desde el entorno de las pruebas.

Estos aspectos los podemos identificar como:

- ✓ Funcionalidad.
- ✓ Finalidad.
- ✓ Usabilidad.
- ✓ Eficiencia.
- ✓ Mantenibilidad.
- ✓ Portabilidad.

Al referirnos a la calidad del uso, nos referimos a aquellos aspectos de percepción que tienen los usuarios a interactuar con el producto en un escenario en específico.

Los aspectos de calidad que se toman en cuenta para evaluar el uso son:

- ✓ Productividad.
- ✓ Efectividad.
- ✓ Seguridad.
- ✓ Satisfacción.

Las medidas que se acaban de tomar no son cuantificables, pero lograr cuantificarlas de alguna manera es necesario para poder utilizarlas de manera objetivas.

En la norma ISO9000:2000 se exigen solamente documentación en 6 aspectos del proceso de negocio. El llevar estas documentaciones como mínimo proporciona a las organizaciones un enfoque estructurado al progreso de la organización.

En esta norma se menciona que los requisitos de la gestión de calidad se pueden utilizar por la organización para aumentar la satisfacción de sus clientes al satisfacer los requisitos planteados por el mismo cliente y por la disposición obligatoria de ser aplicados.

2.3.2 Metodología SIT

Systematic Inventive Thinking (SIT) es una metodología de pensamiento desarrollada en Israel a mediados de la década de 1990. Derivada de la disciplina TRIZ de ingeniería, SIT es un enfoque práctico para la creatividad, la innovación y la resolución de problemas, que se ha convertido en una metodología bien conocida para la Innovación.

2.3.2.1 Historia de la Metodología SIT

SIT trabaja con dos áreas principales de la creatividad: la creación de nuevas ideas, y la resolución de problemas. En la década de 1970, los investigadores del campo de la Psicología Cognitiva establecieron un criterio cuantitativo para medir la creatividad: la persona creativa se definió como una persona con un gran flujo de ideas. Un alto porcentaje de las ideas por unidad de tiempo se considera un indicador de creatividad. Este enfoque dio lugar a una serie de métodos para desarrollar la creatividad basada en la suposición de que un aumento cuantitativo de las ideas necesariamente logra una mejora cualitativa. Tales métodos ampliamente conocidos como lluvia de ideas, Synectics, estimulación al azar y el pensamiento lateral (identificado con Edward de Bono) se pueden remontar a este enfoque. Estudios más recientes revelan la aparición de un enfoque diferente. Estos estudios muestran que la principal dificultad que enfrentan los solucionadores de problemas no está generando una gran cantidad de ideas sino la elaboración de ideas originales. Se descubrió que un gran flujo de ideas no necesariamente conduce a la creación de ideas originales y, además, que el tiempo dedicado a las ideas comunes puede en realidad obstaculizar la creatividad y el pensamiento innovador. Estos descubrimientos han llevado a un nuevo enfoque que sostiene que los resultados originales e interesantes surgen de pensamiento organizado y procesos estructurados en lugar de la generación aleatoria de ideas. En este enfoque, la originalidad reemplaza cantidad como criterio dominante. Este enfoque organizado o estructurado para la generación de ideas es el punto de partida para el pensamiento inventivo sistemático (SIT).

LA HERENCIA TRIZ

SIT es un descendiente de la obra de Genrich Altshuller, un ingeniero ruso que analizó más de 200.000 patentes para identificar los 40 principios inventivos comunes de su fórmula, llamada TRIZ. El descubrimiento principal de Altshuller fue que las soluciones creativas incorporan una eliminación de un conflicto en el estado del problema. Un conflicto es un estado en el que un parámetro se debe cambiar, con el fin de obtener algún beneficio, pero cambiar ese parámetro provoca un deterioro de otro parámetro importante.

El examen de numerosos inventos hizo posible asignar cada conflicto con un conjunto de posibles sugerencias o estrategias sobre cómo abordar la solución al problema. Hay tres tipos de pistas que se utilizan: los principios, las normas y los efectos físicos. Hay 40 principios; cada uno ayuda con la definición de estrategias de alto nivel para resolver el problema. Hay 70 normas, que son las ideas más elaboradas sobre la base de las soluciones del pasado colectivo. Hay una base de conocimiento de aproximadamente 400 efectos físicos que van a través de los aspectos físicos, químicos y geométricos, indexada de acuerdo a las funciones que cada efecto puede llevar a cabo.

DE TRIZ A SIT

El paso de TRIZ para SIT fue motivado por el deseo de crear un método que sea más fácil de aprender y retener (logrado a través de un menor número de reglas y herramientas), más universal en su aplicación (que se logra mediante la eliminación de herramientas de ingeniería específicas) y manteniendo la resolución de problemas con un framework de inventiva real (el principio del mundo cerrado).

2.3.2.2 Principios

EL MUNDO CERRADO

La condición de Mundo Cerrado es crucial para la metodología del SIT. El primer paso en el uso de SIT es definir el mundo del problema. Una vez definido, el solucionador de problemas sabe que todos los bloques de construcción para la solución están ahí delante de él y que la solución se limita a exigir la reorganización de los objetos existentes. Esto añade una gran concentración y energía al método. Se estipula que el desarrollo de un nuevo producto - o al abordar un problema - uno debe utilizar sólo los elementos ya existentes en el producto / problema, o en el entorno inmediato. Esta condición nos obliga a depender de los recursos a nuestra disposición, en lugar de "importación" de nuevos recursos externos para la solución. La condición de Mundo Cerrado a menudo provoca resistencia, ya que va en contra de algunas de las intuiciones más comunes sobre el pensamiento creativo, especialmente la noción omnipresente de "pensar fuera de la caja". La afirmación de "pensar fuera de la caja" es que con el fin de producir ideas que son nuevas y diferentes, de alguna manera se tienen que mover más allá de los

patrones de pensamiento normales, a un universo situado fuera de la caja metafórica. La condición de Mundo Cerrado, por el contrario, obliga al pensador a encontrar una solución creativa por más que se limita el espacio de posibilidades. Dado que el ámbito de posibilidades se limita artificialmente no hay más remedio que reconsiderar las relaciones entre los elementos que se encuentran dentro del problema y prestar más atención a ellos. Por lo tanto, la condición de Mundo Cerrado nos permite llegar a soluciones que son innovadoras (diferente de la habitual) y simple (ya que a partir de elementos existentes y conocidos).

LA FUNCIÓN SIGUE A LA FORMA

Un término acuñado por Ronald Finke, la función sigue a la forma a menudo se considera un proceso "al revés" en el que el punto de partida para pensar en nuevas ideas es la base de recursos existentes en lugar de las necesidades específicas que se han identificado en el mercado. Estas necesidades, sin embargo, nunca se ignoran, simplemente se introducen en una etapa posterior. El proceso comienza con un producto existente (o servicio), continúa con la manipulación de forma sistemática para crearlo, SIT llama un "producto virtual" y sólo entonces examina si satisface las necesidades de los clientes existentes o potenciales.

CAMINO DE LA RESISTENCIA

En la naturaleza, el agua cayendo sobre una montaña siempre seguirá el camino de menor resistencia - la ruta más fácil. El pensamiento, también, nuestras mentes tienden a tomar el camino de menor resistencia, esas avenidas más familiares para nosotros. Al hacerlo, es difícil llegar a las ideas nuevas para nosotros o para nuestros competidores. SIT alienta un enfoque a la trayectoria contraria a la intuición.

2.3.2.3 El enfoque de Desarrollo de Nuevos Productos

SIT se enfoca en el desarrollo de nuevos productos mediante la identificación y aplicación de determinados regímenes bien definidos derivados de un análisis histórico de las tendencias basadas en el producto, los patrones o las llamadas plantillas. Estas plantillas pueden contribuir a la comprensión y predicción de la nueva aparición de productos. La invención de nuevos productos ha implicado tradicionalmente métodos que estimulan la generación de un gran número de ideas. La idea de que los beneficios de la generación de un gran número de ideas son mayores que los costos se remonta a los primeros estudios en la materia. En vista del hecho de que este proceso tiende a ser muy complejo y formalizado, los que participan en la generación de nuevas ideas puede buscar maneras de ser más productivos a medida que avanzan de una tarea a otra ideación. Algunos pueden tener éxito en la identificación de patrones de invención que son comunes a los diferentes contextos y las aplican en una determinada categoría de productos, o incluso tratar de aplicarlos a otras categorías de productos. Las personas que adoptan tal estrategia cognitiva pueden esperar para obtener una ventaja sobre los demás que tratan a cada tarea como nueva y sin

relación con la ideación pasada. SIT se basa en la tesis de que ciertos patrones son identificables, verificables objetivamente, aplicados extensamente y se pueden aprender y estos patrones, denominados plantillas, pueden servir como una herramienta de facilitación que los canales del proceso de ideación, permitan a las personas a ser más productivas y centradas.

El proceso habitual de desarrollo de nuevos productos comienza con una definición de la necesidad del mercado. Esto se hace con base en la intuición o en el análisis de mercado, grupos de discusión, etc. Después de definir las necesidades, se inicia el desarrollo de productos para hacer frente a estas necesidades. Este proceso se denomina "La función sigue a la Forma", como la forma del nuevo producto se deriva de la función que debe cumplir. Este proceso tiene algunas desventajas:

La mayoría de los clientes tienen dificultades para pensar en las necesidades o productos, que no existen. Esto es particularmente cierto para las necesidades que no son de vital importancia. Por ejemplo: ¿cuántos clientes pensaron en la necesidad de un reproductor de casetes transportable compacto? ¿Cuántos clientes pensaban en la posibilidad de utilizar internet como medio para realizar llamadas telefónicas?

Para encontrar a los clientes que no piensan en nuevas necesidades / productos, se necesitan estudios grandes y muy caros. Pero incluso si se tiene éxito en la búsqueda de esas personas, lo más probable es que no estarán dispuestos a compartir sus buenas ideas de forma gratuita.

Si la necesidad es clara y fácil de definir, lo más probable es que, al menos algunos de sus competidores ya lo han definido y están en el proceso de abordar la necesidad.

Con el fin de superar estos problemas, el método SIT sugiere comenzar el proceso de desarrollo de productos desde el propio producto. La aplicación de herramientas de pensamiento sistemático en el análisis del producto puede dar lugar a nuevos productos potenciales o para una definición de las nuevas necesidades. Las ventajas de este método son los siguientes:

El proceso requiere sólo una cantidad limitada de horas y se lleva a cabo en la empresa.

La aplicación del método proporciona muchas nuevas ideas y una definición de muchas nuevas necesidades potenciales.

Uno de los elementos importantes de la metodología SIT es caracterizar el sistema y las variables ambientales. Después de haber definido estas variables, se les pide a los participantes a examinar la correlación entre ellos, y para examinar el impacto de la manipulación de una o más de las variables de producto sobre el uso potencial del "nuevo" producto; por ejemplo cómo este cambio afecta a la

correlación entre el producto y el medio ambiente y quien quisiera usar un producto de este tipo.

2.4 Entrevistas

Las entrevistas son otro mecanismo de relevamiento de procedimientos, metodologías, negocios y buenas prácticas. En estas, gracias a lo rico de la comunicación verbal, es posible relevar detalles y secretos en base a experiencias vividas por el entrevistado. Por ello, se acordaron a lo largo del proyecto, tres entrevistas con expertos en construcción de software, en el mercado uruguayo. Estos de alguna manera lograron construir una aplicación exitosa en mercados paralelos al mercado objetivo o conocen en detalle buenas prácticas y metodologías útiles para la construcción de un producto exitoso.

Las entrevistas sirvieron para relevar aspectos y detalles en tres puntos que consideramos fundamentales para la construcción de un producto exitoso: la lógica de negocio del mercado apuntado, la selección del producto y por último la metodología para la construcción del mismo.

En primer lugar, nos reunimos con el Ing. Gabriel López, creador de la empresa SouthLabs (http://southlabs.com/home.aspx). Gabriel construyó un producto exitoso en el mercado móvil de Apple. Si bien no es el mismo mercado al que apuntaba el proyecto, nos resulta interesante poder hablar con él e interiorizarnos con el proceso que siguió para la construcción de una aplicación móvil. Como fue que llegó a la elección del producto construido, y que aspectos de calidad fueron los que tuvieron en cuenta. Se adjuntó un anexo con los puntos más importantes de la entrevista con Gabriel al final del documento.

En segundo lugar, tuvimos una entrevista con el Ing. Martín Cabrera, creador de la empresa Moove-it (http://www.moove-it.com/). La empresa se encarga de la construcción de software bajo metodologías agiles como Scrum. Con Martín tuvimos la chance de hablar más en profundidad de las metodologías agiles, y de procesos de construcción de productos, como Lean Startup. Se adjuntó un anexo con los puntos más importantes de la entrevista con Martin al final del documento.

Las entrevistas con ellos, y los estudios de los distintos procesos, fueron claves para la definición del proceso elegido.

Por último tuvimos una entrevista con Álvaro Azofra, creador de Ironhide Game Studio (www.ironhidegames.com), creador de una de las aplicaciones más descargas del mercado de Apple, Kingdom Rush. La entrevista con Álvaro nos permitió entender cómo funciona el mercado de aplicaciones móviles, y que consideraciones teníamos que tener a la hora de construir el producto. La entrevista nos permitió tomar decisiones en la construcción del producto elegido. Se adjuntó un anexo con los puntos más importantes de la entrevista con Álvaro al final del documento.

3. Fase 2 Definición del proceso

3.1 Introducción

En este capítulo se realiza la definición del proceso a seguir. En cada etapa se definen sus principales actividades, objetivos y roles.

Cada etapa contendrá un conjunto de actividades básicas a seguir y un conjunto de actividades complementarias que pueden ser personalizadas por cada equipo de desarrollo que lleve a cabo el proceso. Como se trata de la primera vez que se ejecuta el mismo, es normal que se encuentren puntos en la definición que luego se modifiquen en la ejecución. Esto se debe a la metodología de trabajo utilizada: investigación, puesta en práctica, relevamiento de resultados.

3.2 Definición

El proceso descripto a continuación es una unificación del marco teórico presentado. Este busca lo mejor de cada una de las metodologías estudiadas, para construir una aplicación exitosa y comercializable en la Tienda de Windows 8.

Una de las características más importante del proceso es su estructura, al igual que en la mayoría de los procesos este se realiza en base a una división en etapas. En este caso se realizan cuatro etapas, las cuales no es posible la ejecución en paralelo por lo que es necesario finalizar cada una de ellas para dar por comenzada la próxima.

3.2.1 Etapas

Etapa 1 - Inicial: primera etapa del proceso, en la cual se busca llegar a la definición de uno o varios productos a construir. Estos deben ser validados por el cliente. Para ello se parte de un conjunto de ideas en base a gustos y necesidades de los usuarios finales. Para relevar estos se realizan análisis de mercados, entrevistas, encuestas, etc.

El objetivo de esta etapa es identificar puertas hacia mercados o negocios en base a lo existente. El resultado de esta etapa es un conjunto de hipótesis a probar junto con un producto que luego de implementado pruebe las mismas.

Etapa 2 - Elaboración: en esta etapa se busca ejecutar las disciplinas: relevamiento de requerimientos, planificación y gestión, diseño, análisis y calidad del proceso. Al final de la misma se estará en condiciones de comenzar a desarrollar el producto. Se planifica el MVP y salidas al mercado en base a fechas claves en el mercado objetivo. Además se priorizan las funcionalidades a desarrollar conformando un plan de desarrollo, un plan de verificación y por ultimo como se realizará la evaluación de la calidad del producto construido.

Etapa 3 - Desarrollo: etapa en la cual se ejecutan las disciplinas de implementación y verificación. El objetivo de esta etapa es obtener un producto cerrado y aprobado por los criterios de aceptación, tanto de la Tienda de Windows, o del mercado objetivo.

Etapa 4 - Liberación: en esta etapa se colocará el producto en la Tienda de Windows. Se llevarán a cabo las disciplinas del mantenimiento evolutivo del producto, y análisis post mortem del proceso y producto. Se espera aprender sobre los resultados obtenidos a partir de las hipótesis asumidas. En esta etapa se evalúa si el proyecto continúa su rumbo o hace un giro hacia otros horizontes.

Las etapas del modelo se agrupan por disciplinas de ingeniería de software. Las actividades de las mismas deben de tener un rol responsable de llevarla a cabo. Estos roles son definidos principalmente en base a la experiencias anteriores adquirida por medio de la investigación realizada.

3.3 Características adquiridas de procesos y metodologías

Al estudiar y analizar procesos de negocios como Lean Startup y metodologías ágiles como Scrum y XP, se realizó un relevamiento de las distintas actividades y artefactos que componen a cada una de ellas. De esta manera se extrae de cada una de estas, aquellas que son comunes y que aportan un gran valor al proceso en definición, logrando así eliminar todas aquellas que aporten poco valor. Se incorporan además actividades que se consideran que brindan valor al proceso, priorizando aquellas que son definidas en especial para tecnología Microsoft, y contemplando también aquellas que son independiente de la tecnología de desarrollo.

Se estudiaron recomendaciones especiales de calidad sobre procesos de desarrollo en grupos de desarrolladores de Microsoft basados en las experiencias de implementaciones en metodologías ágiles, para obtener una buena calidad en cuanto a las actividades que conforman el proceso.

El estudio de las métricas y normas se enfoca en la mejora de la calidad, a los efectos de incorporar nuevas métricas vinculadas a las distintas actividades del proceso.

Se crea la necesidad de elaborar documentación al final de cada tarea (componente de una etapa), de esta manera se mantiene una gestión del proyecto, con una documentación liviana tal como se recomienda en el manifiesto ágil.

Se considera al MVP como una primera liberación del producto en el mercado. Para ello, este debe ser previamente negociado con el cliente y luego validado ante el mercado y los usuarios finales. Gracias a estos últimos, será posible evaluar el éxito del mismo.

Una vez evaluado el MVP por parte de los usuarios, se realiza una puesta a punto para determinar la continuidad del producto.

3.4 Etapa1 - Inicial

Este proceso no solo se encarga de la elaboración de un producto, sino también de la selección del mismo. Esta primera etapa se encarga de pensar ideas de productos y validarlos, con el fin de seleccionar el producto más conveniente en base a estudios de marketing y análisis de mercados paralelos.

Esta primera etapa consta de tres grandes actividades. En la primera, se formulan ideas. Estas ideas, son en base a los deseos del equipo. Cada pensamiento se considera en un nivel de abstracción, sin considerar tecnologías, metodologías de desarrollo, herramientas, etc.

A partir de las ideas, se seleccionan tres productos candidatos. Para cada una de estos, se elabora una demo donde se baja el nivel de abstracción. De esta manera se analiza viabilidad evitando limitaciones técnicas, o propias del mercado que puedan surgir en un futuro.

Por último se valida el producto. En esta etapa, se validan las ideas con los clientes y además se integran los Stakeholders. El principal objetivo es conocer si de verdad le gusta la propuesta al cliente. Además, se analiza el negocio en base a la futura comercialización.

Es importante destacar que antes de presentar a los productos candidatos a ser validados, es importante identificar que aprendizaje se busca validar una vez salido al mercado el producto a presentar. No olvidemos que la construcción del producto es en conjunto con los usuarios finales y con el cliente. El resultado del producto en el mercado es el que determinará si el cliente continua o no desarrollando el producto.

3.4.1 Formulación de Ideas

La formulación de ideas constituye la primera etapa dentro del modelo de proceso. Este punto de partida es independiente del producto a construir. Puede ocurrir que se cree un producto de cero sin conocer el mismo o se deba de extender uno ya existente para ingresar al mercado de Windows Store. Lo primero a realizarse es un relevamiento y análisis de mercados paralelos así como del mercado objetivo. Para ello se debe analizar:

- ✓ Productos más aceptados.
- ✓ Características de mercados.
- ✓ Perfiles de usuarios.

Todos estos puntos los podemos obtener de diferentes fuentes de información. Podemos acceder directamente a las estadísticas de los mercados, información de estudios realizados sobre los propios mercados o relevar experiencias previas con actores que tuvieron participación en este tipo de medios.

El primer punto, nos indican ideas sobre potenciales productos a desarrollar para introducir en el mercado. Relevando los tipos de aplicaciones más aceptados por los usuarios de la industria que se apunta y de las tiendas de competencia directa, se comienzan a generar ideas de lo que se puede llegar a desarrollar. Una de las recomendaciones es empezar a dividir las aplicaciones por tipos, generando de esta manera una clasificación de las mismas.

Una vez que se releven potenciales productos, se deberá realizar una lista de ideas viables a productos (la misma se puede realizar por medio de un lluvia de ideas o algún mecanismo que se considere acorde) a desarrollar. Estas deben ser en base a los resultados anteriores junto con los gustos individuales de cada uno de los integrantes (tanto el equipo de desarrollo como los Stakeholders). Esta lista será utilizada en pasos posteriores del proceso. Es de vital importancia no desechar ninguna idea en esta etapa, ya que luego cada una tendrá la oportunidad de ser analizada detenidamente.

Para la generación del conjunto de ideas también se recomienda la utilización del enfoque indicado por la metodología SIT. Se realiza un estudio sobre aplicaciones existentes que no tienen un gran acaparamiento del público, y se estudia en que se puede innovar para hacer esa aplicación exitosa, simplemente agregándole nuevas funcionalidades, o combinándola con alguna otra idea que se crea conveniente.

Las investigaciones y estudios no deben centrarse únicamente en aspectos del producto, sino que deberán contemplar aspectos de negocio dentro de la tienda objetivo. Considerar que existen recomendaciones que surgen de la experiencia de los usuarios anteriores en este tipo de mercado, pero que no están difundidas de manera explícita. Sobre todo, es de enorme importancia encontrar personas que estén disponibles a contar sus experiencias en este tipo de interacciones.

Con lo anterior se pude establecer una estrategia de mercado, para realizar publicaciones en las tiendas o poder obtener mayores réditos en la misma. Como así también poder extrapolar a Windows Store ciertas destrezas de mercados paralelos que aún no se estén explotando y se consideren de utilidad.

Al llegar a este instante hemos obtenido información de productos y características de mercados. Como el proceso se enfoca en una tienda de aplicaciones, hay que enfocarse en algo más, en los usuarios. Es necesario poder construir un perfil del usuario de Windows Store. De esta manera se conocerá de manera general los gustos de quienes son el objetivo primario de nuestra comercialización.

Una de las clasificaciones generales que se puede obtener como mínimo observando las estadísticas de la tienda de comercialización es el gusto de comercialización de los usuarios, si prefieren tipo de aplicaciones pagas o gratuitas, de algún estilo en especial o de otro.

Al final de esta etapa se tiene que tener un documento que contenga una lista con los posibles productos a desarrollar. La idea es que la lista sea lo más extensa posible, así en pasos futuros, se puede tener un mejor análisis.

Los análisis de mercado objetivo y mercados paralelos se encuentran en los primeros dos anexos del documento.

3.4.2 Selección de ideas

La elección del producto resulta fundamental para el proceso, y se debe llevar a cabo con mucha cautela. Se deben analizar muy bien cada idea, y compararlas de tal forma de poder seleccionar que producto es el que se quiere construir.

Luego de haber obtenido las distintas ideas entre el equipo de desarrollo y el estudiado de mercados, se debe pasar por una primera instancia de filtrado, en la cual, el equipo de desarrollo se reúne y selecciona una lista con los productos que más tiene ganas de desarrollar. Esta lista debe de ser al menos de 3 productos, y se recomienda tener 10 como para poder tomar la mejor decisión posible.

Esto es a gusto personal del equipo, pero es fundamental poder realizar un producto que se quiera hacer, ya que el equipo se va a encontrar mucho más motivado.

Para estas ideas, se establecieron una serie de criterios, para poder evaluar las ideas entre ellas. Estos criterios son:

- ✓ El tiempo estimado de desarrollo: Si se tiene un tiempo acotado de desarrollo, el tiempo de desarrollo del posible producto puede resultar fundamental para su elección.
- ✓ El público al que apunta el producto: Si se apunta a un público acotado, la posibilidad de realizar un producto exitoso también se acota. Hay que evaluar si esto es un inconveniente a la hora de competir con desarrollar otro producto.
- ✓ Productos existentes: Estudiar que no exista en el mercado un producto similar, y si existe, estudiar cómo se lo puede mejorar para que los clientes se inclinen por este nuevo producto.
- ✓ Experiencia del equipo de desarrollo: Si el equipo de desarrollo cuenta con experiencia en el área de cierta idea seleccionada, es un plus, ya que la curva

de aprendizaje tiende a ser menor que cuando se tiene un total desconocimiento.

- ✓ Ganas de desarrollar el producto: Ponderar de alguna manera las ganas del equipo de desarrollo de querer desarrollar la idea

Para poder seleccionar el producto a desarrollar, lo más conveniente es poder realizar una pequeña demo que baje un nivel de abstracción la idea general que se tiene sobre el posible producto. La demo de los productos no tiene por qué ser un programa funcional, se puede optar por otras alternativas, como ser un video que muestre el funcionamiento de la aplicación, o un conjunto de mockups que muestre la interacción de la aplicación pero con una interface de usuario limitada. La decisión de la demo a desarrollar queda a cargo del equipo, lo importante es que en esta etapa quede como uno de los resultados la demo de cada una de las aplicaciones.

Esta etapa busca bajar un nivel más de abstracción las ideas planteadas en el paso anterior, analizando el potencial desarrollo de las posibles aplicaciones, así evitando comprometerse más adelante con el desarrollo de un producto que no es viable en el mercado, ni viable para el equipo desarrollador.

Para las demos a desarrollar, es deseable un estudio de tecnologías con las que se puede realizar los productos. Dicho estudio consiste en una primera instancia el estudio de tecnologías existentes, así como posibles arquitecturas a incorporar y estándares de desarrollo de aplicaciones. Esto permite tener un mejor conocimiento de las herramientas de desarrollo, poder comparar las distintas tecnologías para identificar las ventajas de utilizarlas, mitigar los distintos riesgos que se pueden encontrar en el desarrollo de aplicaciones con una tecnología desconocida y poder estimar mejor cuanto puede llevar desarrollar una aplicación. Este último aspecto resulta fundamental pero luego tener una mejor estimación de cuánto tiempo lleva desarrollar el producto, con los tiempos del equipo de desarrollo.

También es necesario un estudio de otras herramientas necesarias para llevar a cabo el desarrollo de una aplicación, como ser, herramientas de versionado de código, de documentación, etc. Además es necesario un estudio de la infraestructura necesaria para la aplicación elegida, para luego elegir donde alojar los componentes de arquitectura necesarios y el costo de los mismos.

Hay que estudiar los costos que implican el desarrollo de la aplicación, y como intentar disminuirlos para poder optimizar el desarrollo.

Es conveniente dividir el estudio entre los distintos integrantes del equipo de desarrollo, para poder paralelizar y obtener más temprano los resultados.

Es interesante poner en producción alguna de estas demos, para así poder mitigar los riegos que tiene la puesta en producción de una aplicación, así como

para saber los tiempos que esto conlleva. De esta manera, es posible estimar el tiempo de salida del producto al mercado.

Los resultados de esta sub-etapa son la obtención de una lista de productos a desarrollar, con sus respectivas demos.

3.4.3 Validación del producto

Etapa del proceso en el cual el objetivo principal es validar al menos una de las ideas de la lista de productos formulados en la etapa anterior, basados en las demos desarrolladas. Se proseguirá en reuniones con los clientes del proyecto (o con usuarios finales en caso de no tenerse un cliente) y dar por bueno a productos de la etapa anterior.

El proceso de validación de las ideas será por medio de la formulación y presentación de una demo que muestre los principales aspectos de las ideas de los productos a los clientes, y que estos den por buena la idea presentada. En etapas posteriores se formulará de manera más precisa los requerimientos del producto, en esta etapa es solo para validar la idea general del mismo.

No necesariamente luego de esta etapa se prosiguen con la siguiente, se puede realizar una vuelta atrás, ya que en estas reuniones son una fuente de conocimiento importantísimo para el proyecto. Podríamos encontrarnos que a los distintos Stakeholders (cliente o usuarios finales) no les convenza el producto. Cada expresión que estos tengan frente a la demostración puede significar una idea nueva para el producto que no se había pensado antes, y no forma parte de los supuestos anteriores. Por lo que luego de las validaciones podemos volver a la etapa anterior para poder reformular nuevos productos, con una base de conocimiento fiable y no en base a supuestos como ocurre en un primer momento.

En caso de realizar una vuelta atrás se vuelve a esta etapa con formulaciones de productos más seguros de su aceptación. Lo más natural es que esto ocurra una o dos veces, ya que es muy difícil abstraerse del pensamiento de uno mismo y pensar en los gustos de los demás, por lo que se formulan en una primera instancia productos basados en supuestos relevados en el primer paso del proceso.

Una vez finalizada esta etapa se cuentan con formulaciones de productos validados por los clientes o usuarios finales (dependiendo del caso), dichas formulaciones ya no deben de tener supuestos. De esta forma, como dice lean Startup se obtiene un aprendizaje validado que se respalda en datos empíricos que se obtienen de los consumidores reales, con un feedback tanto cualitativo (como, por ejemplo, si les gusta o no) como cuantitativo (por ejemplo, cuánta gente lo usa y lo encuentra valioso). Como en todo proceso de relevamiento, puede quedar algún punto que se mal interprete es necesario de que a medida que el proyecto avance se mantenga un feedback constante con el cliente para validar cada paso que da el proyecto, y mitigar desarrollos de erróneos.

En estas primeras tres etapas se buscó generación de hipótesis acerca de lo que se quiere por parte del público al cual se apuntó desde un inicio y Stakeholders, y de lo que estarían dispuestos a pagar por ello. Todo esto ya formulado con potenciales productos a desarrollar.

Una vez que se pasa a la siguiente etapa se contará con al menos un producto ya aceptado por parte de los Stakeholder relacionados al producto, por lo que no se volverán a esta etapa ni a las anteriores, y se asumen como validadas las ideas de fondo que se desarrollaran. Como desarrollo ágil se asumen que pueden existir cambios en la formulación del producto pero no se permitirán cambios radicales de enfoque del producto a desarrollar, ya que sería realizar el proceso desde el inicio nuevamente, lo que implica desperdiciar el trabajo ya realizado al ser escaso el aprendizaje validado obtenido.

3.5 Etapa 2 - Elaboración

En la etapa de elaboración las disciplinas que cumplen un papel fundamental son: planificación, análisis, diseño y calidad. El resultado de esta etapa son los documentos de especificaciones del producto y del proceso. En relación al producto se contará con una especificación de requisitos y casos de uso del o los productos en forma de alto nivel, además del conjunto de medidas de calidad definidas para los productos. En cuanto al proceso se establecerán cronogramas de ejecuciones, medidas de calidad del proceso y responsables e involucrados en la ejecución de actividades del proceso.

Al finalizar la etapa se contará con el o los releases que se establezcan para las salidas al mercado con los MVP ya formulados.

Las principales actividades son definición y validación del artefacto: "Story map" (en el cual se especifican los requerimientos funcionales y casos de uso con un alto nivel de abstracción), y la actividad: "Sprint 0", donde se realiza el análisis, diseño y planificación del proceso de desarrollo.

3.5.1 Story Map

El Story Map es una técnica utilizada en metodologías ágiles que permite realizar un enfoque visual del producto en forma de historias en lugar de caso de usos. Es una práctica, en la cual las historias de usuario se organizan de tal forma que ayudan a pensar al producto desde los procesos de negocio y las necesidades de los usuarios. En este modelo es más simple asegurar que el Product Backlog está completo (en el sentido de que soporta todas las actividades de los usuarios del sistema) y se visualizan más claramente las funcionalidades alternativas y la relación entre las historias de alto nivel y los detalles.

Los Story Maps son útiles a la hora de definir un Product Backlog inicial, pero también resultan una herramienta interesante para la creación de un plan de

Releases que aproveche al máximo el aspecto iterativo y evolutivo del desarrollo ágil. Por esto es que se recomienda su utilización para la definición de los MVP del proyecto.

Al ser Scrum la metodología base de construcción del producto como se verá más adelante, es necesario definir las porciones de producto que integraran los distintos Sprints de desarrollo. Entonces, con el objetivo armar el Product Backlog de Scrum pero con una visión del producto, es que se definió esta etapa, que consiste en un artefacto, en donde se arma un flujo de actividades de un usuario o grupo de usuarios, y se describe el flujo funcional de actividades que realiza un usuario a muy alto nivel. La idea es pensar en todas las posibles actividades que pueda realizar un usuario, y armar un árbol de actividades, indicando precedencia entre las mismas.

Una vez obtenido el grafo de actividades de los usuarios, se baja un nivel de abstracción para cada una de las actividades, y se definen las distintas historias que componen la actividad. Y luego para cada una de las historias se puede definir las tareas asociadas a las historias.

Al final de todas estas definiciones, se tiene un Product Backlog armado pero con una estructura funcional, a diferencia de lo que se hace normalmente en Scrum, donde se tiene el Product Backlog pero sin relación ninguna entre las historias. Y en este momento se tienen las historias que son las que tendrían que integrar el MVP, y se ordenan esas historias por prioridad.

El objetivo de la etapa es establecer con historias completas todos los flujos de vida de nuestro producto. Se deberán de incluir todas las historias del mismo (por más simples que estas parezcan). Como se ha observado en ninguno otro momento del proceso hasta este punto se han descrito casos de usos formales, de aquí la vital importancia de especificar todas las historias del mismo por más simples que estas parezcan. Una forma de ver esto, es como ver la especificación de los casos de uso del producto en una forma de más alto nivel por medio de historia.

Una vez establecidas todas las historias del sistema, y validadas con los Stakeholders, es momento de proceder a definir el MVP del producto a elaborar. MVP ya descripto en secciones anteriores donde se establece nuestro mínimo producto viable (para la realización de un Release en la Windows Store), basados en las historias establecidas.

La elaboración del MVP se realiza en colaboración directa de los Stakeholders relacionados al producto. Se busca establecer que historias de las definidas en el Story Map son incluidas en el mismo. En las etapas posteriores se planeará en base a estas historias que constituyen el MVP.

3.5.2 Sprint 0

Es la etapa previa al comienzo del desarrollo del producto utilizando la metodología Scrum. Hasta esta etapa realizaremos una introducción en la metodología Scrum desde la preparación y planificación del proyecto. Los objetivos de esta etapa son la de preparar el proceso desde una perspectiva:

- ✓ tecnológica.
- ✓ metodológica.
- ✓ organizativa.

Como primer punto, el equipo de desarrollo tiene que definir cuánto tiempo va a durar cada Sprint. Este proceso recomienda Sprints de dos semanas, en base al estudio de Scrum realizado y la charla con expertos desarrollando en base a esta metodología (ver anexos). Luego es necesario que se estimen las historias del Product Backlog producido con el artefacto: Story Map. Para la estimación, se recomienda basarse en los tiempos de desarrollo de las distintas demos definidas en las fases anteriores. Se plantea utilizar algún artefacto de planificación, como ser Planning Póker en la estimación de 1/3 de las historias definidas, como una medida alternativa de estimación al juicio de experto que se realiza en las demás historias.

Luego de estimar las historias, y la duración de un Sprint, queda definido en principio que historias se desarrollan en cada Sprint. La idea es realizar un cronograma de desarrollo, teniendo en cuenta estos tiempos, para tener noción de cómo va el desarrollo del producto, si es que se atrasa en algún momento, etc.

Luego, en la finalización de cada Sprint, se va a mejorar la velocidad de desarrollo en base a las historias hechas en ese Sprint, y volviendo a estimar las historias que restan, en cada Sprint queda definido las historias que se desarrollan. Esto automáticamente puede calcular el tiempo de desarrollo que tendría que llevar para tener el producto finalizado. Y se puede ver cómo se va en el desarrollo, en base al cronograma inicial.

También en esta parte, es donde el equipo de desarrollo defina aspectos del producto necesarios antes de comenzar con su desarrollo, estos son:

- ✓ la definición de la arquitectura a utilizar
- ✓ el diseño de la base de datos si es que se tiene una
- ✓ que infraestructura es la que se necesita y cual se va usar
- ✓ el tipo de testing que se va a realizar
- ✓ las definición de que métricas son las que se van a tomar
- ✓ el diseño de ui que tendría que tener el producto
- ✓ los wiframe y ux (diseño de interacción de pantalla y flujo de información)
- ✓ el manejo de seguridad que se quiere tener en la aplicación,
- ✓ como va a ser la Gestión de configuración de software.

Las definiciones de estos aspectos están fuertemente relacionadas al producto a realizar, el proceso solo indica que este es el momento de realizar toda la definición previa necesaria al comienzo del desarrollo del producto.

3.5.3 Criterios de aceptación de una historia de desarrollo

Siguiendo el criterio de las metodologías agiles, las historias se aceptan utilizan el "Todo o nada" para establecer si las mismas están terminadas, o no. No existe un término medio. La historia se considera como terminada, cuando no solo se finaliza su implementación, sino que pasa todos los criterios de calidad establecidos por el proceso. Siguiendo el estudio de la ejecución de Scrum en distintos proyectos de Microsoft, se estableció que los criterios son:

- ✓ La compilación no debe tener errores o Warnings.
- ✓ Realizar los casos de prueba de las historias basados en la técnica BDD.
- ✓ La historia la debe dar por válida otro integrante del equipo que no sea el que la desarrollo
- ✓ Se ejecuten todos los casos de prueba de las historias de los anteriores Sprints sin que estos fallen.

Se recomienda que se sigan todos los puntos anteriores. Estos se pueden complementar con lo siguiente:

- ✓ Todos los test unitarios se deben ejecutar correctamente
- ✓ Los test unitarios deben cubrir al menos el 80% del código.
- ✓ Testing de integración integrado al repositorio del código.

El testing unitario que se recomienda realizar en caso de desarrollar un sistema en capas, es sobre las funcionalidades que expone cada capa. A su vez se recomienda contener los test unitarios integrados al repositorio del código para de esta manera realizar un testing de integración en cada actualización al código del repositorio.

En caso de realizarse un sistema de múltiples usuarios es recomendable realizar testing de performance y de carga una vez finalizada la construcción del producto, para de esta manera conocer el comportamiento del mismo antes situaciones de stress en el sistema.

3.5.4 Métricas

A la hora de determinar cuáles métricas debemos utilizar y que datos debemos recoger, nos basamos en una combinación de ideas de Scrum y XP. A partir de estas metodologías, clasificamos las métricas en tres conjuntos: al comienzo de cada Sprint, durante el Sprint y al final de mismo. Cada una de ellas, nos servirá para medir el desempeño del proyecto con lo que respecta a avance, valor agregado en el producto, satisfacción del cliente, atrasos, etc.

Como mencionamos anteriormente, en cada reunión de planificación de Sprint, seleccionamos las primeras historias del Product Backlog y re estimamos cada una en Story Points (unidad de medida que utilizaremos de ahora en más). Decidimos cuales y cuantas historias incluir en base a las siguientes mediciones:

- ✓ Velocidad objetivo: Cantidad de Story Points que se desean incluir en el siguiente Sprint. Se calcula sumando los Story Points de todas las historias incluidas en el nuevo Sprint, obtenidas del Product Backlog.
- ✓ Horas estimadas: Cantidad de horas que corresponde a las historias seleccionadas para el Sprint. Este es el primer valor que dibujo en el gráfico de Burn-Down correspondiente al registro de horas.
- ✓ Horas disponibles: Total de horas hombre disponibles para el Sprint. No necesariamente dichas horas se corresponden exclusivamente a horas de programación.
- ✓ Rendimiento objetivo: (Horas Estimadas / Horas disponibles) en porcentaje.

En cada reunión diaria, es posible saber si se llega al final del Sprint con las tareas planificadas terminadas, si faltarán tareas y si sobraran las mismas. Para ello existe el grafico: Burn-Down donde podemos expresar lo que falta en horas o Story Points.

Al final de cada Sprint se obtienen las siguientes estadísticas finales:

- ✓ Velocidad real: Cantidad de Story Points realmente completados al final del Sprint. El comparativo de esta métrica con la velocidad objetivo (considerada al comienzo de cada Sprint) nos permite concluir a cerca de las estimaciones.
- ✓ Tareas previstas en Sprint completadas: Cantidad de puntos, horas y cantidad de tareas terminadas en el Sprint con lo que respecta a tareas previstas. Se excluyen las tareas, horas y puntos no planificados en el Sprint que si se realizaron.
- ✓ Horas reales: Cantidad de horas dedicadas al Sprint. En esta medida se suman las horas extras a las estimadas.
- ✓ Rendimiento Sprint: (Horas previstas en Sprint completadas/ Horas reales) en porcentaje.
- ✓ Horas registradas a tareas planeadas: Recuento de todas las horas reportadas para las historias planeadas al comienzo del Sprint.
- ✓ Horas registradas a tareas no planificadas (dentro del Sprint): Cantidad de horas correspondientes a las tareas no planificadas al comienzo de cada Sprint pero que formaron parte del Sprint.
- ✓ Error de estimación:

Horas registradas a tareas planeadas + Horas registradas tareas no planificadas (dentro del Sprint))/ (Tareas previstas en Sprint completadas - 1)

- ✓ Rendimiento real:

(Horas registradas a tareas planeadas + Horas registradas a tareas no planificadas (dentro del Sprint))/ (Horas reales)

- ✓ Horas registradas a tareas no planificadas (fuera del Sprint): Horas destinadas al proyecto que no fueron asignadas a tareas dentro del Sprint, puede ser publicidad, arreglos es ambientes, configuración, etc. Tareas que no tienen que ver con el producto.
- ✓ Total de horas registradas: Horas registradas a tareas planeadas + Horas registradas a tareas no planificadas (dentro y fuera del Sprint)
- ✓ Error registrado: 1-(Total de horas registradas/Horas reales)

Existen además, un conjunto de métricas que permiten demostrar el progreso en el proyecto. Dentro de estas, la más importante es el valor que se está dando al cliente. Mediante esta, el cliente puede conocer la velocidad con que retorna su inversión y saber cuándo ya no es necesario seguir con el proyecto, porque los beneficios pendientes de obtener ya no compensan sus costes.

Sin embargo, cuando se mide a una persona o a un equipo de una determinada manera, sus acciones pueden desviarse en exceso hacia ese objetivo y descuidar otros aspectos también importantes como, por ejemplo, la calidad, los costes, los riesgos, la sostenibilidad de la velocidad con que obtienen objetivos, etc. Por ello, puede ser necesario utilizar un conjunto de métricas de diferentes aspectos relacionados, creando de esta manera un cuadro de mandos integral ágil (agile balanced scorecard).

Para el cuadro de mandos se debe escoger el mínimo número de métricas que permita tomar decisiones sobre los resultados del proyecto y las necesidades del cliente. Se debe minimizar tanto el coste e impacto que se añade al trabajo ordinario del equipo (hay que ir registrando lo que sucede, cosa que puede incluso desvirtuar la métrica) como su coste de recolección, proceso y presentación. Por todo esto, cuando un problema específico esté resuelto, hay que plantear si tiene sentido seguir recogiendo las métricas asociadas.

Es conveniente mostrar en el cuadro de mandos los objetivos de la iteración, entrega, proyecto, programa u organización, según sea el ámbito del cuadro de mandos (mostrar la velocidad con que se consiguen los objetivos del proyecto o guiar e informar sobre la estrategia de la organización y su cumplimiento).

En los equipos ágiles es común que las métricas surjan ante una necesidad del equipo, como una forma de mejorarse. Por ejemplo la velocidad del equipo puede ser una herramienta para que el equipo planifique y tome compromisos, e incluso para cuantificar una mejora.

Será fundamental presentar las tendencias de las métricas a lo largo del tiempo (los valores en un momento dado pueden deberse a situaciones puntuales),

con diferentes niveles de agregación en función de las necesidades: diario, iteración, etc.

Las principales marticas que permiten ver el avance del proyecto son:

- ✓ Cantidad de historias o puntos aprobadas en el Sprint.
- ✓ Cantidad de historias o puntos nuevos por Sprint.
- ✓ Cantidad de historias no aprobadas en el Sprint.
- ✓ Cantidad de puntos correspondientes a historias de errores reportados por Sprint.
- ✓ Cantidad de puntos correspondientes a historias de errores resueltas por Sprint.
- ✓ Tiempo medio de implementación de una historia.
- ✓ Story Points acumulados en el proyecto.
- ✓ Cambios en historias por Sprint.
- ✓ Días de trabajo ideales pendientes.

3.6 Etapa 3 – Construcción

En esta etapa, es donde comienza el desarrollo del producto, hasta alcanzar el MVP que indica que se finalizó la construcción. Hemos elegido la metodología scrum como base de desarrollo, mezclando aspectos de otras metodologías agiles, mencionadas previamente.

Se deberá realizar tantas iteraciones como sea posible hasta llegar al MVP del producto que se ha definido. Al finalizar cada etapa del sprint se debe efectuar una evaluación del desarrollo del proyecto con especial enfoque en el sprint que acaba de terminar.

Cada sprint finalizará en el tiempo que se establezca en el sprint 0, cumpliéndose o no todas las tareas que se han asignado a dicho sprint. De igual manera que si se llegase a finalizar todas las tareas del sprint, se deberá de finalizar el mismo cuando se cumpla el tiempo establecido, lo que se deberá de hacer es de tomar tareas que no estén asignadas en el Product backlog y seguir realizándolas.

Una vez que finalice el sprint, se realizarán evaluaciones del trabajo realizado en dicho sprint, se efectuarán conclusiones sobre el proceso, y se tendrá una reunión de planificación del próximo sprint, en la cual se seleccionarán que historias son las que se realizarán en el próximo sprint. Estas historias ya estimadas con anterioridad pueden ser estimadas nuevamente (para estas estimaciones es posible utilizar algún método de estimaciones, pero se recomienda la experiencia adquirida de los sprints anteriores), esto es debido a que al encontrarse en etapas de desarrollo, se tiene más conocimiento real de cuanto pueden llegar a costar las historias que se consideran. Una vez iniciado el sprint no es posible re-estimar las mismas. En estas reuniones deben de participar el equipo de desarrollo, cómo así cualquier stakeholder que lo desee.

Asimismo se deberá de establecer un calendario de reuniones en cada sprints. Scrum recomiendas reuniones diarias, aquí se deja como parte de la personalización del proceso por el equipo de desarrollo. Dichas reuniones no pueden ser mayores a 3 o 4 días de diferencia con la anterior y una duración no más de 15 a 30 minutos. En las mismas se deberá discutir por parte de cada integrante que se encuentre desarrollando:

- ¿Qué has hecho?
- ¿Qué vas hacer?
- ¿Necesitas ayuda con algo en específico?

3.7 Etapa 4 - Liberación

En esta etapa se procede a publicar el producto en la Tienda de Windows Store. Además, se realizaran mantenimiento evolutivo por errores que puedan ocurrir en la misma.

No solo se procede con la liberación y comercialización sino que también se comienza con actividades de recepciones de feedback desde los usuarios de la tienda. Los mismos son procesados como oportunidades de correcciones y mejoras de los productos del proyecto o como fuente de información para posibles nuevos proyectos.

Luego de un considerable tiempo de puesta en producción y recibimiento de feedback por parte de los usuarios, se realizan reuniones con los involucrados en el proyecto (equipo de desarrollo y Stakeholders) para tomar la decisión de continuar el desarrollo con nuevos MVP o dejar el proyecto en este punto.

Por lo que se definirá en esta etapa dos actividades a realizarse con sus respectivas documentaciones. Primero la liberación (con el impacto de la misma en la tienda, y mantenimiento evolutivo en lo que sea posible) y luego las conclusiones (en el cual se realizan análisis sobre el proceso y productos).

3.7.1 Liberación

La liberación consistes en el proceso de colocar el producto en la tienda de aplicaciones de Windows. Este proceso conlleva un conjunto de pruebas por parte de la tienda, por lo que no se publica de manera inmediata, y es necesario superar todas las pruebas sin errores para su publicación. Esto es usado por parte de Microsoft como una herramienta de calidad en las aplicaciones disponibles en la tienda.

Para tener un alto grado de aceptación del producto y no degradar la calidad de la tienda, Microsoft deja un conjunto de pruebas a ser ejecutadas de forma automática por los desarrolladores. Las mismas buscan emular los criterios de aceptación de una aplicación.

Una vez colocada la aplicación en la tienda de Windows, se espera recibir comentarios por parte de los usuarios. Todos estos comentarios son registrados para ser utilizados como datos objetivos a la hora de evaluar el producto.

Luego de ser publicada la aplicación comienza una fase de mantenimiento evolutivo en los aspectos que se puedan solucionar en caso de errores. En caso de errores que no se pueden solucionar por tratarse de un error en el producto mismo instalado es recomendable corregir el mismo en fase de desarrollo y subir una nueva versión del producto corrigiendo el mismo lo antes posible (para no degradar la imagen de la aplicación).

En caso de recibirse notificaciones de errores que no generen una degradación significativa del producto es posible solucionar los mismos con los nuevos releases que se pienses colocar del producto en el futuro.

3.7.2 Conclusiones

Una vez que se ha alcanzado el MVP por medio de las sucesivas iteraciones de los Sprints, se llega a una etapa en la cual es recomendable realizar conclusiones sobre el proceso que se esté llevando a cabo, principalmente tratando de medir los distintos conocimientos adquiridos en el transcurso del mismo. Esta etapa deberá de realizar especial énfasis en el feedback con los Stakeholders del proyecto, y si es posible con usuarios finales del producto que se realizó, de esta forma se analizará el alcance del producto base que se obtuvo como resultado del proceso, y los diferentes puntos de vista de cada Stakeholder e integrante del equipo de desarrollo sobre el proceso cuestión.

Una de las principal tarea de esta etapa radica en la conclusión sobre el proceso y el producto obtenido, realizar una reflexión analizando si es lo que se esperaba desde un principio, si el proceso nos brindó las herramientas necesarios para el desarrollo de aplicación, si la aplicación es lo que se pensó desde un principio, sí impactó positivamente en el cliente y usuarios finales. Luego de esto se deberá de tomar una decisión de seguir adelante con las demás funcionalidades del producto generando nuevas versión de un MVP (con la historia que se quedaron afuera del primer MVP, por lo cual se regresa a la etapa de Story Map), o si se deberá de reformular el mismo o cancelar.

En caso de seguir con el producto es necesario volver a empezar el proceso, y así poder obtener una nueva versión de nuestro producto. En este caso se habilita la posibilidad de poder generar nuevas historias y de eliminar las que se consideren no necesarias del conjunto de historias que no integraron el anterior MVP.

Cambiando el foco en cuanto al proceso, es un momento de reflexión sobre el mismo, si se considera cambiar, continuar o mitigar ciertos aspectos. La mayoría de las decisiones se toman en base a las mediciones realizadas a lo largo del proceso y al final del mismo. Es importante ser objetivos en el mismo, no dejarse sesgar por el orgullo personal.

Las conclusiones son la base de la evolución y estabilización del proceso, es por esto que constituye una de las partes más fundamentales del mismo, se deberán de tomar un tiempo presiones para realizar las mismas.

Todas las conclusiones a las que se llegué es de gran importancia mantener un registro de las mismas, ya que son una fuente de conocimiento único (por no decir el más apreciado en la industria) de experiencia, con la cual nos permitirá aplicar de mejor manera el proceso en futuros desarrollos o servir como base a nuevos desarrollos relacionados.

Todo lo que se concluya en esta actividad es necesario dejarlo asentado por escrito como análisis post mortem del proceso y producto.

3.8 Roles

Los roles son una de las partes fundamentales de una definición de un proceso de software. Son a los roles a los cuales se le definen funcionalidades en el proceso y no a los integrantes del equipo. A estos simplemente se le asignan responsabilidades de ejecutar los roles que le son asignados.

Esta asignación de roles debe de ser asignada por capacidad de los miembros del equipo de desarrollo, para obtener una mejor performance en las ejecuciones de los mismos por parte de los miembros del equipo. La asignación se recomienda que se realice desde un inicio, y no se deberá de modificar las funcionalidades de cada rol, en caso de ser necesario se modificaran a los ejecutores de los roles en el proceso (aunque no es muy recomendado, ya que degrada la performance del proceso).

Los roles que en este proceso se definen se pueden dividir en dos clases. La primera clase se encuentra vinculada con los que son referentes a aspectos del proceso en exclusiva, la segunda son aquellos que se involucran en el desarrollo del producto que se construya.

Debido a las actividades definidas en cada etapa, se decidió definir roles acordes a esas actividades por etapa.

En la etapa inicial se definieron los siguientes roles:

- ✓ **Coordinador**: Es el encargado de establecer las fechas y la forma de comunicación, ente los integrantes del equipo, con los Stakeholders. También

encargado de coordinar entrevistas con expertos en las diversas áreas que se consideren necesarias.

- ✓ **Analista mercado**: Es el encargado de relevar los aspectos más importantes del mercado al cual se quiere crear el producto, para luego poder identificar qué tipos de productos se pueden construir, y que características tienen que tener.
- ✓ **Analista técnico:** Es el encargado de investigar las diferentes tecnologías con las que se puede construir el producto, para definir cuáles son las que se van a utilizar. También debe mitigar los riesgos que tiene utilizar dichas tecnologías.

En la etapa de elaboración se definieron los siguientes roles:

- ✓ **Analista Funcional:** Es el encargado de relevar los requerimientos del producto, y definir las historias del story map
- ✓ **Arquitecto diseñador:** Es el encargado de definir la arquitectura de la aplicación, también de diseñar la base de datos, y la seguridad que tiene que tener el sistema
- ✓ **Scrum Master:** Es el encargado en armar el plan de releases, indicando cuales historias del story map irían en cada Sprint.
- ✓ **QA Testing:** Encargado de realizar el plan de calidad en base a atributos de calidad identificados en el producto. Encargado de realizar el plan de testing en base a las actividades que crea conveniente.
- ✓ **Sysadmin:** Responsable de definir qué infraestructura es la que se va a utilizar con la arquitectura definida, y se encarga en las demás etapas del mantenimiento de la misma.
- ✓ **Diseñador de UI:** Es el que se encarga de diseñar el diseño gráfico de la aplicación y también las interacciones entre las distintas pantallas.

En la etapa de Implementación, al basarse la elaboración del producto bajo la metodología de Scrum, se definió que los roles de Scrum (Programador, Scrum Master y Product Owner) también estén en nuestro proceso. Además se agregó el rol: QA Testing.

Finalmente en la etapa de liberación se definieron los siguientes roles:

- ✓ **Encargado de Deploy**: Es el responsable de poner en producción la aplicación en el mercado.
- ✓ **Soporte**: Es el encargado en dar soporte y soluciones a los problemas que surgen en la aplicación, luego de que este en

3.9 Documentaciones de las actividades de las etapas

Como se mencionó en la definición del proceso, este se divide en etapas, las misma son compuestas por un conjunto de actividades/tareas a llevarse a cabo, para cada una de ellas se deberá de generar un conjunto de artefactos con el fin de

mantener un registro de las actividades y analizar desde puntos objetivos el proceso.

Se procederá a enumerar etapas y actividades con sus respectivas documentaciones a realizar. Aclarar que no se definirán un conjunto de templetes para los documentos a desarrollar, pero si lo que se deberá dejar por escrito en cada uno de ellos. En todos estos documentos se deberá de mantener un control de cambios y aprobaciones desde el propio documento, de esta forma se facilita el versionado.

- ✓ Etapa 1:
  - Documento de especificación del producto: en este documento se deberá de especificar las tres actividades realizadas en esta etapa. Se recomienda que para cada una de ellas se realice un capítulo (uno para las ideas generadas, otro para los productos especificados y otro para la validación del o los productos aceptados).
- ✓ Etapa 2:
  - Documento de Story Map: se deberá de especificar las historias que componen el Story Map del o los productos a construir, a su vez es se deberá de establecer un orden entre las historias a desarrollar. Como parte final de la documentación se establecerá un plan de release de productos (basados en las historias que componen dicho release) al mercado.
  - Documento de Métricas: el objetivo del mismo es de establecer las métricas del proceso y del producto que se está construyendo.
  - Documento de Arquitectura: deberá de contener información acerca de la arquitectura del producto a desarrollar, como así también capítulos donde se especifique el diseño de base de datos (en caso de que se utilice), y otro capítulo donde contenga información acerca de la infraestructura que se utilizara para el deploy del producto.
  - Documento de Testing: documento donde se especifica el plan de testing a seguir durante la implementación del producto a desarrollar como así de testing posteriores al desarrollo del mismo.
- ✓ Etapa 3:
  - Documento de Métricas de desarrollo: a diferencia del documento de métricas anterior este especifica los números obtenidos durante cada Sprint de desarrollo de las medidas correspondiente al desarrollo del producto establecidas anteriormente.
  - Documento de Lista de Errores: se utilizará para mantener un registro de los errores encontrados en el desarrollo del producto como así de sus correcciones realizadas.
- ✓ Etapa 4:
  - Documento de "Conclusiones": análisis de la ejecución del proceso para el proyecto realizado con conclusiones sobre el mismo.

- Globales: documentación que no se corresponde con solo una etapa, en este caso documento a realizarse y actualizarse durante toda la ejecución del proceso:
- Documento de Horas registradas: documento que contiene las horas realizadas por cada integrante del equipo de desarrollo, estas horas corresponden a toda la ejecución del proyecto.

4. Ejecución del proceso

En este capítulo del documento nos concentraremos en la ejecución del proceso definido en el capítulo anterior. Se realizará un recorrido por cada una de las etapas definidas, para las cuales se menciona su ejecución así como los resultados obtenidos en cada una de las actividades definidas. En base a los resultados obtenidos se concluirá sobre el proceso propuesto.

Como se menciona en la definición, cada una de las etapas del proceso tiene determinado su instante de ejecución, entradas y salidas. Por ello no se admiten paralelismos entre etapas.

El proceso fue puesto a prueba con Marcelo Guerra en un rol de Stakeholder involucrado en el producto (versión cliente), y el tutor del proyecto Jorge Triñanes como Stakeholder supervisando el desarrollo del proceso.

La puesta en funcionamiento del proceso comenzó por mediados del mes de mayo de 2013, con una finalización del proceso en diciembre de 2013 (ver anexo con detalle). Este tiempo incluye una segunda ejecución del proceso una vez finalizado el mismo. De esta manera se cumplía el requisito de realizar un segundo release en el mercado de aplicaciones de Windows Store.

Antes de la ejecución de las etapas del proceso, se realizó la asignación primaria de roles del proceso. Le denominamos primaria debido a que no es definitiva, por cualquier razón bien fundamentada se puede dar un cambio de usuario ejecutor del rol.

La asignación de los roles de desarrollo se realizó en el Sprint 0, por lo que se mencionara la asignación cuando se llegue a dicha actividad.

En cuanto al proceso se mencionara cual fueron las designaciones:

- ✓ **Coordinador:** Para esta Etapa, asignamos a Rodrigo Machín con el rol de coordinador.
- ✓ **Analista de Mercado:** Los tres integrantes del grupo tuvieron este rol, e investigaron sobre el mercado y las definiciones de las posibles ideas de productos.
- ✓ **Analista Técnico:** Alexis Gonzatto fue el encargado de investigar las distintas tecnologías de desarrollo para aplicaciones de Windows 8. Y mitigar riesgos.
- ✓ **Cliente:** A su vez, los 3 integrantes, junto a Marcelo Guerra cumplieron con el rol de cliente para evaluar las ideas propuestas.

4.1 Etapa 1 - Ideas

Como se mencionó en la definición, en esta etapa el objetivo principal es llegar a la etapa de elaboración, con uno o varios productos ya validados por parte

de todos los involucrados en el proyecto enfocados al producto. Su ejecución se realizó tal cual se definió de anteriormente por medio de la división de la etapas en actividades/tareas.

La ejecución de la misma consumió un total de 30 días (durando todo el primer mes de ejecución del proceso).

Recordemos las actividades/tareas que se llevan a cabo en dicha etapa:

- ✓ Generar ideas.
- ✓ Formulación de productos.
- ✓ Validación de productos.

Su ejecución se realizó en el orden que se estableció, debido a que era necesario ya que existen dependencias de las actividades entre ellas (a excepción de la actividad de "Generar Ideas"), lo que no permite su paralelismo.

4.1.1 Generar ideas.

La generación de ideas fue la primera actividad realizada en el proyecto. Los encargados de la misma fueron el equipo desarrollador del proyecto en conjunto con Marcelo Guerra como un Stakeholder involucrado en el producto.

Se llevó a cabo por medio de varias reuniones del equipo y del equipo con Marcelo. En dichas reuniones se buscaba en cada una generar una lluvia de ideas de potenciales productos a desarrollar para ser introducidos en el mercado de Windows Store. Como se menciona en la definición de esta actividad no se descartó ninguna de las ideas que se generó, ya que estas son analizadas en actividades posteriores de la misma etapa y se decide su futuro.

Desde la primer reunión que se mantuvo con Marcelo, se estableció que en primer lugar antes de empezar a generar ideas, se realizara un estudio de aplicaciones de Windows Store, mercados paralelos y aplicaciones populares en distintas redes sociales, para que de esta manera se establezca criterio de aplicaciones que pueden ser aceptadas de manera más simple por los usuarios, y cumplir más rápido los objetivos enfocados del proyecto al producto. Esto debido a que se apunta a un mercado en donde el cliente son los usuarios ordinarios (los cuales no tiene por qué compartir funciones en común) por lo que no se apunta a una persona en específica con necesidades conocidas. Este estudio se realizó con el fin de poder agrupar a grandes rasgos que es lo que pretenden los usuarios de las aplicaciones y que hacen que estas triunfen o fracasen en las tiendas.

Una vez realizado el estudio de productos de los mercados y redes sociales, se cuenta con un conocimiento de las aplicaciones más aceptadas en la industria. El estudio realizado se encuentra en los anexos 1 y 2. En dichos anexos se construye el perfil de usuarios de las tiendas de aplicaciones generales como así también de

usuarios específicos de Windows 8. De esta manera se empieza a conocer qué tipo de aplicaciones tienen un grado más alto de ser aceptadas.

El perfil generado de los usuarios de la tienda indica que estos prefieren el siguiente tipo de aplicaciones:

- ✓ Juegos.
- ✓ Aplicaciones que involucren redes sociales.
- ✓ Aplicaciones de comunicaciones.
- ✓ Aplicaciones de competencias.
- ✓ Aplicaciones de noticias instantáneas.
- ✓ Aplicaciones multimedia.
- ✓ Aplicaciones de almacenamiento.
- ✓ Aplicaciones de Ofimática.

No solo de la investigación de los mercados se obtuvo información acerca de las aplicaciones, sino que se realizó una entrevista con un experto en esta área como es Álvaro Azofra (anexo 3.), el cual nos contó sus experiencias en la industria (sus éxitos y fracasos), además de mencionar los gustos de los usuarios sobre las aplicaciones que se pretenden destacar, "tips" de ayuda al momento de poder diseñar una aplicación a ser comercializada en mercados similares al de Windows Store. Son consejos que se pueden extrapolar a cualquier mercado de aplicaciones y no sesgarlos a solamente al mercado de IPhone que es al que Álvaro trabaja.

Basado en estos perfiles de productos más aceptados por los usuarios, en la entrevista con expertos y en los gustos y ganas de desarrollar por parte de los integrantes del equipo, se decide comenzar a generar ideas basados en los mismos. Aun así se decidió no descartar ningún producto que se pueda generar y que no se incluya en ninguno de los perfiles generados del estudio.

Luego de sucesivas reuniones entre el equipo de desarrollo y el Stakeholder involucrado en el aspecto de producto, se decidió establecer la generación del producto o productos basados en las siguientes ideas:

- ✓ Juegos de uso individuales.
- ✓ Juegos de competencia entre usuarios.
- ✓ Aplicaciones de redes sociales.
- ✓ Aplicaciones de comunicaciones.
- ✓ Aplicaciones de noticias.

Una vez que se ha establecido una base de ideas de potenciales productos a generar entre el equipo de desarrollo y el Stakeholder, se procede a generar el documento correspondiente a dicha actividad en donde se dejan asentadas las ideas que se utilizaran como base en la actividad de "Formulación de productos".

4.1.2 Formulación de productos

Como se menciona en la definición del proceso, esta etapa es desarrollada de forma exclusiva por el equipo de desarrollo (de esta manera se empieza a visualizar mejor el alcance de los productos y de las posibles tecnologías a utilizar). El objetivo de la misma es de formular el o los productos a construir en base a ideas anteriores.

Al utilizar la base de ideas y poniendo en combinación con metodologías SIT, se genera en primer lugar la formulación de un conjunto de productos a desarrollar.

Se planteó el desarrollo de:

- ✓ Juego de competencia entre amigos y usuarios generales: Primero que nada se ha observado que los juegos son uno de los estilos de aplicaciones más descargadas y ejecutadas en este mercado. Además estas son más adictivos en medida que se otorgue un ambiente de competencia entre usuarios. Utilizando la metodología SIT para innovar, se utiliza como base una aplicación popular de la red social Facebook, como es "Triviador mundo", en combinación con los aspectos antes mencionados. Se formula un juego de trivia de competencia entre usuarios amigos y de los que se loguen en la aplicación, aprovechando en este punto los aspectos de juego, competencia, y redes sociales. Como leitmotiv de la aplicación se decidió incursionar con un juego conocido por gran mayoría de los usuarios (de esta manera no les parecería extraño su dinámica), por lo que se planteó el uso del tradicional juego de "Batalla naval". Este producto entonces se formuló como un juego basado en el hilo de la "Batalla naval", con el cual se puede ingresar a una comunidad que se conecte a las redes sociales y así poder competir contra amigos y demás usuarios de la comunidad. El mismo contendrá funcionalidades de redes sociales (como ocurren en las mayorías de las aplicaciones de red social), y fomentará dentro de la propia aplicación la competencia entre los usuarios.
- ✓ Jugo de cartas: Un juego de cartas popular en Estados Unidos (región donde apunta mayormente Windows Store) el cual no se encuentra en ninguna versión digital del mismo. Idea que proviene de Marcelo como una observación del ambiente. El juego se llama Mao, consiste en un juego simple de naipes recomendado para 4 o más jugadores en el que pierde el jugador que consigue una determinada cantidad de puntos. El juego casi siempre se juega con baraja francesa, pero se puede probar con española. En algunos casos y dependiendo de la cantidad de jugadores se puede jugar con dos mazos.
- ✓ Aplicación orientada a funciones económicas: Aplicación orientada a ayudar a los usuarios en sus respectivas economías diarias de manera individual. La aplicación se encargaría de tener un período de conocimiento de los gastos del usuario y luego del mismo a proyectar la economía del mismo a pedido del usuario en un período de tiempo que el usuario demande.

- ✓ Aplicación de lector de noticias: Una aplicación en el cual el usuarios se generé su propio perfil de aplicación, ingrese un conjunto de RSS de noticias para mantenerse informado, luego al recibir una noticia por medio de la aplicación por el uso de algún RSS ingresado, este tendría la posibilidad de poder calificarla, o de compartir por medio de alguna red social con sus propios comentarios. El sistema de clasificación de las noticias tendría que aprender con el transcurso del tiempo los tipos de noticias por los cuales se interesa el usuario para que de esta manera realice un filtro de las noticias antes de notificarle a los usuarios de las mismas.

Luego de haber formulado los productos mencionados arriba, se procede con la elaboración de las demos correspondientes a cada uno de ellos. Como se sabe dichas demos no tiene por qué ser necesariamente un producto funcional, se pueden utilizar alternativas para las mismas, siempre que se logre transmitir un fiel reflejo del funcionamiento y de lo que se pretende del producto mismo.

Para el primer producto planteado de un triviador de preguntas basado en la batalla naval, se generó una demo funcional en donde se mostraban el flujo principal del producto. Usuarios se loguean y entran al sistema. Visualizan a los usuarios conectados (dos listas, una con los amigos del usuario de una red social, y otra con el resto de los usuarios del sistema), luego procede a seleccionar uno de ellos y entra en juego de competencia con el usuario seleccionado. Se muestra el flujo normal del juego planteado.

Para los restantes productos se utilizaron herramientas alternativas para mostrar el funcionamiento y las ideas detrás de cada uno de los productos. Estas alternativas consisten en la elaboración de mockups y de interfaces de usuarios limitadas sin funcionamientos (se realizaron funciones que retornaban siempre los mismos resultados).

No importa cual alternativa se opta para utilizar a la hora de realizar una demo, siempre que se logre expresar las ideas detrás de cada producto.

No solo se buscó realizar demos para poder avanzar al siguiente paso, sino que en este tipo de demos se investigaron los tipos de tecnología posibles que se pueden utilizar en aplicaciones de Windows 8. Estas son de C#, VB, C++ y HTML5, CSS3 y JavaScript. En el caso de la demo funcional del juego de triviador de preguntas se optó su desarrollo por HTML5, CSS3 y JavaScript. La razón es incursionar en esta tecnología y medir la productividad. También para utilizar los estándares del mercado en la creación de aplicaciones móviles.

El medir la productividad en tecnologías es una utilidad que nos sirve para etapas posteriores del proyecto al momento de decidir sobre qué tipo de tecnología desarrollar el proyecto, y el planeamiento del desarrollo del producto.

4.1.3 Validación de productos

La validación de los productos se realizó en coordinación del equipo de desarrollo y Marcelo Guerra. La misma se llevó a cabo por medio de reuniones en las cuales con el uso de las demos creadas anteriormente se fueron mostrando los distintos productos formulados y sus respectivos funcionamientos.

Alcanzo con solamente una sola fase de validación de productos, no fue necesario regresar a generar nuevas ideas ni formulación de productos.

En esta actividad se llegó al acuerdo del desarrollo de uno de los productos formulados. Se desarrolla el triviador de preguntas con base de batalla naval y acoplamiento con redes sociales.

Para los demás productos formulados se consideran útiles pero por temas de alcance del proyecto en el tiempo se opta solamente por el desarrollo de uno solo. Estos productos descartados para este desarrollo pueden servir como productos a futuros desarrollos de proyectos similares.

Se elige la batalla naval basada en preguntas debido al tema de interacción de los usuarios entre ellos y redes sociales. Se ve como un potencial producto que puede llegar a cumplir de manera rápida los objetivos del proyecto en cuanto a cantidad de descargas. También es un producto que se acopla a los distintos perfile relevados de la investigación de productos y mercados.

De esta manera se establece la elaboración de un solo producto.

Al final de esta etapa se elabora un documento en donde se establece el producto a desarrollar (batalla naval basado en preguntas con base en la integración de redes sociales) (ver anexo).

4.1.4 Especificación del producto a desarrollar

El producto a desarrollar se le dio el nombre de "BattleShip Trivia", el mismo consiste en un juego de competencia entre usuarios de la propia aplicación. Busca la integración con redes sociales como Facebook, Microsoft, Google, Twitter, entre otras, logrando de esta manera fomentar la interacción y competencia de los usuarios.

BattleShip Trivia es un juego que basa su lógica en similares como el "Triviador Mundos". En la utilización de la metodología SIT se extiende el concepto del mismo a un juego conocido por la mayoría de los usuarios como es la batalla naval, surgiendo de esta manera un nuevo juego.

Una vez que se encuentra el usuario dentro del juego, se lo dirigirá a una sección denominada Dashboard, en la cual el usuario podrá visualizar las siguientes opciones:

- Sus amigos conectados (amigos de la red social que seleccionó) en el juego, y un conjunto de usuarios que no son amigos pero son usuarios de BattleShip Trivia. Dichas listas se actualizarán de manera automática y a demanda por parte de los usuarios en cualquier momento.
- Opciones de interacción como :
    - Cambiar idioma (de inglés a español o español a inglés).
    - Ver los ranking de usuarios (teniendo tres tipos de rankings “semanales”, “mensuales" y "anuales", y para cada uno de estos ranking la posibilidad de filtrar solo entre amigos o el total de usuarios).
    - Realizar comentarios de la aplicación (posibilidad de dar feedback a los autores de la aplicación, sobre cualquier aspecto que se considere necesario).
    - Ver los retos que ha recibido (funcionalidad que se explicará más adelante en el documento).

Una vez que el usuario ingresa en el dashboard, este puede retar o ser retado por usuarios para entrar a competir en un partida por puntos. La forma de retar que posee el usuario es por medio de la selección de un usuario (amigo o no) de las listas que se le muestran en la pantalla dashboard. Luego de retar a un usuario debe de esperar un tiempo predeterminado para saber la decisión del usuario retado de aceptar o rechazar el reto, en caso de que se acepte el reto se entra en un partida con el usuario seleccionado, en caso contrario se elimina el reto y se le habilita al usuario el dashboard para que vuelva a retar a nuevos jugadores.

En caso de que el usuario fuese retado por otro usuario de la aplicación, a este se le notifica por medio de un sonido el reto recibido, el mismo deberá de seleccionar la opción de ver retos del dashboard para que el sistema le liste los retos que tiene disponible. Una vez seleccionado un reto recibido el usuario dispone de un tiempo para decidir si acepta el mismo o si lo rechaza, si se acepta el mismo se entra en partida con el usuario retador, en caso contrario se cancela el reto (se lo elimina de la lista de retos) y se notifica al usuario retador del rechazo del mismo.

Una vez que un par de usuarios entran en una partida estos ya no se mostraran en las listas de los demás usuarios, lo que significa que no están disponibles para jugar, hasta que se termine la partida. Estando en una partida a los usuarios se le mostrara una pantalla en la cual se le da la posibilidad de colocar sus barcos en el tablero (con las mismas reglas de la batalla naval), en caso de que

se pase un tiempo establecido el sistema procede a colocar los barcos por el propio usuario.

Cuando ya se han colocado los barcos por parte de los dos usuarios de la partida, comienza el juego, el mismo consta de 10 rondas de preguntas de 4 opciones cada una (siempre con una única opción correcta), el usuario deberá de seleccionar una de ellas en un tiempo (en caso de que se pase el tiempo y no se seleccione ninguna se toma que el usuario no optó por ninguna). Aquel usuario que conteste bien la pregunta y en menor tiempo, es el ganador de la pregunta, al mismo se le mostrará el tablero del adversario y deberá de seleccionar a uno de sus casilleros para poder arrojar una bomba (utilizando la misma dinámica de la guerra naval), en caso de que el casillero seleccionado coincida con una de las posiciones de los barcos del adversario se le indica a los dos jugadores y se le dan puntos extras al jugador ganador de la pregunta, en caso contrario solo se le adjudican puntos por la pregunta contestada correctamente.

En caso de que ninguno de los jugadores conteste correctamente se proseguirá con la siguiente pregunta, sin dar puntaje a ninguno de los jugadores.

En todo momento en que el adversario conteste una pregunta se realizará una notificación al usuarios de que el contrincante ya ha contestado pero sin mostrar la respuesta del mismo. Al final de cada pregunta se muestra la respuesta del adversario y cuál era la respuesta correcta de la pregunta. El juego termina si se acaban todas las preguntas, o si un usuario logra atinarles a todos los barcos de su adversario. En este caso siempre gana el usuario que tenga mayor cantidad de puntos al final del juego. Los puntos logrados en cada partida se suman al ranking personal de cada usuario. Así mismo el juego también finalizará si uno de los usuarios abandona la partida, dándole la partida ganado a su adversario.

Al finalizar la partida se envían a los dos usuarios al dashboard de la aplicación, lo que los vuelve a dejar activos para nuevos retos. Una vez que el usuario abandona la aplicación, esta recuerda sus credenciales por un período de tiempo predeterminado (a no ser que el usuario cierre sesión de manera explícita), para que la próxima vez que el usuario regrese al juego este ingrese de manera inmediata al dashboard y no tenga que loguearse nuevamente por una red social.

Una problemática que se considera importante para esta aplicación es el inicio de la misma en el mercado. Los primero usuarios en registrarse no tendrían amigos y usuarios al azar con quien realizar partidas por lo que les resultaría un tanto incómodo, y se correría el riesgo de que no ingresen más en la aplicación. Por lo que se decide la realización de un jugador un inteligente.

El mismo consiste en un proceso que simula a una cantidad de 30 usuarios que nos amigos de nadie en la aplicación, pero siempre aparecerán activos para aceptar nuevos retos de cualquier usuario de la red.

Para esto se diseñará y desarrollará un algoritmo de que simule el comportamiento de un usuario estándar. En caso de ser retado decidirá si acepta el reto basado en un resultado de azar de 50%. Al ingresar en una partida con un usuario de la aplicación, se simulará todo el comportamiento de un usuario normal mediante estadísticas y azar. Para responder una pregunta se simula un tiempo de lectura y un tiempo de decisión de un usuario normal. Responderá basado en las estadísticas de la pregunta. Por ejemplo si la pregunta el 90% de los usuarios suele contestarla correctamente tendrá una posibilidad de contestar bien del 90%, o en caso de que sea 10% el porcentaje de respuestas correctas de la pregunta se tendrá un 10% de posibilidades de contestar correctamente. En caso de haber ganado la pregunta por parte del usuario simulado este realizará el tiro en el tablero de manera totalmente al azar.

Al mismo se les tendrá que mantener un perfil de la aplicación para simular a la perfección a un usuario, esto significa que aparecerán en los rankings globales de usuarios como cualquier usuario.

4.2 Etapa 2 – Elaboración

4.2.1 Asignación de Roles

Los roles asignados en esta etapa fueron:

- ✓ **Analista Funcional:** Los tres integrantes del proyecto tuvieron este rol, para poder definir las historias de la aplicación.
- ✓ **Arquitecto diseñador:** Germán y Alexis se dividieron las tareas de este rol, y definieron la arquitectura de la aplicación.
- ✓ **Scrum Master, QA Testing:** Rodrigo se encargó armar el plan de releases junto con los criterios de calidad y verificación.
- ✓ **Sysadmin:** Germán se encargó de definir y armar la infraestructura necesaria para el desarrollo
- ✓ **Diseñador de UI:** Los tres integrantes compartieron el rol y tomaron las decisiones de UI de la aplicación.

4.2.2 Story Map

En esta etapa utilizamos la técnica denominada: Visual Story Mapping. Esta es una herramienta que permite generar una representación visual de todo el sistema a construir. Ofrece una vista general de todas las funcionalidades que lo componen considerando flujos principales y alternativos con lo que respecta a los casos de uso. Permite identificar todos los flujos posibles para cada tipo de usuario que falten en el Product Backlog. Además permite planificar liberaciones

partiendo el sistema en grandes porciones y visualizar cómo se distribuyen las funcionalidades de acuerdo a las diferentes áreas del sistema.

Es una forma de reorganizar el Product Backlog en dos dimensiones, una dimensión para el tiempo (medido en Releases) y otra dimensión para las funcionalidades.

La construcción fue muy sencilla de realizar, por un lado se identifican las grandes funcionalidades en las que podemos dividir nuestro sistema (Activities) y situarlas en un orden lógico secuencial. Luego se descomponen esas actividades en las tareas que el usuario hace ordenadas secuencialmente. A partir de esto se identifican las sub-tareas que deben ser desarrolladas. Es importante tener en cuenta que cada una de ellas debe entregar valor al producto final. De esta forma es posible determinar la granularidad de las tareas así como las sub-tareas.

Una vez armado el mapa, podemos identificar a la fila de grandes funcionalidades (EPICs) como la columna vertebral de nuestro sistema (The backbone), estas no deben priorizarse y permiten dar contexto a las User Stories.

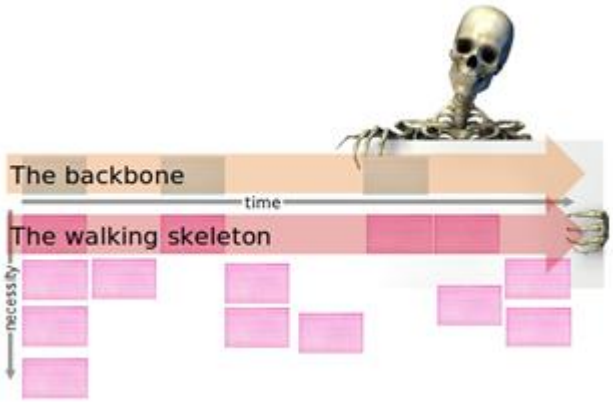


FIGURA 5

La fila de actividades es conocida como "The Walking Skeleton" (el esqueleto que camina) es decir las actividades más básicas que ofrecen una funcionalidad de punta a punta en nuestro sistema. Jeff Patton recomienda construir esto primero para luego mapearlas con las MMF (Minimum Marketable Feature, es decir las funcionalidades más elementales que se pueden poner en producción). Nosotros este término lo conocemos como MVP.

Las sub-tareas sí deben priorizarse, siendo las de más arriba las más críticas o prioritarias. Lo poderoso de Visual Story Mapping es que permite hacer recortes de funcionalidad a entregar. Los cortes se hacen con filas o líneas dibujadas. Cada fila indica un Release del producto indicando que funcionalidades se incluyen en cada una. De esta manera se puede ver a simple vista cuándo se incluirán determinadas funcionalidades, en que partes del sistema se está priorizando más desarrollo y detectar áreas donde no se están haciendo avances.

Concretamente en nuestro caso se identificaron como actividades

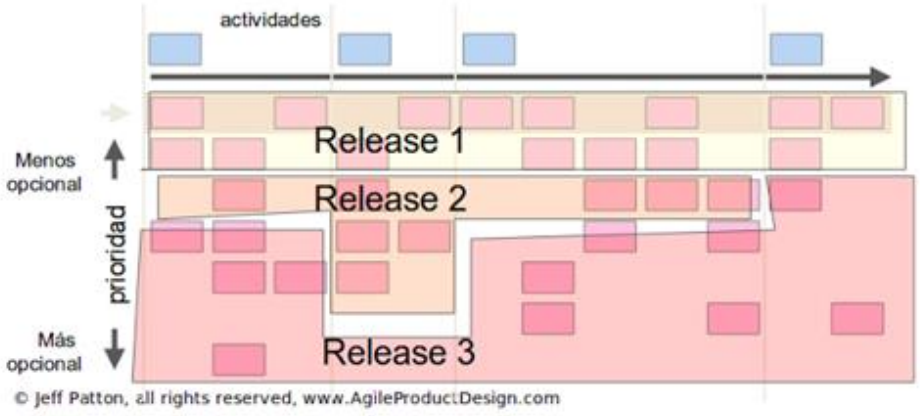


FIGURA 6

principales: Acceso a los Usuarios, Juego y Listado de Resultados. A partir de estas tres grandes actividades, Se desprenden:

- Acceso a los Usuarios:
  - Login con Facebook.
  - Login con Microsoft.
  - Logout Facebook.
  - Logout Microsoft.
  - Asociar Amigos de Facebook.
  - Asociar Amigos de Microsoft.
- Juego:
  - Retar
  - Listar amigos conectados.
  - Listar amigos generales.
  - Aceptar reto.
  - Rechazar reto.
  - Contestar Preguntas
  - Batalla Naval
- Puntuaciones
  - Ranking diario
    - General
    - Entre amigos

- Ranking semanal
  - General
  - Entre amigos
- Ranking mensual
  - General
  - Entre amigos
- Ranking global
  - Generar
  - Entre amigos

A partir de este se idearon dos Releases: el MVP y una segunda salida al mercado con funcionalidades sugeridas por los usuarios más algunas funcionalidades dejadas. Se priorizó la autenticación vía Facebook, dejando de lado Microsoft. Además se restringieron los rankings.

Una vez determinados los Releases, se pasó a escribir detalladamente las historias que componen el Product Backlog. Estas se escribieron con el siguiente formato: "*Como xxx,* quiero hacer yyy con el objetivo de zzz", donde, xxx es el usuario (quien), yyy es lo que el sistema debe permitir realizar (el qué) y zzz es el beneficio o valor buscado (el por qué).

Se utilizó esta técnica para identificar claramente las condiciones de satisfacción de las historias así poder determinar fácilmente las pruebas de aceptación que se realizarán, indicando cómo debe comportarse el sistema (o BDD, Behaviour Driven Development). Básicamente podemos reescribir las historias determinadas con el formato: "Dado aaa, cuando se produzca bbb, entonces ccc", donde aaa es la situación en la que se encuentra el sistema, bbb es un evento que lo hará cambiar y ccc es el resultado. Esta técnica permite evitar la aparición de errores por malos entendidos y evitar re trabajo (siguiendo los principios Lean). Por ello es recomendable no empezar a desarrollar en una iteración sin antes haber escrito los casos de prueba, especialmente porque es más barato escribir texto y pensar en cómo desambiguar los requisitos que arreglar errores importantes debido a su mal entendimiento.

Si bien es una forma bastante estructurada redactar las historias de esta manera, lo importante es poder identificar fácilmente lo que serían pre condiciones, post condiciones y acciones de una historia.

Las historias una vez creadas poseen los siguientes datos:

✓ Título: Breve descripción con el formato: "*Como xxx,* quiero hacer yyy con el objetivo de zzz."

✓ Tipo: Permite determinar el origen de la historia. Si es una funcionalidad nueva se utiliza: Feature y si la historia se origina de un error reportado lo indicamos como: Bug.

✓ Puntos: Estimación en puntos de la historia.

✓ Estado: Estado de la historia.

✓ Creador: Quien reporto la historia.

✓ Responsable: A quien fue asignada la historia.

✓ Seguidores: Permite notificar a seguidores en caso de modificación de la historia.

✓ Descripción: Descripción detallada de la historia.

✓ Categorización: Categorización que permite agrupar las historias por filtros. En nuestro caso fue utilizado para agrupar las historias por actividades generales del usuario. También fueron agrupadas por las distintas técnicas que utilizamos para reportar las mismas.

✓ Sub tareas: Sub tareas claves.

✓ Histórico: Un histórico de comentarios y actualizaciones que permiten mantener una trazabilidad de la historia.

FIGURA 7

Para la estimación se utilizó la técnica: Planning Póker y Piedra angular utilizando como unidad de medida los Story Point. Es importante enfatizar que se utilizó una medida de tamaño para estimar User Stories y no una de tiempo que indique la demanda de la construcción. La medición a través de tiempo está condicionada por cierta subjetividad. Y por otro lado, el paso del tiempo, no

representa necesariamente progreso en la construcción del producto. El concepto de las medidas de tamaño tiene que estar presente tanto en la sesiones de Planning como en las Daily Meetings, ya que esta medida de tamaño relativa es la que se tiene que utilizar para estimar el tamaño en el momento de planificar, como así también en el día a día para monitorear el avance del proyecto. Al estimar el proyecto con Story Points se ve al producto en su totalidad como un conjunto de puntos donde cada historia es una porción del mismo. Sumando cada una de esas porciones que le agregan valor al producto final Sprint tras Sprint obtenemos el producto. Utilizando esta unidad, podemos ver el avance de construcción del proyecto. Al estimar utilizando horas, se agregan un montón de factores subjetivos que no reflejan el avance en la construcción del proyecto. La dificultad, el tiempo de construcción, etc. Son factores que se ponen sobre la mesa a la hora de estimar con horas, y estos no reflejan cuanto producto se construye por unidad de tiempo o cuanto valor se está teniendo en lo que se hace.

La técnica de estimación piedra angular, utiliza un enfoque de comparación relativo entre historias. Para ello es necesario definir cuál es el punto de referencia a partir del cual por comparación se realizan las estimaciones de las distintas porciones del producto a construir. Este punto de referencia es la piedra angular (se utiliza este nombre por analogía a los arcos romanos). Este punto de comparación se utilizará a lo largo de todo el proyecto. Es una funcionalidad simple y conocida por todos. Entonces, en el momento de estimar cada User Story, se debe determinar cuánto más (o menos) compleja es la User Story que se está estimando en relación a la piedra angular.

Una vez estimadas las historias es necesario priorizarlas. La etapa de priorización se realiza en base al valor de negocio que aporta cada historia para el Product Owner. Una manera sencilla para determinar esto es dividir las historias en 3 grupos, según sean imperativas, importantes o cosméticas/prescindibles (de manera que si se llega a una fecha de entrega predeterminada y no se han completado por lo menos hemos aportado el máximo de valor posible). A partir de estas agrupaciones nos resultó más fácil realizar una ordenación relativa por valor y después asignarlo.

La prioridad puede cambiar todo el tiempo; pero el tamaño en Story Points debe mantenerse fijo con la estimación original (o sea: como regla general, no reestimar). Si aparecen historias nuevas, deben estimarse utilizando el mismo criterio que se utilizó originalmente.

Por último realizamos una primera estimación de cuánto tiempo durará la fase de construcción del proyecto. En base a la velocidad alcanzada en el prototipo realizado, podemos tener una primera noción de la duración del proyecto. Estimamos que llevará 4 Sprint de 40 puntos promedio cada uno. Básicamente así fue armado el Product Backlog.

4.2.3 Sprint 0

Se procederá a mencionar cada una de las tareas realizadas en dicha actividad con una descripción de lo que se realizó en la misma.

4.2.3.1 Documento de arquitectura

Se definió el uso del patrón: cliente-servidor, donde los cliente serán las aplicaciones Windows 8, que se comunican a través del protocolo websocket con el servidor. Y este es el que realiza la sincronización de las partidas entre los distintos jugadores. Se decidió por el protocolo websocket, ya que este es de mejor performance para la comunicación de los eventos del servidor con el cliente. Por otra parte, se va a tener un componente de base de datos, que el componente servidor se va a comunicar, para guardar estadísticas de las partidas de los usuarios, y otros datos que se detallaran a continuación en el apartado de diseño de base de datos.

4.2.3.2 Base de datos

Se realiza el diseño de la base de datos y se implementa el mismo en servidores de SQL Server 2012. Se utilizan servidores locales para realizar el desarrollo de la aplicación, en tanto para los ambientes de testing y producción se cuenta con servidores SQL Azure para realizar las tareas de los mismos. Se realiza un documento de descripción de la base de datos para facilitar la compresión de la misma por cualquier cambio que se realice en el transcurso del proyecto.

4.2.3.3 Infraestructura

Para la infraestructura se contó con un servidor dedicado, que fue una máquina virtual en Windows Azure. Allí se colocó un servidor de aplicaciones (IIS 8.0) que tenía toda la lógica de comunicación entre los jugadores y negocio de la aplicación. Por otra parte, al usar Windows Azure, se contó con otro servidor separado que el DBMS con nuestra base de datos.

Se creó en los mismos servidores, aplicaciones separadas a las de producción, como ambiente de testing

4.2.3.4 Testing

La verificación y el control de calidad de la solución se dividen en tres etapas. En una primera etapa se realzan pruebas de validación al final de cada Sprint tal como se definió en el proceso. La granularidad de estas fue sobre cada User Story definida y con los criterios antes mencionados. El diseño es basado en la técnica BDD.

Una vez finalizado un Sprint, se liberaba una versión a un ambiente de pruebas donde se evaluaba la calidad de la solución. Para ello, se corrían unos

conjuntos de test basados en casos de usos y flujos de la aplicación. Estos casos eran construidos en base a agrupaciones de User Stories, epics y criterios de aceptación. Este conjunto de pruebas era ejecutado por un equipo de verificación y calidad paralelo al equipo de desarrollo.

Al finalizar la ejecución de todos los Sprints por haber alcanzado el MVP del producto que se colocará en el mercado, se realizará una prueba de aceptación con usuarios seleccionados por el equipo del proyecto, de esta forma se podrá medir la relación de los usuarios finales con el producto obtenido, y realizar alguna mejora en caso de que se considere necesario antes de que el mismo quede publicado en la tienda.

En forma paralela se realizarán pruebas de performance sobre el producto en el servidor de producción, para conocer el desempeño del producto, de esta forma se conocerá con mayor detalle los niveles máximos de nuestro producto ante la demanda de los usuarios.

En nuestro caso se toma la decisión de no realizar testing unitario en el servidor del repositorio del código debido a que no se contaba con una infraestructura necesaria para la implementación del mismo en el instante en que se empieza el desarrollo del producto.

Como se apuesta a desarrollar un producto en una cantidad de cuatros Sprints para luego realizar una nueva iteración (a evaluar al final del desarrollo del proceso), y no contar con un gran número de integrantes en el equipo de desarrollo que aumente la productividad por Sprint se toma la decisión de no realizar testing unitario sobre las capas desarrollada en el sistema. En caso de contar con un número mayor de integrantes y de contar con mayor tiempo de desarrollo del producto se ve la necesidad de desarrollar testing unitario sobre cada capa desarrollada.

4.2.3.5 Seguridad

Para la seguridad, se tomó la decisión de que la única acción anónima sea iniciar sesión. Luego de que un usuario inicie sesión, al utilizar websocket para la comunicación con las aplicaciones clientes, cada request se da en alguno de los websockets de los usuarios logueados, no permitiendo acciones anónimas. Por otra parte, la comunicación a la base de datos, solo se puede dar desde nuestro servidor, ya que introducimos una restricción de acceso por IP.

4.2.3.6 Gestión de configuración de software

Para el versionado de código, se eligió la herramienta Mercurial HG, que es un sistema de control de versiones multiplataforma, para desarrolladores de software. Todas las operaciones de Mercurial se invocan como opciones dadas a su programa motor, hg (cuyo nombre hace referencia al símbolo químico del mercurio). Las principales metas de desarrollo de Mercurial incluyen un gran

rendimiento y escalabilidad; desarrollo completamente distribuido, sin necesidad de un servidor; gestión robusta de archivos tanto de texto como binario; y capacidades avanzadas de ramificación e integración, todo ello manteniendo sencillez conceptual. Incluye una interfaz web integrada.

Para la documentación se eligió Skydrive de Microsoft, ya que tiene un buen editor de texto online, y permite editar en línea concurrentemente.

4.2.3.7 Diseño de UI y UX

El diseño de la interface gráfica e interacciones con los usuarios se realizó en conjunto. Se diseñó una interface de usuario de tres pantallas, buscando poca navegabilidad y rápido acceso a las funcionalidades del juego. Se ambientó con barcos de guerra y sonidos de bombas y cañones buscando trasmitir un ambiente de competencia y rivalidad entre amigos. Ver anexo 2, sección de buenas prácticas.

4.3 Etapa 3 – Implementación

4.3.1 Asignación de Roles

Los roles asignados en esta etapa fueron:

- ✓ **Desarrollador:** Alexis, Germán y Rodrigo.
- ✓ **QA Testing:** Germán y Rodrigo.
- ✓ **Scrum Master:** Rodrigo.
- ✓ **Product Owner:** Germán.

4.3.2 Desarrollo

La etapa de implementación como se mencionó en la definición del proyecto consistió en la construcción del MVP definido en la etapa anterior. Para esto se utilizó la metodología de desarrollo de Scrum, por lo cual se dividió el desarrollo del producto en Sprints en los cuales se incluyeron historias del Story Map definido previamente.

La duración de esta etapa fue de 3 meses de Julio de 2013 a Septiembre 2013, con un total de 6 Sprints desarrollados.

Los Sprints se desarrollaron con una duración de 2 semanas cada uno tal como se había establecido en la definición del proyecto. Al final de cada uno de ellos se realizó una reunión del equipo de desarrollo en la cual se validaban o rechazaban las historias desarrolladas (con el criterio de "todo o nada"), se obtenían las medidas establecidas en la calidad, y se planeaba el siguiente Sprint y se realizaba un análisis sobre el Sprint que se acababa de cerrar, en los mismo se ponía a estudio si era recomendable realizar alguna modificación al desarrollo o continuar de la misma forma en que se encontraba desarrollando.

También se realizaban las reuniones diarias del Scrum tal como se planteó en la definición del proceso con una separación de 2 a 3 días cada una de ellas. En las mismas se le dio un contexto más informativo acerca de en qué situación se encontraba cada uno de los desarrolladores en dicho momento con respecto a las historias que se hizo cargo.

Desde el segundo Sprint hasta el final se introduce una tercera técnica de estimación debido a un atraso en la implementación. Se decidió al comienzo de cada Sprint sub dividir las tareas en sub tareas y estimar con horas cada una de ellas para luego verificar si cada User Story estaba estimada correctamente. Estos se deben a que la velocidad de avance bajó debido a que las historias correspondientes a la interface de usuario abarcaban actividades ocultas. Se usó esta nueva forma de estimar en forma paralela a las dos definidas por el proyecto.

Con el desarrollo de cada historia en un Sprint dado se introducen posiblemente nuevas funcionalidades a la aplicación ya sea del lado del servidor como desde la propia aplicación.

Se procederá a realizar una introducción de la realización de cada uno de los seis Sprints de la etapa de implementación. Posteriormente se realizará un análisis estadístico de las medidas obtenidas en cada uno de los Sprints.

4.3.3 SPRINT 1

El enfoque se estableció para este Sprint fueron de historias de manejo de usuarios en la aplicación desde la lógica de negocio y construyendo una interfaz de usuario que permitiera la interacción de las funcionalidades que se construía con un usuario final.

Al finalizar este Sprint se cuenta con un gran avance de las historias de los usuarios dentro de la aplicación sin contar cuando estos entran en una partida.

4.3.4 SPRINT 2

El objetivo en este Sprint era finalizar la lógica de la parte del dashboard del juego, y el comienzo de la lógica de las partidas entre los jugadores. Por otro lado, se inició el proceso de mejorar aspectos de UI de la pantalla de presentación y la del dashboard, agregándole animaciones y mejoras.
Luego de haber ejecutado el primer Sprint, se realizan mejoras en las estimaciones de algunas historias, que se consideraron mal estimadas.

Cuando se finalizó el Sprint, se cumplió con el objetivo de finalizar las historias de lógica, y el comienzo de las partidas de los usuarios, y se llegó a un 25 o 30 % de las pantallas de inicio y dashboard.

Como conclusión luego de finalizar este Sprint, se agregaron nuevas historias de UI que notamos que faltaban, e indicamos que las historias de UI

estaban mal estimadas, lo que nos hizo dar cuenta, que el producto iba a estar finalizado en 2 Sprints adicionales de lo que en un principio se había considerado.

En base a la inestabilidad de la velocidad del grupo se decidió hacer una variante a la hora de planificar el Sprint con respecto a las estimaciones. A partir de este Sprint, cada historia se dividió en un conjunto de tareas a realizar, donde cada una de estas tareas se estimó basándonos en el esfuerzo en horas que lleva resolverla. Esto nos permite identificar si existe alguna tarea oculta que no fue considerada o algún esfuerzo extra no detectado. Entonces, una historia la podemos medir con Story Points y además con el tiempo que lleva desarrollarla. A partir de esto podemos trazar un paralelismo entre puntos y horas, lo que nos permite determinar si una historia está bien estimada. Al comienzo de cada Sprint al tener la cantidad de horas que van a ser dedicadas al desarrollo, además es posible determinar si las tareas asignadas no exceden esa cantidad de horas.

4.3.5 SPRINT 3

Para este Sprint se empieza a mejorar en las estimaciones desarrollo, debido a las experiencias de los dos Sprints anteriores.

El enfoque que se establece en este Sprint es finalizar las historias restantes de lógica de partidas de la aplicación así como las restantes historias relacionadas a la lógica de negocio de la aplicación. Para las nuevas funcionalidades desarrolladas siempre se construyó una interfaz de usuario que permitiera realizar un testeo de las funcionalidades desarrolladas.

En cuanto a historias de interfaz de usuario se buscó avanzar con la construcción de las pantallas de inicio (buscando finalizar la misma) y dashboard (completar las historias seleccionadas para el Sprint).

Además comenzaron los primero resultados con respecto a la calidad de la solución, por lo que se asignaron horas a la corrección de errores.

4.3.6 SPRINT 4

Este Sprint tuvo como objetivo la finalización de las pantallas de presentación y dashboard, se agregaron todas las animaciones faltantes y se corrigieron todos los detalles que restaban. Por otro lado, se corrigieron distintos bugs que fueron encontrados en la lógica de las partidas y retos.

Se agrega una nueva historia a este Sprint que consiste en realizar optimizaciones en las operaciones de una partida para jugadores automáticos (la historia fue desarrollada pero sin ser lo más performance que se podía desarrollar).

4.3.7 SPRINT 5

En este Sprint se marca el inicio del desarrollo enfocado solamente a la interfaz de usuario de las partidas. Aparte de corregir cualquier error que se pueda encontrar durante el desarrollo del mismo o de integración que se está desarrollando con su parte lógica.

Aquí se marca el inicio de historias de desarrollo del sitio web público del producto que se encuentra desarrollando.

Desde este Sprint en adelante se encuentra un producto maduro con respecto a errores encontrados, lo que no significa que no se encuentren nuevos.

4.3.8 SPRINT 6

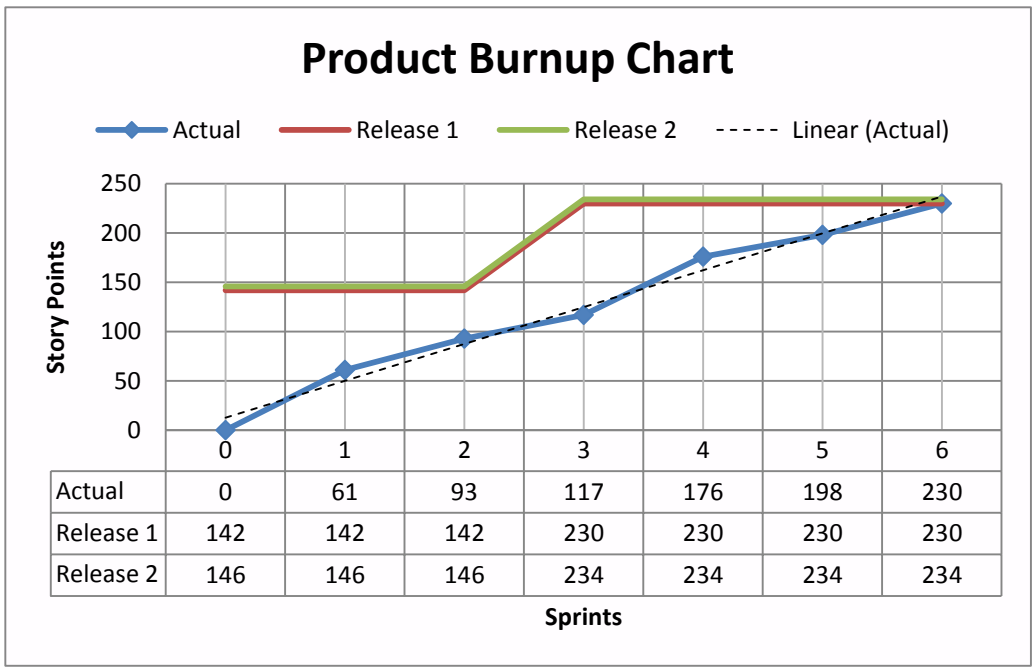
En este Sprint, se finalizó el desarrollo de las partidas entre los usuarios. Se corrigieron errores encontrados del Sprint anterior, y se dio por finalizado el desarrollo del producto, alcanzando el MVP.

Por otra parte, se finaliza el sitio web publicó, se compra un dominio (www.battleshiptrivia.net) y se pone en producción como promoción de la aplicación.

4.3.9 Resultados obtenidos

En esta sección presentaremos los resultados obtenidos en base a las métricas consideradas. De esta manera, con herramientas objetivas se evaluara el desempeño de la fase de implementación del proceso. Los valores fueron obtenidos en base a la ejecución de los seis Sprint.

En la gráfica 1 podemos observar la velocidad con la que se construyó el producto. Sobre el eje horizontal los 6 sprint que fueron necesarios para la construcción del MVP. En el eje vertical indicamos los Story Points. Las líneas rojas y verdes corresponden a los objetivos a alcanzar, es decir, la primera y segunda liberación oficial al mercado.



	0	1	2	3	4	5	6
Actual	0	61	93	117	176	198	230
Release 1	142	142	142	230	230	230	230
Release 2	146	146	146	234	234	234	234

Con esta herramienta a lo largo del proyecto obtenemos una estimación de cuánto tiempo estaría faltando para salir al mercado. El objetivo inicial fue liberar el producto en víspera del Black Friday ya que es la etapa del año donde se realiza mayor consumo de dispositivos con Windows 8. Al salir antes de esta fecha, aprovechamos la publicidad extra de la tienda ya que la aplicación será visualizada en el área de novedades.

Podemos observar un salto de 88 Story Point en el final del segundo sprint. Esto se debe a un reajuste de las estimaciones con respecto a las historias correspondientes a las interface de usuario junto a una re planificación de la granularidad de las historias de dicho tipo. Se relevaron animaciones y efectos a agregar para obtener una interacción con el usuario más amigable. Esto lo que llevo fue a postergar la salida a producción dos Sprints mas tarde. Básicamente fue un sprint más por nuevos requerimientos, más un sprint extra por el ajuste de estimaciones.

En cuanto a las estimaciones iniciales, con respecto a los requerimientos contemplados en la primera liberación al mercado, de un total de 146 Story Points, se desarrollaron 186. Esto indica un 22% de desvío con lo que respecta a la estimación inicial.

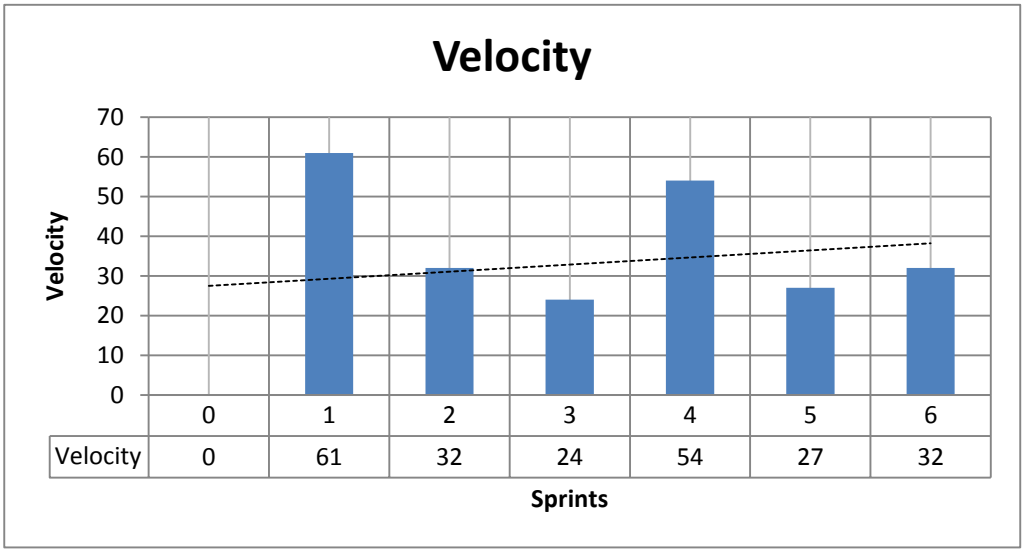
Si extrapolamos los puntos más alejados a la línea de tendencia, podemos observar que la salida al mercado siempre estuvo acotada a un sprint antes o un sprint después de lo pactado. Es importante destacar esto ya que en un proyecto de seis Sprints, tener un sprint de desviación es un margen muy bueno ya que hablamos de un 16% de error en las estimaciones.

El siguiente gráfico fue construido a partir del anterior. En él se refleja el proceso que llevo la estabilidad de la velocidad de construcción tras sucesivas iteraciones. Para ello se indican en el eje horizontal las iteraciones realizadas mientras que en el eje vertical cantidad de Story Points correspondientes a historias aceptadas.

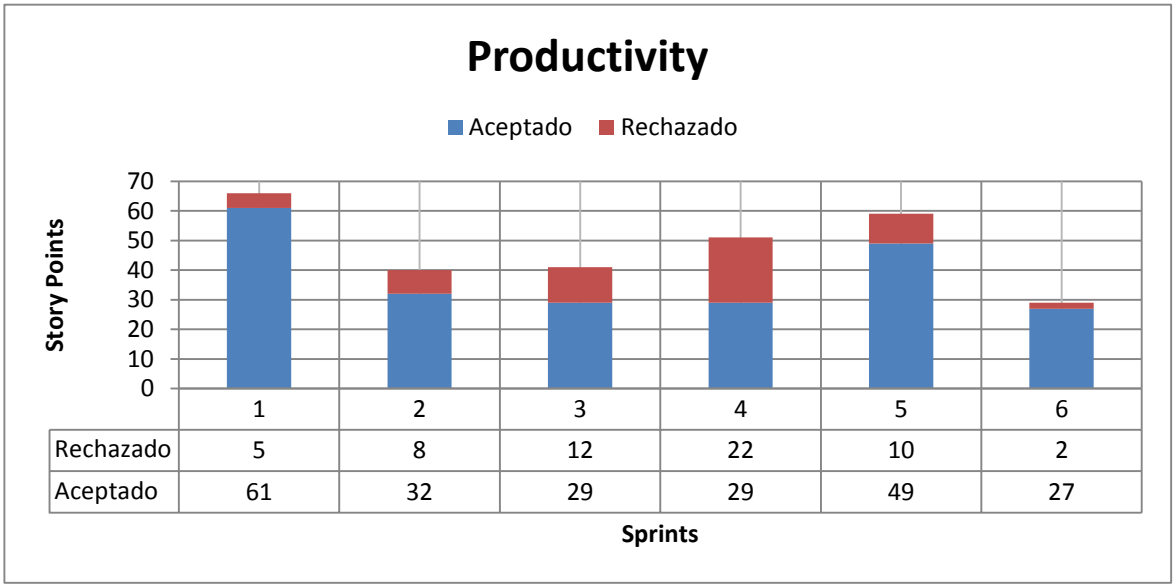
Es importante observar que la velocidad la medimos en función de cantidad de producto elaborado que supere los criterios de calidad y aceptación básicos por Sprint. Esto significa que por más que se construyeran User Story por 75 Story Points en un sprint, si solo 61 de estos fueron aceptados, la velocidad será 61, quedando 14 Story Points para el próximo sprint. Recordamos que utilizamos el criterio de aceptación "Todo o nada". Es importante considerar que toda historia rechazada, iba a quedar alojada al comienzo del Product Backlog para ser considerada en el próximo Sprint. Como resultado de esta decisión, podemos observar las grandes diferencias de avance entre el tercer y cuarto sprint. Muchas historias correspondientes a interface de usuario fueron rechazadas por diseño.

Por otro lado, la sustancial diferencia en el primer Sprint, se debió a que muchas User Story cerradas en dicho sprint, fueron historias ya construidas en el prototipo, por lo que no demandaron mucho trabajo de construcción sino de ajustes.

En síntesis, varios de estos inconvenientes detectados fueron corregidos logrando así aumentar la velocidad y llegar a una estimación realista a cerca de cuanto nos podría llevar el último Sprint.

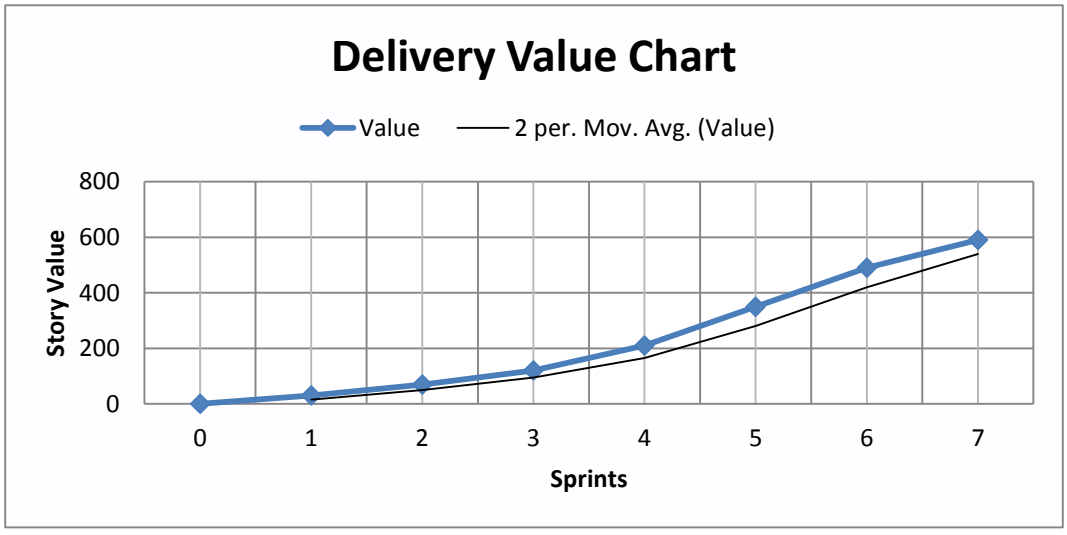


En la siguiente grafica podemos ver cuál fue la productividad efectiva en el trascurso de cada sprint. Como podemos observar salvo a los casos excepcionales, mencionados anteriormente, la velocidad de construcción se mantuvo constante y mediante el transcurso de interacciones se logró aumentar la cantidad de Story Points que cumplen los criterios de aceptación.



	1	2	3	4	5	6
Rechazado	5	8	12	22	10	2
Aceptado	61	32	29	29	49	27

En el siguiente gráfico, podemos ver el valor entregado al cliente por sprint. Para ello graficamos Story Value en función de sprint. Es importante destacar que al tratarse de un proyecto donde el cliente es el equipo de desarrollo junto con Marcelo Guerra, el gráfico intenta reflejar el valor agregado a este último. Como mencionamos anteriormente, cada User Story posee dos unidades de medida, por un lado Story points y por otro Story value. Este último valor, se estimó en función a cuanta importancia le da el cliente a una funcionalidad.

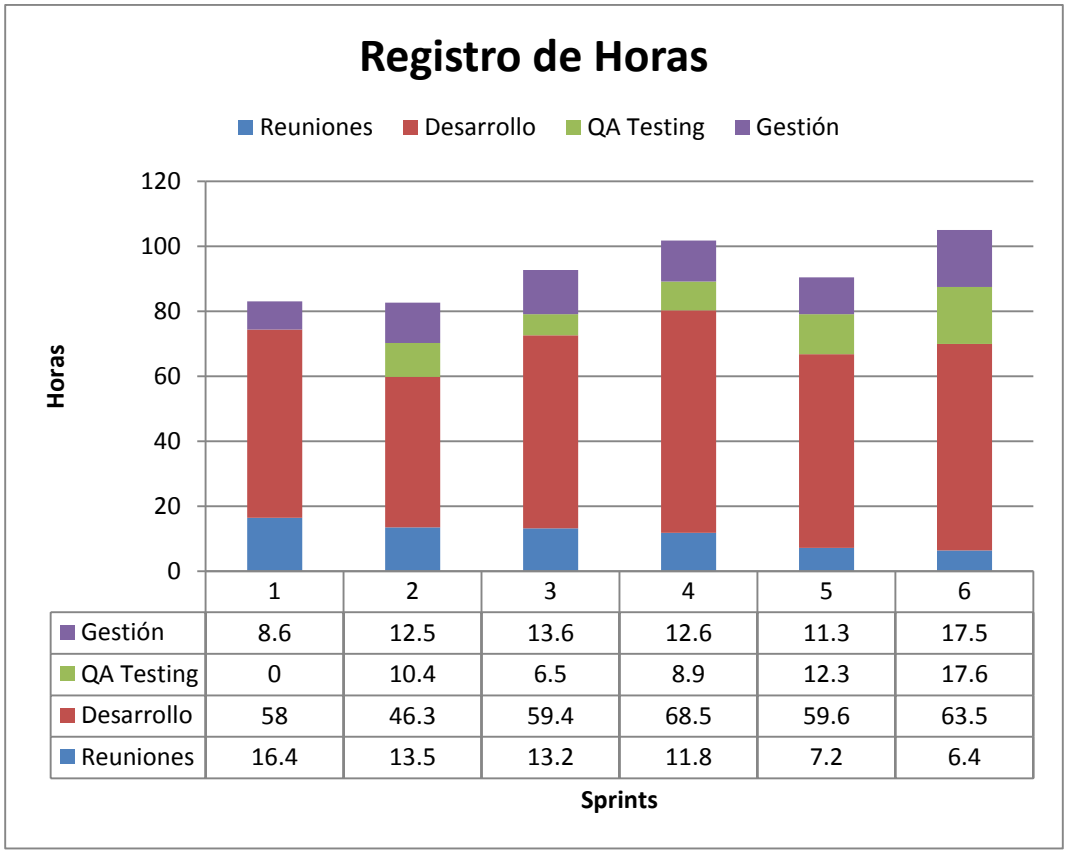


Como podemos observar al comienzo fue limitado el valor entregado al cliente. Esto se debió, a un pequeño atraso con lo que respecta a las interface de usuario ya que las historias fueron desarrolladas pero no podían ser mostradas por falta de baja calidad en la interface de usuario. Al re ajustarse las estimaciones en el segundo sprint, creció de manera deseada el valor del producto para el cliente.

A continuación podemos observar la distribución horaria de cada sprint. Junto con las horas detalladas para cada integrante. Con el transcurso del proyecto se fue consolidando la metodología de desarrollo, abarcando más horas de desarrollo y menores horas de gestión.

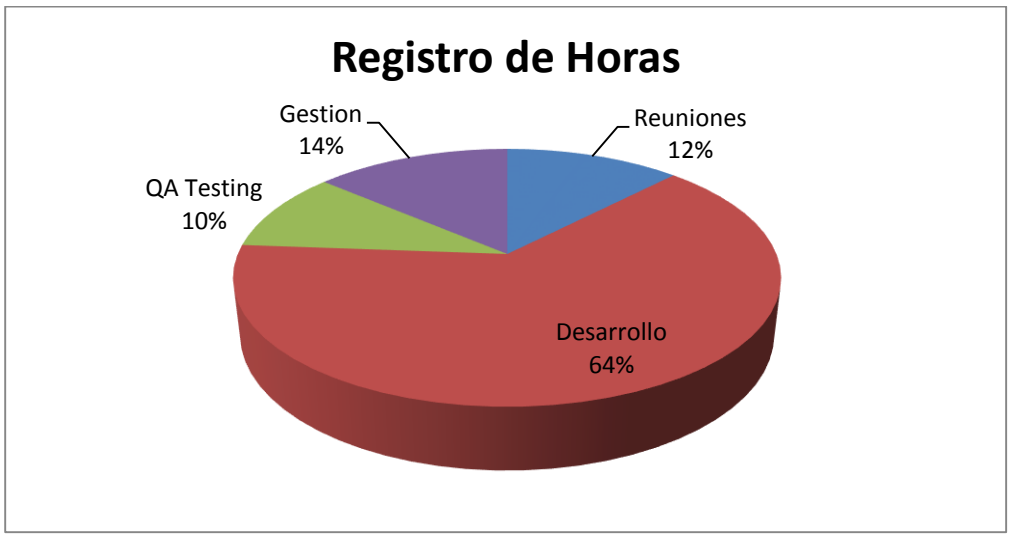
A la hora de planificar cada Sprint se decidió asignar las tareas en función al histórico de horas realizado, de esta manera manteníamos una relación entre horas de desarrollo, verificación y gestión. Es importante destacar esto porque al ser los equipos de Scrum auto organizados, no necesariamente las actividades de un programador se centran solamente en desarrollar código. Además, asiste a reuniones diarias y semanales o documenta la solución. Consideramos un grave error cometido asignar el 100 % de las horas disponibles a programar.

Por otro lado, se manejó la alternativa de registrar porcentajes de rendimiento en función a las horas. Tal como es explicado en la sección de métricas. Evaluamos registrar este valor y decidimos que no valía la pena sub dividir el registro en rendimiento.



	1	2	3	4	5	6
Gestión	8.6	12.5	13.6	12.6	11.3	17.5
QA Testing	0	10.4	6.5	8.9	12.3	17.6
Desarrollo	58	46.3	59.4	68.5	59.6	63.5
Reuniones	16.4	13.5	13.2	11.8	7.2	6.4

Como resultado obtuvimos las siguientes relaciones:



4.4 Etapa 4 - Liberación

Esta etapa cuenta de dos secciones, una en la cual el objetivo principal es la liberación del producto al mercado y su mantenimiento evolutivo, y la otra de conclusiones sobre el proceso llevado a cabo desde el ambiente de proceso e implementación.

La asignación de roles de esta etapa fue la siguiente:

- ✓ **Encargado de Deploy**: Germán fue el responsable de subir la aplicación a la tienda.
- ✓ **Soporte**: Alexis fue el encargado de dar soporte de la aplicación en producción.

Por otra parte, los tres integrantes fueron los encargados en sacar las conclusiones del proceso.

4.4.1.1 Liberación

Una vez finalizada la construcción de nuestro MVP, se realizó la puesta en producción del mismo en la Tienda de aplicaciones de Windows. En primer lugar, se tuvo que obtener una licencia de desarrollador por parte de Microsoft, necesaria para poder subir aplicaciones a la tienda. Esto nos llevó un par de días, ya que tuvimos que conseguir una tarjeta de crédito internacional, para poder pagar la licencia. Luego de obtenida la misma, se realizó la ejecución del Release, que consistió en ir llenando un formulario de 7 pasos, donde el primer paso es seleccionar el nombre de la aplicación en la tienda de Windows. Luego se debe llenar la descripción de la aplicación, subir capturas de imágenes, indicar con que

palabras los usuarios pueden encontrar la aplicación cuando realicen búsquedas, indicar la directiva de privacidad, y algunas otras características más.

Como nuestro producto requiere la característica de conexión a internet, Microsoft exige que la edad mínima con la que un usuario puede utilizar la aplicación debe ser 12 años, y al ser un juego, exige que un juego debe tener un certificado internacional donde indica que el contenido del juego es apropiado para la edad que se sugiere, en este caso para los mayores de 12 años. Para América del Sur, América del Norte, y ciertos otros países del mundo, la organización internacional es ESRB (http://www.esrb.org/) mientas que para Europa es PEGI (http://www.pegi.info/es/). La obtención de ambos certificados nos llevó aproximadamente una semana.

Para otros países de Asia, se exige un certificado de otras organizaciones, como CERO o GRB, pero nos fue mucho más complicado poder obtener el certificado, por lo que se descartó la salida a producción para algunos países asiáticos.

Luego de obtenidos estos certificados, realizamos el primer intento de puesta en producción, donde se sigue el Proceso de certificación de una aplicación por parte de Microsoft (Anexo 4). El mismo tiene un tiempo estimado de 10 días si previamente no se detecta que no se cumple con las políticas de la tienda de Microsoft.

Nuestra primera versión de la aplicación fue rechazada, porque no existía un link desde la aplicación, a la política de privacidad. Una vez hecha esta corrección, se realizó un nuevo Release del producto en la tienda.

Esta vez, luego de los 10 días, nuestra aplicación paso la certificación de Microsoft y se encontraba en la tienda.

Una vez que se colocó la aplicación en la tienda de Windows Store, se comenzó a recibir comentarios por parte de los usuarios que descargaron el producto, así como también amigos y conocidos a los que le mostramos la versión definitiva. La idea es obtener Feedback desde felicitaciones, desagrados o sugerencias que nos permita identificar mejoras en el producto.

4.4.1.2 Calidad de la solución

Se realizaron entrevistas con allegados al equipo de desarrollo de manera de medir la usabilidad de la interface gráfica. En base a los resultados, se decidió mejorar la amigabilidad. Si bien no se registraron resultados, todos los entrevistados comprendieron la dinámica del juego rápidamente y lograron realizar las acciones pedidas en un tiempo razonable.

A cada entrevistado se le solicitaba que intente jugar una partida contra un usuario al azar, dándole la aplicación cerrada. A partir de allí, monitoreábamos la

secuencia de acciones que realizaba y en caso de trancarse se le interrogaba a cerca del juego. La idea es, relevar impresiones de los usuarios ante el juego y determinar si la interface gráfica comunica lo deseado.

Otro atributo de calidad identificado fue la robustez de la solución. Es deseable no generar ninguna mala experiencia en un usuario ya que este podría calificar negativamente la aplicación en la tienda. Las pruebas de robustez se encargó de realizarlas la tienda de Windows por lo que no se registraron resultados de las mismas. Se detalla en el anexo: X, el detalle del conjunto de pruebas efectuado.

PRUEBAS DE CARGA

Se realizaron pruebas de carga sobre la aplicación con el objetivo determinar cuáles son las condiciones límite con las que el sistema interactúa correctamente. Para ello se consideró la obtención del ranking de los amigos como un servicio critico que puede condicionar la carga de la pantalla principal.

La obtención del Ranking de los amigos del usuario ingresado, se utiliza a la hora de mostrar los mismos en la pantalla principal. Al ser este ranking cargado por defecto en la pantalla inicial, es de vital importancia registrar sus condiciones límites.

El servicio se probó contra producción antes de salir la aplicación al mercado, con 10000 registros en la tabla de puntuaciones. El cliente fue un navegador iexplorer 10 instalado en una Notebook con procesador: Intel Core2Duo con 3GB de memoria RAM y Windows 8. Este se encontraba sobre una red WIFI con salida a internet de 2.0 Mbps.

Estos fueron los resultados obtenidos:

Request	Demora (Segundos)	Responses/Segundo	Fallos
100	26.635	3.7488	0
1000	230.209	4.3439	0
2500	596.453	4.1914	0
5000	1252.548	3.9917	0
10000	2620	3.8168	0

Se puede observar cómo tiempo de respuesta medio se mantiene oscilante en 4 responses/segundo. No se decidió probar con más datos por el tiempo de demora total.

4.4.2 Conclusiones

En esta sección se concluye en base a los resultados obtenidos luego de completar un ciclo de ejecución del proceso.

En cuanto a las etapas de elaboración e implementación, se logró poner en práctica la metodología de trabajo diseñada, la misma surgió de la combinación de varias metodologías y herramientas existentes (Scrum, XP, BDD, etc.). Para ello se acudió a la bibliografía existente y a blogs, foros y papers donde se exponen sugerencias y consejos en base a experiencias personales muy ricas. Se buscó, tomar como base Scrum y agregarle las disciplinas de calidad y verificación de menara de obtener un producto de calidad con un buen impacto en el mercado de Windows 8.

En base a la combinación de herramientas como fueron: Planning Póker, piedra angular, BDD, división de sub tareas, etc., logramos estabilizar un metodología de desarrollo para enfocarnos en una última puesta en producción. Esta es independiente de los lenguajes de programación así como del mercado de aplicaciones móviles al que apuntamos.

Encontramos en la técnica de investigación y puesta en práctica una buena técnica para construir la solución presentada ya que mediante sucesivas refinaciones, fuimos puliendo la metodología de desarrollo presentada como ideal para el contexto de negocio que nos rodeaba.

En cuanto a la planificación rescatamos la combinación de las técnicas: Planning póker, piedra angular y sub división de tareas, en su conjunto como una buena herramienta para la planificación. Por un lado, orientar las estimaciones en función a comparaciones por tiempo de dedicación, por otro lado considerar el tamaño del producto que se está construyendo y por ultimo re afirmar estas dos visiones con una sub división de tareas que permita por un lado verificar las sub componentes detrás y sus respectivos tiempos de trabajo.

En cuanto a la verificación y calidad de la solución, se estabilizo el procedimiento planeado. Mediante sucesivas sub etapas de control de calidad se buscó entregarle al cliente al final de cada sprint un producto de calidad básica, evitando un proceso pesado y tradicional de control.

Como lecciones aprendidas nos queda:

- ✓ Mejorar las demos realizadas, sobre todo mitigar todos los aspectos que hagan al desarrollo del producto, en nuestro caso mejorar las mitigaciones de aspectos con interfaz de usuarios.
- ✓ Realizar las estimaciones de desarrollo con una cota de error, para que de esta manera si ocurren sucesos que generen atrasos se puedan amortiguar de mejor manera.
- ✓ Mantenimiento de una comunicación fluida y constante con los clientes del proyecto, para que de esta manera se mitiguen de manera temprana aspectos que no se entendieron correctamente desde un principio.
- ✓ Utilización de una nueva técnica de estimación, que permite mejorar la performance en los tiempos.

Una vez realizada las conclusiones del proceso, se toma la decisión de continuar con un segundo release ya que la aplicación obtuvo más de seiscientas descargas el primer mes. Si bien ya se completaron los objetivos del proyecto, se decide continuar tal como se había pactado desde un principio (realizando dos salidas al mercado).

Se asume que los aspectos que tuvieron mayor déficit en la primera ejecución del proyecto se encuentran mejor mitigados para esta segunda ejecución debido a la experiencia previa.

Para ello durante 20 días recolectamos las sugerencias de los usuarios, y seleccionamos las siguientes propuestas:

- ✓ Mejoras en aspectos de UI: Mejorar ciertos aspectos de ui que fueron marcados por los usuarios.
- ✓ Inicio de sesión con otro sistema: Surge de buscar una alternativa al inicio de sesión.
- ✓ Generación de preguntas por parte de usuarios: Esto permite un auto mantenimiento por parte de la comunidad que utilice el juego, y así no se precisaría un mantenimiento permanente del producto.
- ✓ API De preguntas: Este último ítem, surge debido a que nuestro producto en realidad se pudo haber visto beneficiado por una api que contenga preguntas, y así no utilizar el tiempo de crearlas, y poderlo enfocar en otro aspecto importante del desarrollo. También debido a que nuestro producto en realidad se puede extender, a hacer otro totalmente distinto a la batalla naval, pero que utilice la Trivia como base, y para ello, el api de preguntas resulta muy interesante.

4.5 Segunda ejecución

Al finalizar el proceso con la puesta en la tienda de Windows Store del producto desarrollado, se optó por la realización de un segundo release a la tienda tal como estaba establecido en la letra del proyecto. Este segundo release se realizó por medio de la ejecución nuevamente del proceso definido. El mismo tuvo una duración de 1 mes de desarrollo (noviembre 2013).

Se establece que se desarrollara una nueva versión del producto y no un producto nuevo.

Es esta segunda ejecución del proceso se aprovecha el feedback obtenido por medio de la aplicación de la tienda de Windows Store como fuente de ideas para la nueva versión del producto. Así como también ideas o gustos personales que se quieran desarrollar y pasen por las etapas debidas.

Se procederá con el mismo formato que la ejecución anterior, se especificara la etapa de desarrollo y su ejecución.

4.5.1 Etapa 1 - Ideas

Dicha etapa se comenzó a desarrollar una vez que se colocó el producto en la tienda y se empezó a captar feedback de la aplicación por parte de los usuarios. Como se mencionó anteriormente se utilizó estos comentarios de los usuarios para realizar las nuevas ideas de la aplicación. No solo de los comentarios de los usuarios se realizaron las nuevas ideas, sino que desde gustos personales también se obtuvieron nuevas ideas.

Basados en estas nuevas ideas en esta ocasión no se realizaran las etapas de "Formulación de productos" y "Validación" (ya que no se formula ningún nuevo producto sino que se extiende uno ya existente). Esta vez se realizó un filtro de las ideas que se podían llegar a desarrollar y por medio de la intervención del cliente Marcelo Guerra en conjunto con el equipo de desarrollo, se decidieron cuáles de las ideas se incluiría en el desarrollo de la nueva funcionalidad.

Algunas de las ideas más importantes desarrolladas que surge por medio de los comentarios de los usuario de la aplicación es la de crear preguntas para el juego por parte de los propios usuarios. En conversaciones con los Stakeholders del proyecto, se llegó al acuerdo de extender el concepto de creación de preguntas a elaborar un ambiente en donde los propios usuarios de la aplicación sean los encargados de innovar y mantener las preguntas del producto. Como así aumentar el alcance de las preguntas por medio de una API pública de preguntas para servir como motor a cualquier proyecto de similar índole.

4.5.2 Etapa 2 - Desarrollo

En esta etapa se desarrollan las tareas de Story Map y Sprint 0 del proceso que se ejecuta.

En el Story Map se establecieron las nuevas historias que se incluirían en el producto, ordenando las mismas por una calificación de prioridad a desarrollar.

En el Sprint 0 se actualizaron los artefactos creados del anterior tarea de Sprint 0, para que se incluyan las nuevas características en caso de que se corresponda.

4.5.3 Etapa 3 - Implementación

La implementación de las historias correspondiente a esta nueva ejecución del proceso se llevó a cabo en un solo Sprint. Al final del mismo se validaron todas las historias que se incluyeron para el desarrollo de todo el producto.

4.5.4 Etapa 4 - Liberación

Para esta fase, se mantuvo el release al mercado de aplicaciones durante 1 semanas. El motivo del mismo se debió a que se consideró lanzar la nueva versión cerca de la fecha de navidad, la cual es considerada una fecha clave en los aspectos de negocio.

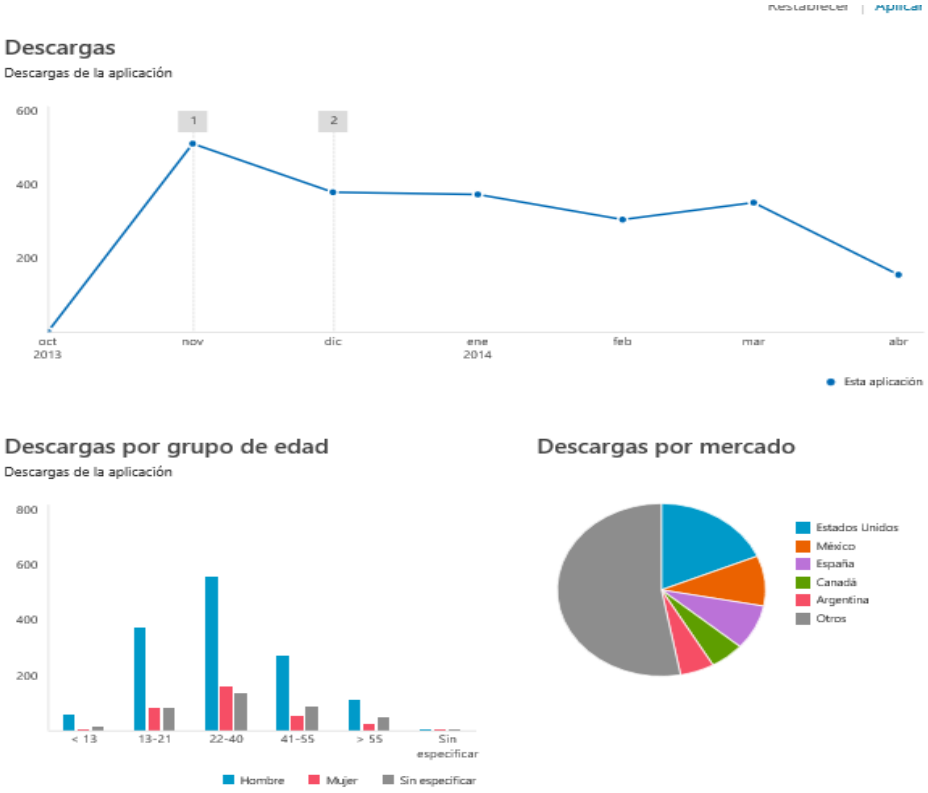
5. Conclusiones finales

5.1 Resultados Obtenidos

Luego de 5 meses de salido el producto al mercado, existe una descarga constante de aproximadamente 400 descargas mensuales. Estas corresponden en su mayoría a hombres aproximadamente entre 22 y 40 años de edad. Creemos que se debe al diseño militar y de guerra que realizamos de la interface gráfica.

Además se re afirma la hipótesis manejada desde un principio, mayoritariamente apuntamos al mercado estadounidense eligiendo la aplicación en ingles además del español. En base a los análisis de mercados realizados, este era el más grande en base a dispositivos y tablets con acceso a la tienda de Windows.

Podemos decir que se cumplió el primer objetivo planteado, construimos y publicamos un producto con más de cien descargas. Consideramos que el éxito se debe a no cobrar la misma, y a su diseño ante funcionalidad. De todas maneras presentamos un modelo de proceso que permite ir validando toda hipótesis generada.



También se crea un producto que perfectamente puede servir como base de un framework de juegos de estilo trivia. En nuestro caso el leitmotiv elegido para el juego fue la batalla naval, pero perfectamente se puede convertir en cualquier otro juego de trivia sin mayores esfuerzos.

5.2 Proceso Construido

Como conclusión general, se construyó un nuevo modelo de proceso para construir productos comercializables en la tienda de Windows 8. Este surgió en base al estudio de metodologías agiles y metodologías para la creación de un negocio exitoso. Se tomó como base este último dividiéndolo en cuatro etapas bien distinguidas y se lo combino con Scrum, XP, BDD, Planning Póker, planificación piedra angular, etc. Además se construyó en factor a análisis de mercados paralelos al mercado objetivo y mediante relevación de casos exitosos den los mismos.

Básicamente en las etapas de elaboración y construcción del producto, se combinó Scrum con normas de calidad y actividades de verificación ya que este no especifica nada en estas disciplinas.

Se lograron un total de 2500 descargas 6 meses luego de la primera salida al mercado y un producto por continuar desarrollando, basándolo en la aceptación y respuesta de los usuarios.

El proceso género no solo como salida el producto deseado con los tiempos y costos planificados, sino que también genero una base de conocimiento y una metodología estable que sigue enriqueciendo la misma. La base de esta metodología, es el principio de conocimiento validado propuesto por Eric Ries [1]. La idea de este proceso es generar un producto en continuas internaciones en base a formulación de hipótesis y validación de las mismas en cuanto a los deseos de los usuarios. A diferencia de los procesos tradicionales, donde el resultado el producto deseado y un conjunto de mediciones y lecciones aprendidas, en este caso además se agrega una base de conocimiento para luego seguir construyendo el producto comenzado.

El modelo de proceso creado fue puesto en práctica con resultados satisfactorios tanto para el equipo de desarrollo, como para el cliente. Si bien el equipo de desarrollo también oficio como cliente, Marcelo Guerra se mostró muy satisfecho con la solución lograda y con la aceptación del producto en la tienda.

Tanto los estudios de mercados realizados como lo relevamientos de metodologías y casos de éxito, sirven como guía para futuros emprendimientos en mercados de aplicaciones virtuales. No solo por el resultado que obtuvimos a partir de ellos, sino como experiencias anteriores.

El proceso se enmarco en el desarrollo de un modelo de proceso para Windows Store, pero el resultante del mismo es un modelo de proceso el cual no se encuentra sesgado solamente a Windows Store. Esto es que cualquieras de las actividades que se indican que se realicen en el marco de Windows Store perfectamente es extrapolable a cualquier mercado y ambiente de desarrollo al que se apunte.

Considerando características de calidad podemos mencionar la obtención de información (por medio de la investigación con expertos) de fechas de claves en este tipo de negocios. Pudiéndose salir al mercado en cualquier fecha pero habiendo fechas que brindan una mayor ventajas que otras.

Como otra medida de calidad obtenida por medio del relevamiento mencionado anteriormente, se obtiene la idea de generar feedback interno en la propia aplicación, para que de esta manera ante posibles malas experiencias de usuarios finales estos aprovechen este medio privado de comunicación con los desarrolladores propio y no quede dicho registro al público en gene

5.3 Trabajo futuro

A continuación se describen algunas consideraciones que serían importantes a nuestro entender contemplar para trabajos futuros con el objetivo de lograr una mejor calidad del proceso y la mejora del producto construido.

Para mejorar el producto construido sugerimos:

- ✓ Realizar mejoras en animaciones y diseño gráfico del producto. Cuantas más se agreguen, mayor impacto va a tener el producto.
- ✓ Agregar publicidad, o una tienda para comprar funcionalidades extras, y así poder generar ganancias con el producto.
- ✓ Generar una campaña de marketing que pueda generar más descargas a la aplicación en el mercado.
- ✓ Ya que la aplicación de Windows 8 se construyó bajo estándares de CSS3, HTML5 y JavaScript, podría exportarse a otros mercados de aplicaciones como el de Android y Apple.

Para lograr una mejor calidad del proceso sugerimos:

- ✓ Poder contar con un experto en diseño gráfico para elaborar la capa de presentación del producto a construir. De esta manera, se perderá menos tiempo en la construcción de esta capa, y se realizará de manera más productiva.
- ✓ Poder ejecutar este proceso con otro producto, para poder medirlo y comparar con la ejecución realizada en este proyecto, incluso poder medirlo contra otro mercado para ver si se adapta fácilmente al mismo.
- ✓ Generar una memoria organizacional, que es de gran ayuda ya que se tiene la información de todos los años, se debe indicar que se tome sólo como referencia para el proceso, pero es de gran ayuda poder contar con la misma.
- ✓ Probar el proceso con más integrantes, así los distintos roles no están muy cargados en las pocas personas del equipo.

6. Referencias

Metodologías de Desarrollo

1. The Lean Startup - Eric Ries - *Publication Date*: 13/9/2011
2. http://agilealliance.org – Ultimo Acceso : 08/07/2014
3. http://scrumalliance.org - Ultimo Acceso : 08/07/2014
4. http://agilemanifesto.org/principles.html - Ultimo Acceso : 08/07/2014
5. http://www.proyectosagiles.org/- Ultimo Acceso : 08/07/2014
6. Essential scrum A practical guide to the most popular agile process - Kenneth S. Rubin - *Publication Date*: 5/8/2012
7. Flexibilidad con Scrum - Juan Palacio *Publication Date*: Octubre, 2008
8. Scrum Manager Gestión de proyectos - Juan Palacio & Claudia Ruata - *Publication Date*: Enero, 2011
9. Scrum y XP desde las Trincheras - Henrik Kniberg : Agosto, 2007
10. Software Configuration Management in Scrum Projects - Andreas Bergström – Octubre, 2003
11. Scrum + Engineering Practices: Experiences of Three Microsoft Teams - Laurie Williams, Gabe Brown, Adam Meltzer & Nachiappan Nagappan. - 2011
12. Reporte de experiencia: utilización de métricas en Scrum para analizar y mejorar la productividad de un equipo - Vanesa Mola, Luis Mariano Bibbo & Leandro Antonelli. - 2012
13. http://www.caminoagil.com/2013/02/visual-story-mapping-aplicado.html - Ultimo Acceso : 08/07/2014
14. http://www.javiergarzas.com/2014/01/estimacion-agil-scrum.html - Ultimo Acceso : 08/07/2014
15. http://www.scrummanager.net/bok/index.php?title=Velocidad,_trabajo_y_tiempo - Ultimo Acceso : 08/07/2014
16. http://xprogramming.com/xpmag/whatisxp/ - Ultimo Acceso : 08/07/2014
17. http://www.extremeprogramming.org/ - Ultimo Acceso : 08/07/2014
18. Extreme Programming and Agile Software Development Methodologies - Lowell Lindstrom & Ron Jeffries - *Publication Date*: 21/06/2014
19. http://normaiso9126.blogspot.com/- Ultimo Acceso : 08/07/2014
20. http://www.cuatrorios.org/index.php?option=com_content&view=article&id=163:norma-iso-9126-para-an%C3%A1lisis-de-software&catid=39:blogsfeeds - Ultimo Acceso : 08/07/2014
21. Quality management systems — Fundamentals and vocabulary
22. Las normas ISO 9000:2000 de Sistemas de Gestión de la Calidad - Leticia Colín O.
23. Calidad en la Industria del Software. La Norma ISO-9126 - María Antonieta Abud Figueroa
24. Krug, Steve - Don't Make Me Think - A Common Sense Approach To Web Usability (Second Edition) - *Publication Date*: 01/09/2005

25. http://centrodeartigos.com/articulos-noticias-consejos/article_138812.html - Ultimo Acceso : 08/07/2014

ANEXO 1 - Análisis de Mercados Paralelos

Con la consolidación de los dispositivos móviles como medios de comunicación y entretenimiento, el desarrollo de las aplicaciones móviles ha evolucionado de manera significativa a lo largo de los últimos años, de tal forma que han surgido tendencias completamente nuevas en términos de usabilidad y plataformas de desarrollo. Por ello, a la hora de desarrollar una nueva aplicación, es necesario obtener un completo panorama del mercado objetivo. Para ello, en las siguientes secciones se realizará un estudio de mercado sobre el mercado de las aplicaciones móviles considerando las estadísticas registradas en: Estados Unidos (uno de los países con mayor influencia en la comercialización de este tipo de aplicaciones).

Dado esto, antes de comenzar el análisis, es importante definir exactamente a que llamamos Aplicaciones Móviles. Para ello vamos a diferenciar correctamente el software por su tipo. Dentro de los dispositivos móviles (Smartphones o Tablets) tenemos:

- ✓ Aplicaciones Nativas (común mente llamadas: Aplicaciones móviles): aplicaciones que son instaladas en los dispositivos y que no requieren de transferencia de datos con un servidor (funcionalidad sin acceso a una red). Los datos son almacenados en el mismo dispositivo.
- ✓ Aplicaciones Web Móviles: Son Aplicaciones que no se encuentran instaladas en los dispositivos lo que las hace accesibles a través del navegador del dispositivo por medio de una dirección URL en la web. En este caso la memoria del dispositivo no es relevante ya que los datos no son almacenados en este, pero si depende de la calidad del navegador ya que toda la información proviene desde el servidor y es reproducida al acceder a la dirección URL.
- ✓ Aplicaciones Cliente – Servidor: También denominadas "Aplicaciones Semi-nativas" son aquellas que pueden ser instaladas en el dispositivo pero que no pueden ser utilizadas en la ausencia de una red, ya que descarga datos desde el servidor para su funcionamiento. Sin una conexión, la aplicación no procederá – es el caso de las aplicaciones comerciales como la Banca en línea, donde se puede visualizar la interface con el usuario, pero toda la información proviene desde un servidor, por ello la memoria del dispositivo es parcialmente dependiente ya que solo es utilizada para la instalación de la aplicación.

Adicionalmente, es necesario identificar las distintas plataformas en las que se basará nuestro análisis. Estas son donde operan las aplicaciones y en general corresponden a sistemas operativos que se diferencian según los dispositivos y la arquitectura de los mismos o el código en el que se encuentran programados. En el siguiente análisis, en general nos referimos a las principales plataformas:

- ✓ iOS

- ✓ Android
- ✓ Windows Mobile
- ✓ Blackberry
- ✓ Symbian OS

Principales características

El mercado de Aplicaciones móviles en los Estados Unidos, se estima que está compuesto por 165 millones de usuarios activos que tiene acceso a dispositivos móviles de última generación. Esto representa un 54% de la población. Según el siguiente grafico podemos observar la distribución etaria por uso de Smartphone o dispositivo convencional. A partir de este, podemos observar una gran concentración de dispositivos en adultos y jóvenes.

Grupo	Convencional	Smartphone
Generación Z (18-23)	95%	64%
Millennials (24-32)	97%	72%
Generación X (33-46)	95%	61%
Boomers Jóvenes (47-56)	92%	39%
Boomers (57-67)	89%	28%
Generación Dorada (68-88)	85%	16%
Total	**93%**	**50%**

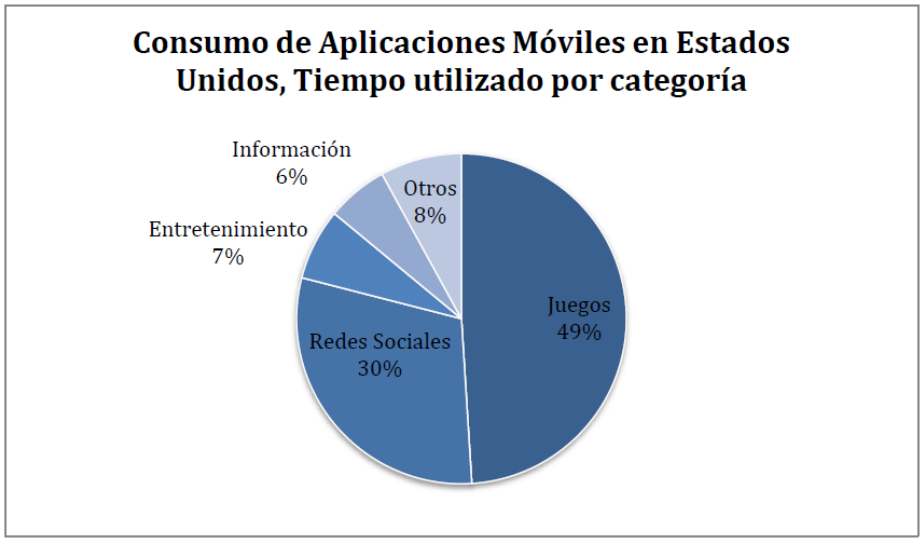
FIGURA 8

En los últimos 5 años, este mercado se ha incrementado de manera sostenida, sumando 15 millones de nuevos usuarios (promedio) cada trimestre. En la actualidad se estima que la tasa de participación de los dispositivos móviles alcanza el 56%, lo cual representa un total 131 millones de dispositivos.

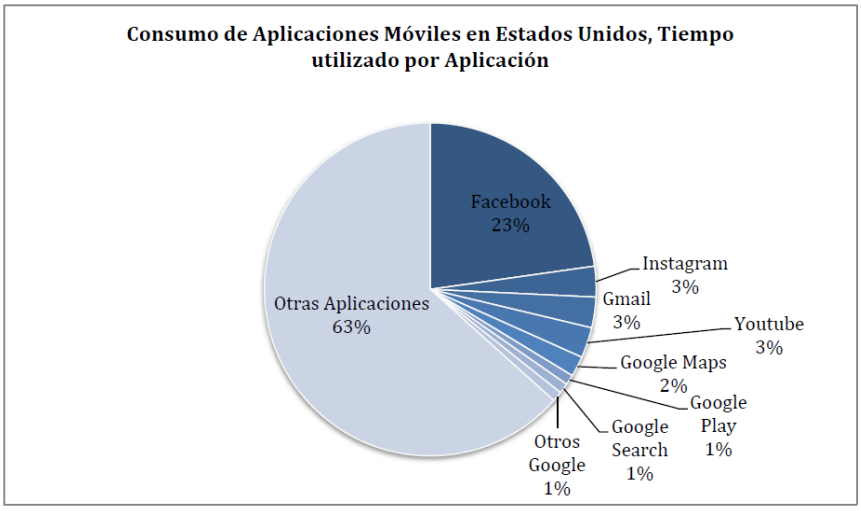
Comportamiento de los usuarios

El número de descargas se ha visto un incremento exponencial e independientemente del tipo de las mismas (aplicaciones pagas o gratuitas) o plataforma. Particularmente los usuarios demuestran mayor interés en el consumo de aplicaciones relacionadas con juegos donde cerca del 49% del tiempo de los usuarios es dedicado a su uso, 30% es utilizado en redes sociales, 7% en aplicaciones relacionadas con entretenimiento, 6% en consumo de información en

diarios y revistas, y el 8% restante es dedicado a otras actividades misceláneas. Se considera que el tiempo de uso promedio diario es de 58 minutos.



Por su parte, el tiempo de uso en redes sociales por usuarios adultos (mayores de 18 años) entre los distintos sistemas operativos se desglosa de manera tal que: el 23% del tiempo es dedicado a Facebook y 3% a Instagram. El uso de aplicaciones ofrecidas por Google indica que los usuarios destinan 3% a Gmail, 3% en Youtube, 3% en Google Maps, 2% en Google Play, 2% en Google Search y 1% en otras aplicaciones del mismo desarrollador. El restante 64% corresponde al tiempo de uso en otras aplicaciones. En general, el tiempo promedio que los usuarios utilizan sus dispositivos móviles es cercano a 58 minutos diarios.



Con respecto a los canales de distribución, las tiendas de aplicaciones en línea que capturan (prácticamente) la totalidad del mercado, corresponden a

Apple App Store y Google Play, ambas orientadas hacia iOS y Android, respectivamente.

Si se considera el tamaño de estas, se puede señalar que Apple lidera en número de aplicaciones disponibles, superando el millón de aplicaciones. Por su parte, Google Play dispone de alrededor de 900.000, mientras que Windows posee 125.000. A la fecha, el canal de Apple registra más de 50 billones de descargas, y Google Play 50 billones. Si bien Apple y Google Play ambas poseen el mismo modelo de negocios, difieren en el costo de membresía que cobran a los desarrolladores; Apple exige un pago anual de US$ 99 dólares y Google Play $25. Ambas participan del 30% de las utilidades por concepto de venta de aplicaciones para cada descarga.

Es importante tener en consideración que Google Play ha alcanzado mayores ratios de crecimiento que Apple App Store, tanto en el número de descargas como en utilidades. Mientras Google Play registró cerca del 90% del total del número de descargas de Apple App Store en el primer trimestre de 2013, la tienda de distribución de Apple mantiene su liderazgo en monetización, generando aproximadamente 2,6 veces los ingresos por venta de Google Play. En comparación a 2012, dicho múltiplo bordeaba las 4 veces.

	Google play		amazon apps
No. of app downloads to date	10 billion	25 billion	31,000
Membership costs to developers	$25	$99	$99
Revenue taken by app store for in app purchases	30%	30%	30%
No. of apps in the app store	450,000+	550,000+	31,000
Revenue comparison per user*	$0.23	$1	$0.89

* Flurry

Estados Unidos es el responsable del mayor crecimiento en las utilidades de ambos canales en el último periodo, aunque China se posicionó como el segundo gran contribuyente en las ventas. Por otra parte, Rusia fue un factor clave en el mayor número de descargas desde Google Play durante 2012.

A continuación se indican las aplicaciones gratuitas de mayor número de descargas por plataforma para Estados Unidos:

- ✓ ***Apple App Store***
  - Badland (Frogmind)
  - Infinity Blade II (Chair Entertainment Group)

- Traktor DJ for iPhone (Native Instruments GmbH)
- How to cook Everything (Culinate, Inc)
- True or False (Games for Friends)

✓ ***Google Play***

- Facebook (Facebook)
- Candy Crush Saga (King.com)
- Pandora Internet Radio (Pandora)
- Despicable Me (Gameloft)
- Instagram (Instragram)

✓ ***Amazon***

- Feat the Boss 2 (Game Hive Corp.)
- Hair Salon – Kids Games (6677g ltd.)
- Can You Escape (Mobigrow)
- Guess Song Quiz (Alfa Production LLC)
- Bubble Witch Saga (King.com)

✓ ***Windows Store***

- Xbox SmartGlass (Microsoft Corporation)
- Face book Now (Two Guys' Wikipedia, Facebook)
- Skype (Skype)
- YouTube Now (Two Guys' Instagram, Facebook)
- Google Search (Google Inc.)

Dinamismo del mercado

Se espera que para los próximos años, un reemplazo y actualización de los dispositivos existentes y no la aparición en masividad de nuevos usuarios, lo cual indica que la industria se encuentra en una etapa de maduración avanzada. A partir de esto, podemos decir que la descarga de las aplicaciones móviles será dependiente de las ventas en las distintas tiendas de distribución y también de los ingresos percibidos por publicidad. Esto no significa que el volumen de mercado sea menor, ya que a medida que la velocidad de transmisión de datos en la redes se incrementa (producto del uso de nuevas tecnologías) los dispositivos existentes quedarán obsoletos en términos de capacidades técnicas y por ello se esperaría un proceso de recambio importante durante los próximos años. Esto produce nuevos requerimientos en las nuevas aplicaciones (conectividad) que se desarrollen que a su vez demandarán componentes de mayor capacidad para su funcionamiento.

En cuanto a las aplicaciones móviles, habrá un incremento en la migración desde las plataformas existentes a las nuevas actualizaciones (impulsado principalmente por iOS y Android). Constantemente se realizarán revisiones y actualizaciones de los principales sistemas operativos buscando compatibilidad con los nuevos dispositivos. Además surgirán nuevas aplicaciones que sustituyen distintos servicios disponibles en otros medios, como: Banca Comercial, Software Corporativo, Software Operacional, Juegos, aplicaciones de productividad, entre otros.

Producto de la incorporación de nuevas tecnologías en los dispositivos móviles, hay una versatilidad de uso de estos en cuanto a las prestaciones ofrecidas por las aplicaciones. Al inicio el foco de atención se fijaba en mensajería instantánea. En la actualidad el espectro de usabilidad se ha expandido a tal punto que los usuarios pueden utilizar sus celulares o tablets para jugar, escuchar música, compartir fotos, etc. Gracias a esto, mientras cambie el foco de interés de los usuarios (y se vuelva cada vez más específico) el número de aplicaciones disponibles en el mercado se incrementa de manera significativa cada año, así también la cantidad de descargas y utilización de distintas plataformas de distribución de software. En el siguiente grafico podemos ver este crecimiento.

Comercialización

La comercialización de las aplicaciones dentro de estos mercados se puede diferenciar por tipos de compras. Estas compras, pueden efectuarse mediante el uso de tarjetas de crédito, gift-cards, cupones, entre otros, gracias a mecanismos implementados dentro de cada tienda, lo que facilitan la comercialización a los desarrolladores.

El primer tipo de compra es: aplicaciones gratuitas. Estas no tienen un cargo asociado al uso por parte del usuario, por lo general son financiadas a través de publicidad. Por otro lado están las de pago. Estas cargan un costo al usuario por su uso, ya sea en términos de un cobro inicial único o bien mediante suscripciones por un período de tiempo determinado. Por último existen las denominadas: Intra-Aplicaciones. Son descargas gestionadas al interior de las aplicaciones también denominadas “add-ons” y en particular, este tipo de descargas se ha vuelto más popular ya que transforman aplicaciones en puntos de venta.

Generalmente las tiendas de aplicaciones cobran un cargo fijo a los desarrolladores por concepto de alojamiento y uso de servidores, si bien dichos ingresos son marginales las mayores utilidades son percibidas a través de su participación en el precio de venta de las aplicaciones.

Demanda

La demanda de una aplicación dentro de un mercado puede ser cuantificada gracias al número de descargas ejecutadas por usuarios únicos en las distintas plataformas. Se toma en consideración el tiempo de utilización del software como un porcentaje del total destinado diariamente. También se contemplan variables demográficas a modo de establecer una diferenciación más precisa en el perfil de cada usuario, su patrón y grado de utilización. Asimismo, el número de dispositivos móviles determina el tamaño de mercado atendible, y su utilización (en tiempo) también corresponde a un indicador.

Como se discutió anteriormente, las tiendas de aplicaciones canalizan la descarga y venta del software, lo cual refleja el impacto de cada aplicación en el mercado. Dependiendo de las capacidades de cada desarrollador y la infraestructura sobre la que opera cada aplicación, es posible extraer información más precisa en cuanto a la utilización que hacen los usuarios del software sólo si existe consentimiento expreso de parte de estos.

Particularmente, en Estados Unidos las aplicaciones híbridas de mayor utilización/demanda corresponden a aquellas basadas en Redes Sociales, como es el caso de Facebook y Twitter. Las Aplicaciones Nativas de mayor uso corresponden a Juegos. Particularmente Candy Crush Saga, y en la categoría Cliente-Servidor es Whatsapp Messenger.

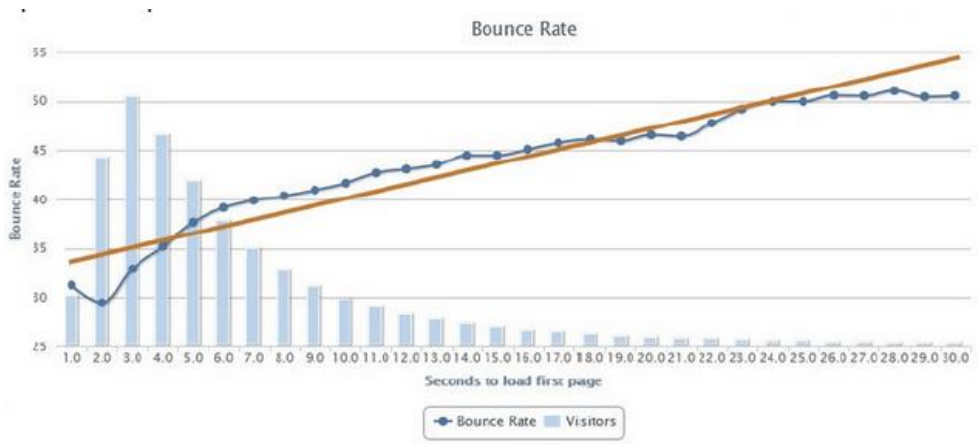
Se puede considerar el número de aplicaciones por categoría que se encuentran disponibles en cada canal, determinando que tipo de usabilidad buscan los usuarios/consumidores finales. En esta categoría ganan los videojuegos.

Buenas prácticas

A continuación se enumeran buenas prácticas para el diseño de aplicaciones móviles. Según expertos, la clave del éxito pasa por diseñar experiencias para el usuario que aprovechen las funcionalidades de los dispositivos para los que son diseñadas (tanto webs móviles o aplicaciones) y aprovechar las sinergias implícitas (como la posibilidad de hacer llamadas).

Uno de los factores que mejora la experiencia de los usuarios móviles es la coherencia de un producto. Mantener los colores corporativos, logos, iconos y otros elementos específicos de la web o publicidad será fundamental para conservar la coherencia. Esto facilita y reduce el tiempo que le lleva al usuario entender una aplicación disminuyendo el público casual.

Optimizar el tiempo de carga de datos de aplicaciones en línea es de vital importancia ya que cada sesión de conexión con su dispositivo móvil es bastante más bajo en comparación a los tiempos medios de otros dispositivos más potentes, por lo que la velocidad de carga es aún más determinante. Los usuarios que esperan demasiado tiempo por un contenido si bien utilizan en una primera vez la aplicación son propensos a que no vuelan.



En el grafico anterior podemos observar que el volumen de descargas de una aplicación que utiliza contenidos online aumenta su tasa de rebote en 0.65 cada segundo, luego de los 4 segundos.

Priorizar la sencillez y practicidad gracias a combinaciones táctiles mejora la interacción con el usuario. Para ello, hay que asegurar que ciertos elementos de la aplicación están bien optimizados y pueden ofrecer una buena experiencia a los usuarios. Entre los errores más frecuentes destacan:

- ✓ Botones no suficientemente grandes
- ✓ Botones, enlaces o campos de rellenar colocados demasiado cerca uno a otro. Esto genera clics erróneos en dichos elementos y frustración en el usuario.

Existen miles de diferentes tipos de dispositivos móviles, por ello es importante llegar a todas las resoluciones de la pantallas. Aunque las diferencias en algunos casos son muy ligeras hay que asegurar que la aplicación aparece con una resolución perfecta para todas las pantallas.

Por último hay que aprovechar las máximo las funcionalidades de los dispositivos, ya que estas puede representar una ventaja competitiva respecto a la competencia. Las cámaras, la geolocalización o mapas interactivos son sólo algunos de los servicios utilizados día a día por los usuarios de los dispositivos móviles.

Referencias

1. Windows 8 User experience guidelines - *Publication Date*: 14/08/2014
2. http://msdn.microsoft.com/es-es/windows/ - Ultimo Acceso : 08/07/2014
3. https://play.google.com/store - Ultimo Acceso : 08/07/2014
4. http://news.softpedia.com/news/Windows-8-Ends-2013-with-142-000-Metro-Apps-Available-for-Download-412989.shtml - Ultimo Acceso : 08/07/2014
5. June 2013 WHAT IS NEEDED FOR TOP POSITIONS IN THE APP STORES? By Hendrik Koekkoek, Analyst at Distimo. – Junio, 2013

6. PMS i+e Estudio de Mercado Aplicaciones Moviles Junio 2013 Documento elaborado por Oficina Comercial de ProChile en New York – Noviembre 2013
7. UK mobile devices usage and demographic roundup An overview of recent research and data on smartphone and tablet ownership in the UK, compiled January 2013. Enero, 2013
8. Libro Blanco Web Mobiles - Mobile marketing association
9. Comportamiento de usuarios en dispositivos móviles - e-interactive
10. Android Market Analysis with Activation Patterns - Peter Teu, Stefan Kraxberger, Clemens Orthacker, G□unther Lackner, Michael Gissing, Alexander Marsalek, Johannes Leibetseder, and Oliver Prevenhueber - 2012
11. http://www.xatakawindows.com/aplicaciones-windows/windows-store-comparado-con-el-resto-de-tiendas-de-aplicaciones-1 - Ultimo Acceso : 08/07/2014
12. http://blogs.msdn.com/b/jennifer/archive/2013/04/25/comparison-of-windows-store-vs-google-play.aspx - Ultimo Acceso : 08/07/2014
13. http://apcmag.com/windows-store-vs-app-store-google-play-the-fight-is-on.htm - Ultimo Acceso : 08/07/2014
14. http://tabtimes.com/resources/the-state-of-the-tablet-market - Ultimo Acceso : 08/07/2014

ANEXO 2 - Análisis del Mercado Objetivo

Introducción

Windows Store es la tienda de comercialización de aplicaciones de Microsoft para sistemas con Windows 8 o Windows Server 2012. Fue creada en noviembre de 2012 y enfoca su mercado a computadoras personales y tablets. Con esta tienda, Microsoft incursiona en el mundo de tiendas virtuales al igual que lo hicieran Google y Apple en 2011.

Si bien el objetivo primario es la comercialización de aplicaciones para tablets y computadoras personales, se espera a futuro proveer contenidos para dispositivos móviles. Al existir Windows Phone, como tienda para dispositivos móviles, Microsoft buscará la unificación de sus dos tiendas virtuales, llegando así a un único producto que provea contenidos para los tres tipos de dispositivos. De esta manera, el usuario no tendrá que cambiar de contexto al cambiar de dispositivo.

Principales diferencias

Windows al centrarse en computadoras personales, en teoría se enfrenta a tiendas similares como Mac App Store de Apple o Ubuntu Software Centre de Ubuntu. El hecho de que Windows Store ocupe la pantalla completa, permite a la aplicación centrar toda la atención del usuario. Por lo demás, la mayoría de las tiendas son muy similares en cuanto a: diseño, las páginas de categorías, las páginas de las aplicaciones o los resultados de búsquedas, las capturas de pantalla o la información de las aplicaciones.

Otro de los puntos claves que ha ganado relevancia con las tiendas de aplicaciones son las opiniones de los usuarios. A diferencia de otras tiendas, donde las opiniones aparecen junto a todas las imágenes y datos de la aplicación, Windows Store opta por mostrarlas en una pestaña aparte. Esto puede tener dos consecuencias: por un lado, dará mayor importancia a las opiniones dotándolas de un espacio propio, pero, por otro, también contribuirá a ocultarlas del primer vistazo con el que los usuarios nos acercamos a la aplicación.

Dinamismo del Mercado

Luego de un año de puesta en funcionamiento, Windows Store ya se encuentra en la conversación con lo que respecta a las tiendas de aplicaciones. Observando cifras de descargas de aplicaciones en el último año, Windows Store ha logrado triplicar las descargas. Un logro muy importante por el poco tiempo de puesta en funcionamiento. Pero no es todo satisfacción cuando se compara lo económico. App Store ha logrado facturar 5 veces más que Windows Store.

Este último fenómeno se logra explicar basado en los números de tipo de aplicaciones disponibles que existen en cada tipo de mercado. Windows Store se ha caracterizado por ser la tienda que ofrece la mayor cantidad de aplicaciones gratuitas en comparación con las pagas, este número haciende al 86% de las aplicaciones disponibles en las tiendas.

Otro aspecto que diferencia al mercado de aplicaciones de Windows 8, es que los tops de las aplicaciones descargas son dominados por aplicaciones que se pueden llamar localizadas. Las aplicaciones localizadas son aquellas que son diseñadas para un determinado país. El 10% de las aplicaciones en el top 300 cumplen este requisito. De hecho, se puede mencionar que cuando los desarrolladores se enfocan a los grandes mercados, ellos se dedican a crear las aplicaciones en idiomas específicos. Por ejemplo, en Japón, el 41% de las tops aplicaciones son locales y el 30% del as de Corea están en ese idioma.

De aquí se puede entrar un dilema por la parte de los desarrolladores, siempre se va a poder tener retribución a los mismas de manera económica por su trabajo realizado, pero dejar las aplicaciones de manera gratuita no un aspecto que genere ganancias a los mismos. Por lo que se plantea una opción para generar las ganancias al comercializar las aplicaciones de manera libre. Microsoft recomienda la venta a terceros de espacios de publicidad en las aplicaciones de las tiendas, por lo que el desarrollador la pública de manera gratuita y si con el curos del tiempo la misma adquiere cierto grado de alcance en el público esta se propensa a poder comercializar de manera más simple espacios de publicidad.

Comportamiento de los usuarios

Analizaremos las aplicaciones más populares de Windows Store, para poder continuar refinando un perfil de usuario de esta tienda. Se realiza una división entre aplicaciones gratuitas y pagas:

- ✓ ***Gratuitas:***
  - Xbox SmartGlass
  - Cliente Skype
  - Netflix
  - Google Search
  - Facebook Touch
  - Kindle
  - Jetpack Joyride
  - YouTube MP3 & Videos downloader
  - Microsoft Solitaire Collection
  - Cut the rope
- ✓ ***Pagas:***
  - Angry Birds Star Wars
  - Draw a Stickman
  - Plex

- Angry Birds Space
- Pin Steam
- Touch File
- Fruit Ninja
- Tweetro+
- YouTube+
- Shark Dash!

En ambas categoría se observan que los tipos de aplicaciones populares son aquellas de entretenimiento, dejando de lado aquellas para la realización de tareas específicas. Entre el entretenimiento encontramos dos formas que polarizan las descargas, aplicaciones sociales (utilizadas para navegar en las distintas redes sociales), y los juegos. Este comportamiento se repite en las diferentes tiendas de aplicaciones, lo que no es propio de Windows Store.

Comercialización

Microsoft como uno de los dominantes en el campo de las computadoras personales, le asegura a los programadores que sus aplicaciones alcancen un mayor público en poco tiempo, basados en dicho lema ha tomado decisiones para que los desarrolladores se animen a incursionar en la tienda y así completar la mismas con una gran cantidad de ofertas. Estas medidas se encuentran en el aspecto económico de la comercialización de las aplicaciones.

Dejar un margen mayor de ganancias es otra de las atracciones de esta tienda, si se compara con las demás tienda, la de Microsoft es la más productiva ya que esta deja hasta un 80% de las ganancias al publicador, comparado con la mayorías de la demás tienda donde el margen de ganancia es de 70%.

La de ofrecer un mercado ya consolidado de PC es otra de las atracciones que propone este gigante de la computación.

Costos

Desarrollar una aplicación no posee costo alguno. Visual Studio Express es la herramienta necesaria y se encuentran a disposición de los desarrolladores de forma gratuita en el sitio Oficial de Microsoft. Una vez obtenida esta, es necesaria una licencia de desarrollador de Visual Studio que se adquiere también de forma gratuita por medio de una cuenta de Microsoft.

Una vez instalado el ambiente de desarrollo, Microsoft deja a disposición, de forma gratuita, una comunidad muy amplia como consulta y guía. Además existen una gran variedad de foros auto mantenidos por los usuarios de la red en los cuales se brindan soluciones en las mismas tecnologías.

Para publicar la aplicación, es necesario registro en la comunidad de Windows Store. La creación de un perfil de publicador tiene un costo anual de U$S 49. Una vez abonados, es posible subir cualquier cantidad de aplicaciones. Las mismas no quedan publicadas inmediatamente ya que deben pasar por un conjunto de pruebas en las cuales se mide la calidad de la aplicación. Con este procedimiento se busca no degradar la calidad de la tienda ni poner en riesgo los dispositivos de los clientes con aplicaciones malignas o con malas experiencias para los usuarios.

Certificaciones

En la actualidad, la seguridad es siempre un factor a tener en cuenta. Windows Store en este tema apunta a ir más en línea con la App Store de Apple. Microsoft realiza un control previo sobre las aplicaciones que se suben a su tienda; todas deben cumplir una serie de condiciones que incluyen seguir la línea marcada por el estilo 'Modern UI'.

Dentro de las condiciones que debe cumplir una aplicación existe un conjunto de pruebas automáticas que debe pasar. Estas se pueden dividir en dos etapas. La primera etapa mide los aspectos de robustez de la aplicación. Ante situaciones de cierres abruptos o mal funcionamiento del sistema verifican la consistencia de datos y liberación de recursos. Una vez pasadas dichas pruebas, en una segunda etapa, un encargado de manera exploratoria verifica que no se generen experiencias desagradables para el usuario final por el uso de la aplicación.

Microsoft deja de manera gratuita un conjunto de pruebas automatizadas similares a las realizadas en la tienda. Con estas el desarrollador podrá verificar su aplicación una vez finalizada, para que al momento de publicar exista una alta probabilidad de que la aplicación sea publicada con éxito.

ANEXO 3 - Relevamiento de Metodologías de Desarrollo

6.1 Entrevista Martín Cabrera

Al comienzo del proyecto, los tres integrantes del grupo nos reunimos con Martin Cabrera un experto en desarrollo de aplicaciones bajo metodologías agiles. Nuestro objetivo fue interiorizarnos más en las metodologías agiles para utilizarlas en la fase de construcción del proceso que estábamos definiendo.

Martin nos comentó, que el proceso de creación de una aplicación desde la visión de un desarrollador, consta de la elección de un producto, que luego se valida con el cliente que quiere el mismo, para luego iterar n veces en la construcción del mismo hasta que el cliente diga que le gusta, lo necesita y pagaría por eso.

Cuando le comentamos de lo que se trataba el proyecto, nos indicó que el proceso en el cual nos podíamos interiorizar era Lean Startup, que se basa, como fue explicado en el marco conceptual, en eliminar practicas inútiles y aumentar las practicas que producen algo valioso durante la fase del desarrollo del producto, para así tener una mejor chance de éxito sin requerir gran financiamiento externo. Con un feedback del cliente durante la fase de creación del producto que se torna fundamental para el proceso, y asegura que el productor, no invierta tiempo diseñando características o servicios que un cliente no quiere. Y la construcción de un MVP (Mínimum viable product) que no esté completamente finalizado, y se permita obtener feedback temprano de los clientes, y así poder identificar las necesidades específicas de los clientes con una inversión de tiempo y dinero mucha menor. Y luego poder incorporar y mejorar el producto con características solicitadas por los propios clientes.

Luego al consultarle como era el proceso de desarrollo en la empresa que lidera, nos contó que era un poco distinto a lo que apuntamos, ya que no eran ellos mismos los que definían que producto iban a desarrollar, sino que existía un cliente, que quiere o precisa cierta aplicación. También nos dijo que el proceso que siguen, es una unión de las mejores características de otros proceso agiles.

El proceso de construcción que utilizaban consistía en lo siguiente: primero se realizaba una reunión con el cliente, a partir de allí en una primera etapa se define la visión y los objetivos del producto a desarrollar. Esto es, definir el público objetivo, a que apuntar a la hora de crear la aplicación, etc.

Luego de finalizada esta primera parte, existe una etapa en donde se tienen que definir las historias a desarrollar, donde se validarán con el cliente. En Scrum, que es la metodología base de construcción del producto en la que se basan, cuenta con un Backlog donde se encuentran las historias que definen el producto, y que

luego integran los distintos Sprints de desarrollo. El problema es que este Backlog no tiene orden entre las distintas historias.

Entonces, con el objetivo armar el Backlog de Scrum pero con una visión del producto, definieron una etapa llamada Story mapping, que consiste en un artefacto, en donde se arma un flujo de actividades de un usuario o grupo de usuarios, y se describe el flujo funcional de actividades que realiza el un usuario a muy alto nivel. La idea es pensar en todas las posibles actividades que pueda realizar un usuario, y armar un árbol de actividades, indicando precedencia entre las mismas. La idea en este momento es no descartar ninguna actividad, por más descabellada que parezca, ya que más tarde se va a analizar la factibilidad de construcción de esa actividad.

Una vez obtenido el grafo de actividades de los usuarios, se baja un nivel de abstracción para cada una de las actividades, y se definen las distintas historias que componen la actividad. Y luego para cada una de las historias se puede definir las tareas asociadas a las historias.

Al final de todas estas definiciones, se tiene un Backlog armado pero con una estructura funcional, a diferencia de lo que se hace normalmente en Scrum, donde se tiene el Backlog pero sin relación ninguna entre las historias. Y en este momento se charla con el cliente para definir que historias son las que tendrían que integrar el mvp, y se ordenan esas historias por prioridad.

Luego, se estiman las historias del mvp, y en base a la velocidad del equipo de desarrollo, se definen los distintos Sprints de Scrum necesarios para lograr finalizar el desarrollo. Nos comentó, que si durante un Sprint, se finaliza el desarrollo de las historias de dicho Sprint, se continúa con la primera del siguiente Sprint.

Nos indicó que previo a iniciar el desarrollo mediante Scrum, se define un Sprint 0 que se compone de la definición del Team de desarrollo, el Agree de definiciones de acuerdos entre el equipo, los wiframe y ux (diseño de interacción de pantalla y flujo de información), la infraestructura, la arquitectura de la aplicación, los criterios de revisión (calidad, trabajar de a pares, test automatizados, etc.), métricas, etc.

Luego si comienza la fase de desarrollo del producto, en la que se itera cada Sprints de Scrum, y cuando este finaliza, típicamente el Scrum Master genera un reporte, indicando las métricas, cuantas horas consumió las tareas, cuantas se hicieron, cuantas quedaron afuera, cuántos puntos se hicieron. Una vez obtenido el reporte, se realiza una reunión de retrospectiva, donde se juntan el equipo de desarrollo, el cliente (product owner), y el Scrum master, y se exponen las cosas que estuvieron bien y mal, se genera un plan de acción de qué cosas se comprometen a hacer para una mejora en el siguiente Sprint. Después de la reunión de retrospectiva, se seleccionan que historias van para el siguiente Sprint, y comienza el nuevo Sprint. Y así se itera hasta llegar al mvp.

6.2 Entrevista Gabriel López

Gabriel López, luego de varios años de trabajo dependiente bajo una misma tecnología, junto a un par de colegas decidió formar una empresa propia. Dicha empresa tenía como objetivo desarrollar aplicaciones para las tiendas de Android y Apple.

Básicamente el negocio en dichas tiendas, consistía en subir aplicaciones y a partir de esta los clientes podrían adquirirlas por menos de diez dólares. De dicho monto un porcentaje iba hacia los desarrolladores mientras que el resto a la empresa del dispositivo móvil correspondiente. Este era uno de los principales puntos de contacto o similitud con nuestra situación.

Otro de los puntos en común con el entrevistado es que el decidía junto a su equipo qué era lo que se desarrollaba y bajo qué condiciones, es decir eran sus propios clientes. Además tanto el mercado de Apple como Android, como el de Windows Store son bastante similares por los dispositivos utilizados ya que los tres apuntan a los mimos clientes.

Una vez formado el grupo con sus colegas, comenzaron a desarrollar una aplicación de conversión de unidades con el fin de tener una primera aproximación a las tecnologías y a la metodología de trabajo. Además de adaptarse al nuevo mercado al que apuntaban. Luego de varios ajustes y varios proyecto y prototipos "Hola mundo", llegaron a lanzar el producto al mercado con baja aceptación. A partir de esto, se dieron cuenta que el éxito no iba de la mano de la calidad del producto o de la complejidad o cantidad de funcionalidades del mismo. Al poseer un competidor directo con el mismo producto con ciento de miles de descargas, se dieron cuenta que las mejoras que hacían diferencia con dicho competidor no hacían suficiente peso como para inclinar la balanza favorablemente. No podían competir con la fama ya generada por su competidor. De allí obtuvieron como experiencia que no se busca calidad en las soluciones generada sino que se busca innovación y salir al mercado lo antes posible. Buscar que el cliente encuentre algo innovador o sencillo. Por más que su producto fuera más completo y claro, eso no basta para hacer la diferencia.

A partir de allí, decidieron refinar el procedimiento de desarrollo. Se buscó brindar servicios en sus respectivos empleos dependientes en busca de capital para el desarrollo en EE.UU. Esto era un importante punto a tener en cuenta ya que la en la primera aproximación al mercado se dieron cuenta que los productos que poseían mayor éxito eran los juegos. Además veían que los desarrolladores buscaban inversión externa para financiar sus aplicaciones.

Otro punto a tener en cuenta fue la salida al mercado del producto, el procedimiento de implementación no debía durar más de 3 o 4 meses ya que en caso de no ser exitoso el producto, se perdía la única oportunidad que brinda la tienda para aparecer en la primera plana (los productos nuevos). Esto los llevo a

tomar la decisión de sacar al mercado un producto a cada tanto con el fin de poder ver el éxito del mismo. En caso de ser exitoso se continuaba desarrollando y creciendo. Lo que tenía de bueno este mecanismo era que no se apostaba de una a un producto sino que se iban aproximando a la logia del mercado.

Como contraparte de esta decisión, notaron que la cantidad de descargas del producto aumenta en los primeros días pero luego es incierto. Puede que aumente exponencialmente una vez consolidado el producto o que decrezca rápidamente sin llegar a cubrir sus costos de mantenimiento.

Luego de desarrollar el primer "Start Up" desarrollaron cuatro productos más donde el más exitoso y el que los ayudo a despegarse de los demás competidores fue un cliente de SharePoint. Básicamente habían buscado en la creación de juegos, sin tener suerte ya que las grandes empresas invertían mucho dinero en animaciones y gráficos, cosa que ellos no tenían alcance. Por más que habían pagado a un diseñador para realizar la interface gráfica de su segundo juego, no podían competir con las grandes empresas.

SharePoint era una de las herramientas más conocidas y utilizadas por los desarrolladores. Al conocer dicha herramienta, vieron que el éxito no solo fue en la innovación (ya que no había un producto con similares características en la tienda), sino en el desarrollo de la misma. Salió rápidamente al mercado utilizando metodologías agiles con sprints muy agresivos.

No solo la salida rápida al mercado era la base del éxito, sino que el producto no tuviera errores catastróficos. Encontraron en las devoluciones de los clientes en las primeras versiones de los productos anteriores muy ricos comentarios. Pensaban que era mejor que el cliente se descargara en caso de no colmar sus expectativas vía mails a ellos que con una mala calificación en la tienda. De esta manera, ellos podían saber cuál era el origen del fracaso o no éxito.

En futuras salidas a mercado vieron que la calidad no era lo más importante, sino que impactar en una primera instancia con el producto y seguir contrayéndolo junto a las devoluciones de los clientes. Pero no solo utilizar a los clientes como verificadores del producto, sino como "clientes" propiamente dicho. Implementar o considerar las mejoras sugeridas, cuidarlos con funcionalidades especiales, etc. Vieron que comparado con el principio del proceso, el producto se iba construyendo y adaptando a las necesidades y sugerencias con los clientes. Se construía en entre ambos donde los requerimientos eran muy cambiantes y la calidad del producto no era lo más importante. Sino que la innovación, la salida temprana al mercado, el valor de las funcionalidades agregadas, etc.

La creación de varias aplicaciones los ayudo además a comprender el tipo de cliente que tiene cada uno de los mercados y a partir de este ver cómo es posible atraparlo. Por un lado vieron que los clientes de Apple son personas que se sienten orgullosas de comprar aplicaciones y pagarlas. De alardear los juegos que descargaron, de manifestar y mostrar su Tablet o teléfono luciéndose. Totalmente

opuesto era la mayoría de los clientes de Android. Estos tenían un perfil muy parecido a un "Hacker". En la comunidad o entorno de los mismos, pagar por un juego o aplicación era de tontos, la principal idea no era lucirse o esta orgullosos con los productos que habían comprado sino con los productos que habían obtenido gratuitamente. Gabriel comentaba que es un mercado un poco más difícil de acceder, no solo por eso, sino por la cantidad de aplicaciones que había en este.

También hay que considerar cuando se sale al mercado, no solo con qué y qué hay parecido en la tienda. Notaron que la mayor cantidad de compras se daban en Navidad, Reyes o en la salida de algún nuevo dispositivo al mercado o cuando hay rebajas en las tiendas de Estados Unidos. No aseguraba el éxito sacar la aplicación en estas fechas, peor dicha iba a ser vista por un montón de gente ya que una vez subida a la tienda la primera versión, esta permanece en la página principal en exhibición durante unos días. Es importante considerar ese primer empuje.

El precio y la venta o presentación del producto es un importante factor. En un mercado como el de Android con ochocientas mil aplicaciones cualquier combinación de palabras claves en la corta descripción junto a la foto de la aplicación es muy importante. Estos factores son muy críticos ya que es muy difícil captar clientes por estos medios al ser imposible encontrar una combinación de palabras que describan al producto y no estén usadas pada describir otro similar y con más descargas. Si la aplicación no tiene una descripción clave, un buen ranking y un precio acorde no entra en el resultado de ninguna búsqueda. Al haber una cantidad impresionante de productos, a la hora de filtrar y listar salen los primeros 300.

Como primera aproximación al estudio del mercado que nos depara nos pareció muy rica la charla con Gabriel. No pusimos al tanto de un montón de situaciones que tenemos que considerar a la hora de salir a un mercado con tantas similitudes.

Por otro lado no sirvió para definir el modelo de procesos a seguir y la agenda del proyecto. Acordamos realizar una siguiente entrevista con un colega del entrevistado que desarrolla para ambos mercados.

ANEXO 4 - Proceso de certificación de una aplicación por parte de Microsoft.

Durante el proceso de certificación, la aplicación atraviesa varios pasos.

- ✓ **Carga:** La aplicación empieza el proceso de certificación cuando la cargas a la Tienda. Durante el proceso de carga, comprobamos el cumplimiento técnico de los paquetes de la aplicación según nuestros requisitos del paquete de la aplicación. Si la aplicación supera estas pruebas, verás un mensaje de carga completada en la página Paquetes. De lo contrario, verás un mensaje de error. Consulta Resolver errores de carga de paquetes para obtener ayuda en la corrección de problemas de carga.
- ✓ **Pre procesamiento:** Después de cargar los paquetes de la aplicación y de enviar la aplicación para su certificación, los paquetes se ponen en la cola de las pruebas automatizadas.
- ✓ **Pruebas de seguridad:** La primera prueba comprueba si hay virus y malware en los paquetes de la aplicación. Si la aplicación no pasa esta prueba, necesitarás comprobar el sistema de desarrollo ejecutando el software antivirus más reciente y luego recompilar el paquete de la aplicación en un sistema limpio.
- ✓ **Pruebas de cumplimiento técnico:** El cumplimiento de los requisitos técnicos se prueba con el Kit para la certificación de aplicaciones en Windows. (Debes asegurarte de probar tu aplicación con el Kit para la certificación de aplicaciones en Windows antes de enviarla a la Tienda).
- ✓ **Cumplimiento de contenido:** Nuestros evaluadores de certificación instalan y revisan tu aplicación para determinar si cumple con los requisitos de contenido. El tiempo que lleva esto depende según la complejidad de la aplicación, la cantidad de contenido visual que presente y la cantidad de aplicaciones que se hayan enviado recientemente. Asegúrate de proporcionar toda la información que los evaluadores deben conocer en la página Nota para los evaluadores.
- ✓ **Informe de certificación:** Después de que se complete el proceso de certificación, obtendrás un informe donde se indicará si tu aplicación pasó o no la certificación. Si la aplicación no pasó la certificación, el informe indicará qué pruebas no se superaron o qué requisitos de certificación no se cumplieron. Consulta Resolver errores de certificación para obtener ayuda y corregir los problemas, de modo tal de poder enviar nuevamente la aplicación para su certificación. Si el problema se debió a que tu aplicación se bloqueó o dejó de responder durante la certificación, recibirás datos de informe de bloqueo para ayudarte a identificar y resolver el inconveniente.
- ✓ **Lanzamiento:** Cuando tu aplicación supera la certificación, pasa inmediatamente al proceso de registro y publicación, a menos que hayas especificado que no debe lanzarse hasta una determinada fecha.

- ✓ **Registro y publicación:** Los paquetes de la aplicación se firman digitalmente para protegerlos de alteraciones después de su lanzamiento. Una vez iniciada esta fase, ya no puedes cancelar el envío ni cambiar la fecha de lanzamiento.

Una vez superados los pasos anteriores, la aplicación estará disponible en la Tienda Windows para que los clientes descarguen y podrás agregar vínculos a la página de descripción de tu aplicación. Ten en cuenta que llevará un poco más de tiempo antes de que la descripción de la aplicación aparezca en los resultados de búsqueda.

ANEXO 5 – Agenda de Desarrollo

La ejecución del proyecto se dividió en tres partes: relevamiento del trabajo y formulación de propuesta, ejecución de la propuesta y documentación y cierre de proyecto. En la siguiente tabla se encuentra detallada la agenda del proyecto.

Fases	Fecha	Actividades Realizadas
Inicial de estudio análisis y evaluación de los procesos de software	01/04/2013 al 28/4/20013	Estudio de procesos de ingeniería de software.
		Reuniones con tutor y cliente.
		Estudio de mercado de Windows 8 y paralelos.
		Estudio de Metodología ágiles.
		Charlas con expertos de metodologías agiles y mercados móviles.
Definición del proceso	29/04/2013 al 26/5/20013	Reunión con tutor y cliente.
		Reunión con experto en metodologías ágiles.
		Definición y validación del proceso.
Ejecución del proceso	27/5/2013 al 23/12/2013	Ejecución del proceso y construcción del producto.
Etapa 1 – Validación de ideas	27/05/2013 al 16/06/2013	Análisis y estudio de producto.
		Demos y estudio de tecnologías.
		Reunión con tutor y cliente, y validación del producto a construir.
Etapa 2 – Elaboración	17/06/2013 al 07/07/2013	Creación de historias del producto, con artefacto story map.
		Ejecución de sprint 0.
		Charla con experto en creación de video juegos en mercados móviles.
Etapa 3 - Construcción	08/07/2013 al 29/09/2013	Construcción del producto.
	30/07/2013 al 20/10/2013	Testing de performance, feedback con amigos antes del lanzamiento a producción y estabilización del producto.
Etapa 4 – Liberación	21/10/2013 al 17/11/2013	Puesta en producción en la tienda y conclusiones del proceso.
	21/10/2013 al 03/11/2013	Puesta en producción de la aplicación.
	04/11/2013 al 17/11/2013	Conclusiones del proceso y feedback
		Mantenimiento evolutivo de la aplicación.

Etapa 5 – Segunda Ejecución	18/11/2013 al 23/12/2013	Desarrollo de las mejoras en producto.
Escritura documento	13/1/2014 al 28/4/2014	