## ESQUEMA DEL CICLO DE VIDA.

**Ciclo de vida de Prototipos**

Cuando hablamos del modelo como tal, que es la creación de un prototipo no necesariamente es crear o modificar algo que ya previamente se ha hecho, sino hacer énfasis en la necesidad del cliente y en la perspectiva que se tiene sobre el modelo como tal (Ingenieria del Software , 2006).

El ciclo de vida tiene varios procesos por los cuales pasa un prototipo que conllevan a una mayor simplificación y entendimiento por parte del cliente, esto nos permitir poder regresar a algunos de los procesos después de haberlo puesto en funcionamiento para realizar los diferentes cambios que necesita, y así poder llegar a un producto final que contenga todas las mejoras para lograr que esta se confiable (Ingenieria del Software , 2006).

Teniendo en cuenta que es un prototipo y su función, podemos definir que un ciclo de vida por prototipos es el tiempo en que transcurre desde que se inicia el prototipo hasta su desarrollo fina (Ingenieria del Software , 2006)l.

Ventajas (martha, 2008):

* Permite la construcción del sistema con requisitos poco claros o cambiantes.
* El cliente recibe una versión del sistema en muy poco tiempo, por lo que lo puede evaluar, probar e, incluso, empezar a utilizarlo.
* Se pueden introducir cambios en las funcionalidades del sistema en cualquier momento.
* Involucra al usuario en la evaluación de la interfaz de usuario.
* Se reduce el riesgo y la incertidumbre sobre el desarrollo.
* Genera signos visibles de progreso, que se utilizan cuando existe una demanda en la velocidad del desarrollo.
* Permite entender bien el problema antes de la implementación final.

Desventajas (martha, 2008):

* El cliente puede quedar convencido con las primeras versiones y, quizás, no vea la necesidad de completar el sistema o rediseñarlo con la calidad necesaria.
* Requiere trabajo del cliente para evaluar los distintos prototipos y traducirlo en nuevos requisitos.
* Requiere un tiempo adicional para definir adecuadamente el sistema.
* No se sabe exactamente cuánto será el tiempo de desarrollo ni cuantos prototipos se tienen que desarrollar.
* Si un prototipo fracasa, el coste del proyecto puede resultar muy caro.

**Ciclo de vida en cascada**

El ciclo de vida inicialmente propuesto por Royce en 1970, fue adaptado para el software a partir de ciclos de vida de otras ramas de la ingeniería. Es el primero de los propuestos y el más ampliamente seguido por las organizaciones (se estima que el 90% de los sistemas han sido desarrollados así) (INGENIERIA DE SOFTWARE, 2005).

Este modelo admite la posibilidad de hacer iteraciones, es decir, durante las modificaciones que se hacen en el mantenimiento se puede ver por ejemplo la necesidad de cambiar algo en el diseño, lo cual significa que se harán los cambios necesarios en la codificación y se tendrán que realizar de nuevo las pruebas, es decir, si se tiene que volver a una de las etapas anteriores al mantenimiento hay que recorrer de nuevo el resto de las etapas (INGENIERIA DE SOFTWARE, 2005).

Ventajas (martha, 2008):

* Es un modelo sencillo y disciplinado.
* Es fácil aprender a utilizarlo y comprender su funcionamiento.
* Está dirigido por los tipos de documentos y resultados que deben obtenerse al final de cada etapa.
* Ha sido muy usado y, por tanto, está ampliamente contrastado.
* Ayuda a detectar errores en las primeras etapas a bajo costo.
* Ayuda a minimizar los gastos de planificación, pues se realiza sin problemas.

Desventajas (martha, 2008):

* Los proyectos raramente siguen el proceso lineal tal como se definía originalmente el ciclo de vida.
* Es difícil que el cliente exponga explícitamente todos los requisitos al principio.
* El cliente debe tener paciencia pues obtendrá el producto al final del ciclo de vida.
* No refleja exactamente cómo se programa realmente el sistema, en el que suele haber un gran componente iterativo.
* Puede resultar complicado regresar a etapas anteriores (ya acabadas) para realizar correcciones.
* El producto final obtenido puede que no refleje todos los requisitos del usuario.

**Ciclo de vida en "v"**

El modelo en V es una variación del modelo en cascada que muestra cómo se relacionan las actividades de prueba con el análisis y el diseño. La codificación forma el vértice de la V, con el análisis y el diseño a la izquierda y las pruebas y el mantenimiento a la derecha (INGENIERIA DE SOFTWARE, 2005).

La unión mediante líneas discontinuas entre las fases de la parte izquierda y las pruebas de la derecha representa una doble información. Por un lado, sirve para indicar en qué fase de desarrollo se deben definir las pruebas correspondientes. Por otro sirve para saber a qué fase de desarrollo hay que volver si se encuentran fallos en las pruebas correspondientes (INGENIERIA DE SOFTWARE, 2005).

Por lo tanto, el modelo en V hace más explícita parte de las iteraciones y repeticiones de trabajo que están ocultas en el modelo en cascada. Mientras el foco del modelo en cascada se sitúa en los documentos y productos desarrollados, el modelo en V se centra en las actividades y la corrección (INGENIERIA DE SOFTWARE, 2005).

Ventajas (INGENIERIA DE SOFTWARE, 2005):

* La relación entre las etapas de desarrollo y los distintos tipos de

pruebas facilitan la localización de fallos.

* Es un modelo sencillo y de fácil aprendizaje.
* Hace explícito parte de la iteración y trabajo que hay que

revisar.

* Especifica bien los roles de los distintos tipos de pruebas a

realizar.

* Involucra al usuario en las pruebas.

Desventajas (INGENIERIA DE SOFTWARE, 2005)**:**

* Es difícil que el cliente exponga explícitamente todos los

requisitos.

* El cliente debe tener paciencia pues obtendrá el producto al

final del ciclo de vida.

* Las pruebas pueden ser caras y, a veces, no lo suficientemente

efectivas.

* El producto final obtenido puede que no refleje todos los

requisitos del usuario.

Objetivos del ciclo de vida en V

Minimización de los riesgos del proyecto

Mejora la transparencia del proyecto y control del proyecto, especificando los enfoques estandarizados, describe los resultados correspondientes y funciones de responsabilidad. Permite una detección temprana de las desviaciones y los riesgos y mejora la gestión de procesos, reduciendo así los riesgos del proyecto.

Mejora y Garantía de Calidad

Como un modelo de proceso estándar, asegura que los resultados que se proporcionan sean completos y contengan la calidad deseada. Los resultados provisionales definidos se pueden comprobar en una fase temprana. La uniformidad en el contenido del producto mejora la legibilidad, comprensibilidad y verificabilidad.

Reducción de los gastos totales

El esfuerzo para el desarrollo, producción, operación y mantenimiento de un sistema puede ser calculado, estimado y control de manera transparente mediante la aplicación de un modelo de procesos estandarizados. Reduciendo la dependencia en los proveedores y el esfuerzo para las siguientes actividades y proyectos.

Mejora de la comunicación entre todos los inversionistas

La descripción estandarizada y uniforme de todos los elementos pertinentes y términos es la base para la comprensión mutua entre todos los inversionistas. De este modo, se reduce la pérdida por fricción entre el usuario, comprador, proveedor y desarrollador.

**Ciclo de vida a implementar**

El ciclo de vida o modelo de vida de software que se implementará será el ciclo de vida en V debido al ritmo de trabajo que se tiene en el equipo de trabajo, además de ser el mas indicado para proyectos de software con poco tiempo para la realización de estos, otro de los factores que influyen para tomarlo como ciclo de vida a utilizar es que es un modelo muy sencillo de utilizar y de fácil aprendizaje.

Otro aspecto importante del ciclo de vida en V es que existe una relación entre las etapas con las pruebas y cada prueba valida un apartado de verificación esto hace que la realización del proyecto sea más certera para que al final se pueda elaborar lo que el cliente solicito con mas exactitud.

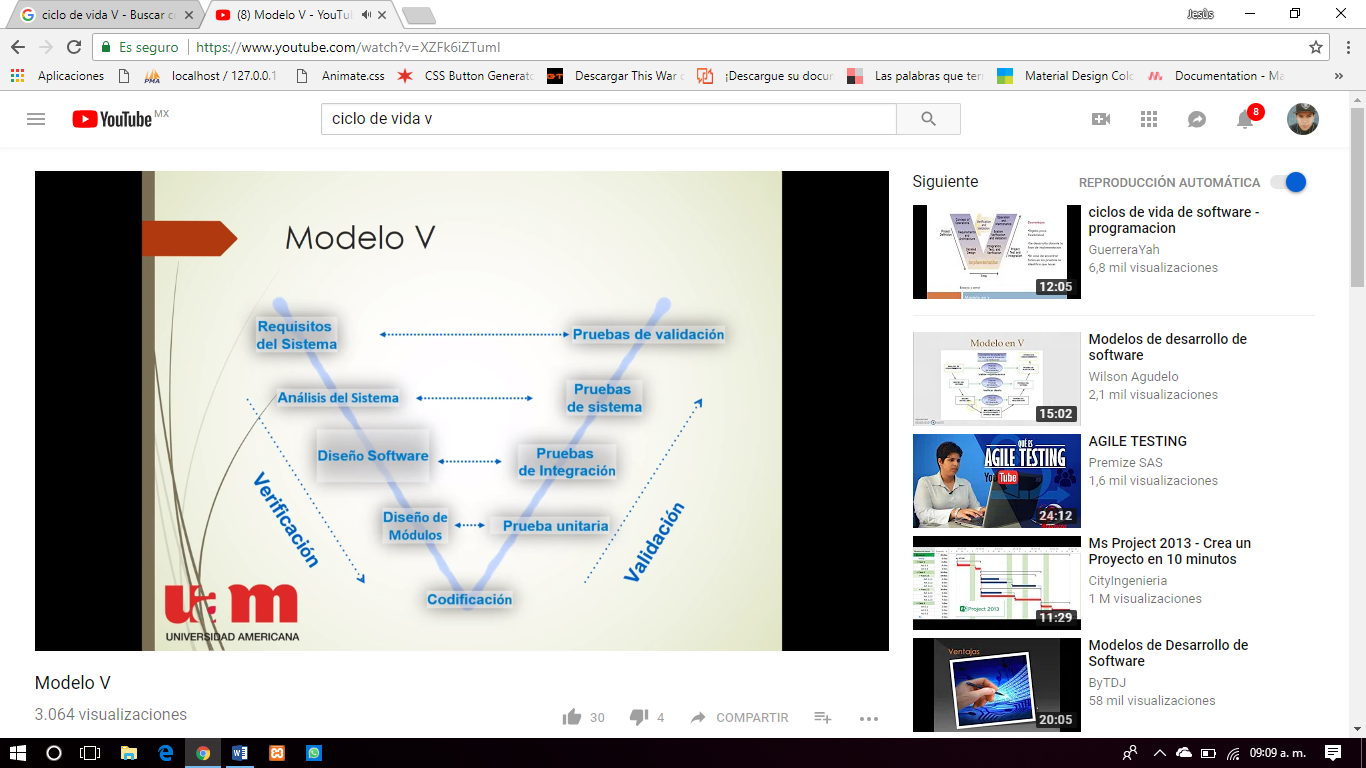


Ilustración 1 (Cam, 2016).

## JUSTIFICACIÓN DE LAS ACTIVIDADES DEL PROCESO.

## Fases de verificación

Requisitos del sistema:

En esta fase o etapa del proyecto se colectan los requisitos del sistema, así como también se analizan las necesidades del usuario es decir porque necesita el cliente el software para poder darle un enfoque el proyecto, en otras palabras, aquí se determina que es lo que desea que el software realice, sin embargo, no se debe confundir ya que no determina como será construido o diseñado.

Análisis del sistema:

En esta fase del proyecto se analizan los requisitos que el usuario proporciono en la fase anterior, por medio de este análisis se determina que es lo que en realidad el software debe realizar, se podría decir que es el puente entre fase de requerimientos y el código.

Diseño del software:

En esta etapa los técnicos analizan y entienden el funcionamiento o negocio del software para poder determinar las especificaciones que se implementaran y las que no, y así elaborar diferentes diagramas para poder desarrollar el producto, los técnicos se basan en las especificaciones de los consumidores, se calcula cada posibilidad de las exigencias del cliente.

Diseño de módulos:

En esta etapa se codifican los módulos del sistema es decir que se hacen por separado(fragmentos).

Codificación:

Esta es la etapa central de todo el ciclo de vida en esta etapa se desarrolla el producto es decir se programa todo lo establecido para poder elaborar el producto final.

## Fases de validación

## Prueba unitaria:

En esta etapa se valida el diseño de módulos en caso de que existan errores mediante el análisis del código se identifican los primeros errores y esto pude reducir costos al proyecto ya que se identifican inmediatamente los errores.

Pruebas de integración:

En esta fase se valida el diseño de módulos, también se realizan pruebas de los diferentes fragmentos del software, pero ya en conjunto y poder detectar errores en interfaces y en la integración de los componentes.

Pruebas del sistema:

En esta etapa del ciclo de vida se valida el análisis del sistema se hace una comparación entre el análisis de lo que se deseaba realizar en contra del sistema producido para poder determinar si realmente se elaboró lo que se planeó.

Pruebas de validación:

Finalmente, en esta etapa se validan los requerimientos del software y se revisa el producto final para determinar si se cumplieron las exigencias del cliente o lo que deseaba que el software realizara.