**8.1 Modelo de Ciclo de Vida**

El modelo de ciclo de vida es el conjunto de actividades, las cuales conducen al desarrollo del producto de software de manera exitosa. Atlantis-Software tomo como referencia 3 modelos del desarrollo de software para la definición del modelo de ciclo de vida del grupo de trabajo y de esta manera desarrollar el proyecto para el curso de Ingeniería de Software.

**Desarrollo Iterativo e Incremental:**

Este modelo de ciclo de vida consiste en separar el proyecto en una serie de iteraciones**,** estas iteraciones eligen requerimientos del proyecto según la priorización es decir, los requerimientos más importantes se desarrollan al principio del proyecto y a medida que se desarrolla el proyecto se va implementando todos los requerimientos según la priorización, después de cada iteración se hace una retroalimentación.

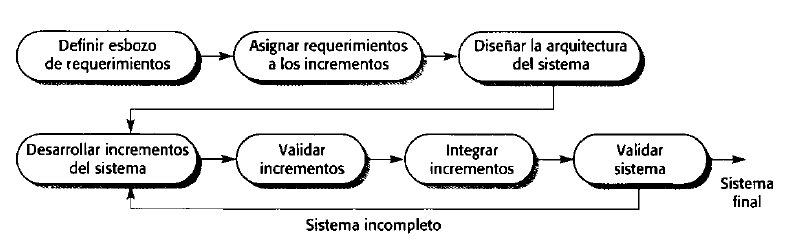


Figura 8.1.1 Desarrollo Iterativo e Incremental Tomada de [1]

En la figura 8.1.1 se puede ver como el desarrollo iterativo e incremental integra los distintos iteraciones desarrollados, es decir, las iteraciones que ya se han desarrollado dejan como resultado una parte implementada del producto a entregar al cliente el cual se va refinando y completando a medida que se va avanzando en el proyecto, las iteraciones para este proyecto se tomaran entre 1-3 semanas, en donde el grupo tendrá una serie de requerimientos a desarrollar dentro del tiempo estipulado.

**Scrum:**

Scrum es un método de desarrollo de software ágil el cual se basa en el modelo iterativo y a partir de esto genera diferentes estrategias de trabajo. Las estrategias que se utilizaran para este proyecto son:

* Sprint Backlog: es un documento en el cual se indica las tareas a desarrollar en un Sprint, dentro del marco de este proyecto se utilizara para definir las áreas a realizar dentro de cada iteración.
* Product Backlog: es un documento el cual tiene la priorización de las necesidades del cliente, en este proyecto se utilizara este documento para saber cuál es la prioridad más alta dentro de cada iteración.
* Burndown Chart: es un documento que tiene diferentes medidas para ver el esfuerzo del grupo de desarrollo y por medio de esto poder medir en forma general como está trabajando el equipo de desarrollo. Para este proyecto se utilizara este documento para poder ver el avance o retraso que se tiene dentro del proyecto.
* Reuniones Diarias: Es una práctica la cual consiste en verificar el avance del sprint sin tener la necesidad de esperar al final del sprint sino abarcar las dificultades que se presenten en el sprint de manera inmediata en reuniones cortas de quince minutos aproximadamente. Para este proyecto por la disponibilidad de tiempo no se podrá hacer reuniones diarias pero se hará reuniones después de la clase Ingeniería de Software para poder abarcar las dificultades con respecto a la iteración en desarrollo.
* Revisión del Sprint: es una práctica para verificar el incremento y actualizar el Product Backlog si es necesario. La idea es comunicar las distintas experiencias que se vivieron durante el Sprint para próximos incrementos y poder tomar medidas para mejorar como grupo de desarrollo. Para este proyecto se hará revisión de cada iteración para saber qué problemas se han tenido con las tareas asignadas y también conjunto con el cliente se harán distintas revisiones para tener una retroalimentación con respecto a las tareas que se estén desarrollando.
* Reunión de Planeación del Sprint: Es una reunión en donde según la priorización de los requerimientos y las habilidades de cada una de las personas del grupo de trabajo se asigna tareas para implementar dentro del Sprint. Para este proyecto, en las reuniones semanales se planeara las tareas a realizar durante la iteración.

**Extreme Programming:**

Es un método de desarrollo de software ágil el cual su objetivo es tener todo el tiempo al cliente, para saber bien cuáles son las necesidades y los requerimientos para su proyecto. Esta metodología tiene prácticas de las cuales en este proyecto se han seleccionado:

* Programación en parejas: Todo el código es desarrollado en parejas que trabajan en una sola máquina, colaborándose en el mismo diseño, algoritmo, código, prueba, etc.
* Revisiones cruzadas: Se hacen revisiones según unos criterios definidos por el grupo de trabajo.

**Definición Fases de Atlantis-Software:**

Tomando en cuenta el desarrollo incremental iterativo, se definió las siguientes fases: (llamar las fases por el nombre que aparezcan en el ciclo de vida incremental iterativo

**Inicio de Proyecto**: se define el nombre del grupo de trabajo, las reuniones de trabajo, parejas de trabajo, roles del grupo, reglas de trabajo, definición del producto a desarrollar.

**Planeación y Análisis**: Definir esbozo de requerimientos, asignar requerimientos a los incrementos, análisis de requerimientos, priorización de requerimientos, tareas por iteración, asignación de tareas.

**Implementación**: desarrollar incrementos del sistema, investigación.

**Retroalimentación**: validar incrementos, revisiones cruzadas, pruebas, análisis de riesgos.

**Cierre de proyecto**: integraciones finales, documentación programa, entrega formal al cliente.

En la figura 8.1.2 se puede ver como se implementará el ciclo de vida en cada una de las fases, También se puede ver los diferentes documentos que se realizaran con respecto a cada actividad para poder asegurar las buenas prácticas para el proyecto y tener un soporte físico del trabajo que se esta realizando

.

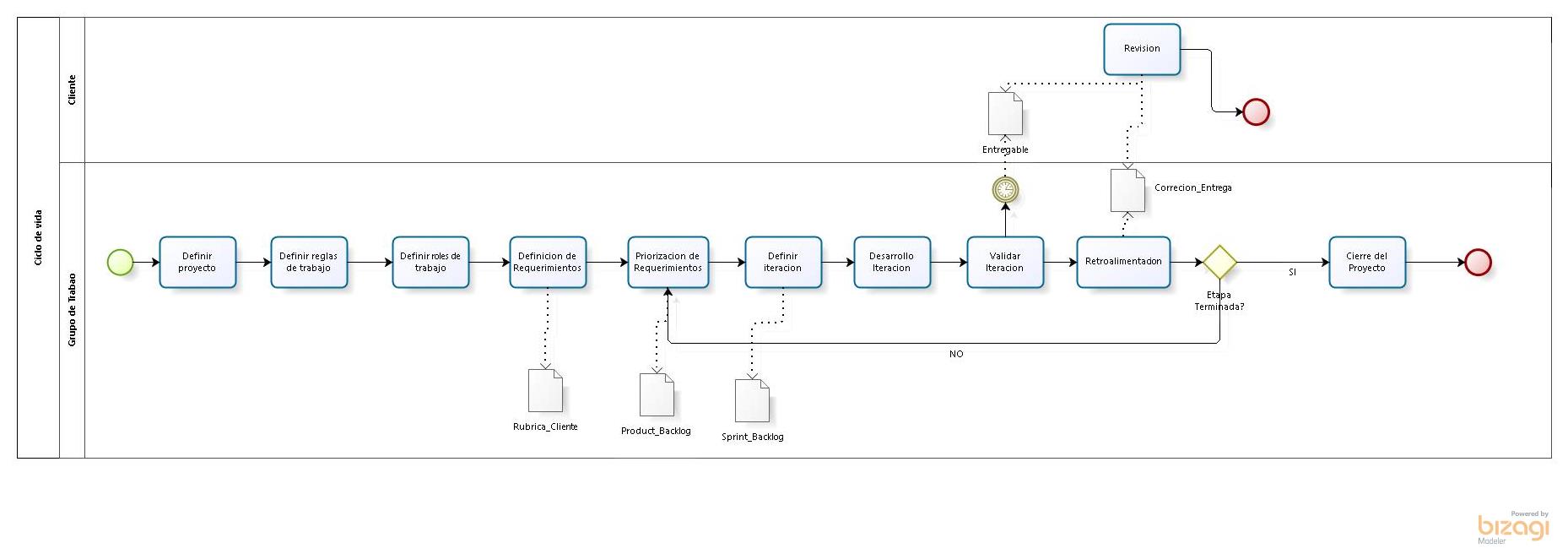


Figura 8.1.2 Descripción BPMN de alto nivel del proceso de Atlantis-Software

**8.1.1 Análisis de Alternativas y Justificación**

Los anteriores modelos y buenas prácticas fueron elegidos por Atlantis-Software porque se realizó un estudio y una investigación con respecto a otros ciclos de vida, dentro de los cuales está el modelo en espiral, el cual tiene un enfoque importante en el análisis de riesgos y es similar al iterativo incremental, pero se decidió escoger el iterativo incremental ya que en el iterativo incremental lo que importa es la parte de la priorización de requerimientos y siempre después de cada incremento se valida y actualiza la lista de requerimientos del proyecto. Por otra parte, los modelos de ciclo de vida secuencial como Cascada, Diente de Sierra y Diente de Tiburón también se descartaron ya que estos modelos de ciclo de vida no tienen una retroalimentación continua y no se puede pasar a fases futuras sin haber terminado las fases previas.

Por el otro lado, los métodos agiles no se pueden implementar de manera completa ya que aunque la parte de desarrollo es muy importante, el enfoque de este proyecto es la parte de la documentación y el proceso de cómo se realizará conjunto con su planeación, por esta razón las referencias de Scrum y Extreme Programming son más enfocado a las buenas practicas dentro del grupo de trabajo que la parte de gestionar nuestro modelo de ciclo de vida.

# Bibliography

|  |  |
| --- | --- |
| [1] | I. Sommerville, in *Ingenieria del Software*, Pearson , 2005, p. 67. |
| [2] | U. o. Missouri, "Extreme Programming," [Online]. Available: http://www.umsl.edu/~sauterv/analysis/f06Papers/Hutagalung/#xp. [Accessed 14 8 2014]. |
| [3] | U. o. M.-C. Science, "Agile Methods," [Online]. Available: http://www.cs.umd.edu/~mvz/cmsc435-s09/pdf/slides7.pdf. [Accessed 14 8 2014]. |
| [4] | "Scrum Guide," [Online]. Available: https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf#zoom=100. [Accessed 24 8 2014]. |