Unidad 1 / Escenario 1

Lectura fundamental

Conceptos generales de la Ingeniería del *Software*

Contenido

Palabras clave: *Software*, Procesos, Métodos, Modelos, Desarrollo de *Software*, Ingeniería del *Software*, Procesos del *Software*.

1. Ingeniería del *Software*

Se dice que la palabra ingeniería tienes sus raíces en la palabra ingenio, que viene del latín *ingenium*, la cual hace referencia a la capacidad de imaginar o inventar cosas con inteligencia y habilidad con los conocimientos que posee y los medios de que dispone. Se puede decir entonces que la ingeniería del *software* es la capacidad de inventar-solucionar un problema o aprovechar una oportunidad a través de la elaboración de un *software*. Sin embargo, más allá de eso, la ingeniería del *software* es la disciplina que se ha hecho cargo de resolver los problemas o necesidades a que se han enfrentado los ingenieros en el proceso mismo de construir el *software*, ya que, como se verá más adelante, las características del *software* han evolucionado ha pasado agigantados jalonados entre otros por:

- Las oportunidades que han ofrecido el vertiginoso crecimiento de las capacidades del *hardware* y las tecnologías de comunicación.
- El impacto que el *software* ha tenido en el desarrollo de la industria y prácticamente de todas las actividades humanas.
- El desarrollo de las competencias y habilidades de uso y apropiación de tecnologías de información y comunicaciones.

Todo lo anterior ha hecho que constantemente se hagan nuevas preguntas como ingenieros, para poder evolucionar a la par de las posibilidades, cumpliendo con los requisitos que se nos imponen. Cada vez que se construye un *software* se enfrentan retos como: mayor complejidad, más exigencia en la calidad (más adelante se verá cuáles son los atributos de calidad del *software*), más personas involucradas en su implementación, entre otros. Todo esto, haciendo el mejor uso posible de los recursos que incluyen: el equipo de trabajo, recursos financieros, recursos de *hardware* y el tal vez más cuestionado de todos en la industria de la construcción del *software*: el tiempo. Por resolver todas estas inquietudes es que aparece la ingeniería del *software* que es la disciplina encargada de desarrollar y mantener *software* de calidad a través de metodologías y técnicas específicas.

Este Escenario se enfoca en comprender los conceptos más importantes de la ingeniería del *software*, así como los enfoques que se han desarrollado a lo largo de la historia. Para lograr esto se aborda: qué es, cómo nació y se ha desarrollado y que elementos la componen. En las siguientes Unidades se exploran con mayor detalle las principales aproximaciones y herramientas en el desarrollo de *software* y tendrá la oportunidad de aplicar estos conceptos en tu proyecto. Para entender cómo se ha desarrollado la ingeniería del *software* vale la pena empezar por resolver la pregunta inicial.

2. ¿Qué es *Software*?

En el dominio de la informática básica se suele diferenciar el *software* y el *hardware* como las dos partes que componen un ordenador, el *software* es la "parte no visible" conformada por el conjunto de programas, instrucciones y reglas informáticas que hacen funcionar el *hardware*, que es el conjunto de los componentes físicos de los que está hecho el equipo. Sin embargo, con la evolución que ha tenido el *software* y en el contexto de la ingeniería del *software*, este es un concepto que se queda corto; entonces, hablaremos de que el *software* está compuesto por los elementos que se muestran en la **Figura 1**. Elementos que componen el *software*.

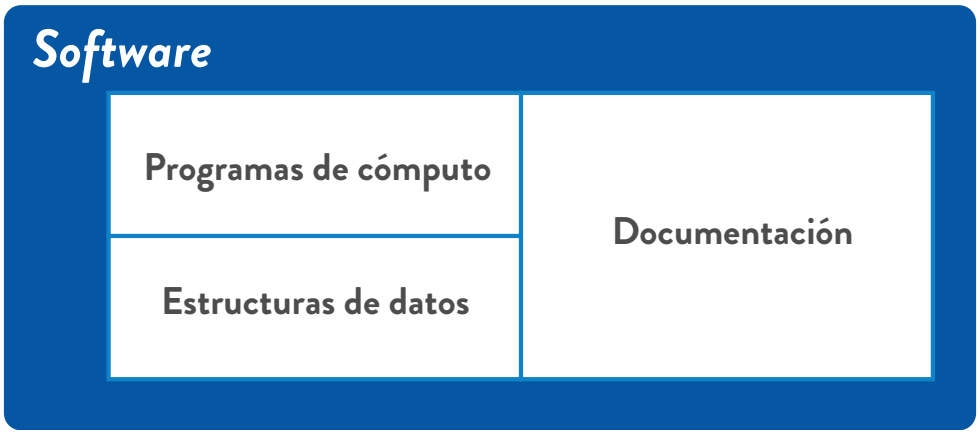

Figura 1. Elementos que componen el *software*

Fuente: elaboración propia

1. Programas de cómputo: son esa parte "no visible" que cuando se ejecuta ofrece las funciones y desempeños deseados.
2. Estructuras de datos: que permiten que los programas de cómputo gestionen la información.
3. Documentación del *software*: es la información que describe el diseño, operación y uso de los programas.

Este último elemento ha sido frecuentemente subestimado; sin embargo, un *software* de calidad no está completo sin la documentación que garantice su mantenibilidad y evolución. El prestigio como ingeniero de *software* depende en gran medida de la calidad de la documentación que genere. Siempre recuerde que está construyendo un *software* para otros, no solo para sus usuarios sino para que en un futuro otros ingenieros puedan evolucionar su *software*.

Durante la experiencia profesional en este campo una de las mayores satisfacciones es visitar las empresas donde se ha desarrollado *software* y ver que siguen en producción a pesar del tiempo y los cambios por lo que han pasado las empresas y que nunca me han necesitado para comprenderlo y poder mantenerlo, eso habla de un buen trabajo. Con esta definición de *software* podemos pasar a una definición más formal de lo que es la ingeniería del *software*.

3. ¿Qué es la ingeniería del *software*?

Es el área de la ingeniería que ofrece métodos y técnicas para gestionar el proceso de construcción, mantenimiento y evolución de un producto de *software* de calidad. Ampliando un poco más el concepto expuesto anteriormente y, de acuerdo con Pressman (2010), la ingeniería del *software* es una tecnología que comprende al menos cuatro capas que se ilustran en la **Figura 2.** Capas que componen la ingeniería del *software*.

Figura 2. Capas que componen la ingeniería del *software*
Fuente: elaboración propia

- Capa de compromiso con la calidad: esta capa es la base de la ingeniería del *software*, que define una cultura de mejora continua lo que lleva al desarrollo de enfoques cada vez más eficaces de ingeniería del *software*.
- Capa de Procesos: Un proceso de *software* es un conjunto coherente de: políticas, estructuras organizacionales, tecnologías, procedimientos y artefactos; que son necesarios para: concebir, desarrollar, instalar y mantener un producto *software*.

- Capa de métodos: proporcionan las herramientas o técnicas para elaborar *software*, lo cual incluye la definición de las tareas que se realizan para construir el *software* como son: análisis de requerimientos, modelación del diseño, construcción del programa, pruebas y mantenimiento.
- Capa de herramientas: dan apoyo automatizado o semiautomatizado para el proceso y los métodos.

Vamos a ahondar, en el concepto de la segunda capa, la capa de procesos de *software*, las capas de métodos y herramientas las abordaremos con más detalle en los siguientes escenarios.

- Sabías que el término Ingeniería del *software* se usó por primera vez en 1968 en la *Software Engineering Conference* organizada por la OTAN.

4. ¿Qué es un proceso de *software*?

En general, un proceso, como se ilustra en la **Figura 3.** Proceso, es un conjunto de tareas o actividades que transforman unas entradas en una salida o resultado esperado usando una serie de recursos.

Figura 3. Proceso
Fuente: elaboración propia

La definición de un proceso da respuesta a las preguntas fundamentales que se conocen como las 5w y 1h que es una regla mnemotécnica que se utilizó en el periodismo para generar noticias con información relevante y luego se expandió a otros ámbitos.

Tabla 1. Tabla de preguntas fundamentales en la definición de un proceso

What	*Who*	*Why*	*Where*	*When*	*How*
¿Qué?	**¿Quién?**	**¿Por qué?**	**¿Dónde?**	**¿Cuándo?**	**¿Cómo?**
¿Qué resultado se espera o cuál es el objetivo de la ejecución del propósito?	¿Quiénes están involucrados en la ejecución del proceso?	¿Por qué y para qué es necesario que se ejecute este proceso, qué impacto tendrá en el entorno la ejecución de este proceso?	¿Dónde se ejecutarán las actividades del proceso?	¿Cuándo se deben ejecutar las actividades del proceso?	¿Cómo se ejecutan las actividades que conforman el proceso?

Fuente: elaboración propia

Ahora bien, si hablamos específicamente del proceso de *software*, diremos en primer lugar que es un conjunto de actividades que toman como entrada las necesidades de unos usuarios y generan como salida un *software* que satisface estas necesidades, como se ilustra en la Figura 4. Entradas y salidas del Proceso de *Software*

Figura 4. Entradas y salidas del Proceso de *Software*
Fuente: elaboración propia

En segundo lugar, veremos que las preguntas de las 5w y 1h propuestas por Lasswell (1979), que mencionamos en la **Tabla 1.** de preguntas fundamentales en la definición de un proceso, toman un sentido particular cuando hablamos de un proceso de *software* como veremos a continuación:

- **¿Qué? *(What)*:** en este caso hablamos de las características o condiciones del producto que se requiere construir, más específicamente, ¿cuáles son los requerimientos de los usuarios del *software*? Al resolver esta pregunta se busca definir con la mayor precisión posible, **¿cuáles son las características del *software* que va a satisfacer las necesidades de los usuarios?**; para esto seguramente las 5W y 1H pueden ser preguntas que nos pueden ser muy útiles para entender estas necesidades y especialmente el contexto de aplicación del *software* que vamos a construir.

- **¿Quién? *(Who)*:** como lo mencionaba anteriormente, la creciente complejidad de los productos de *software* ha generado la necesidad de involucrar a más personas en el proceso de construir, mantener y evolucionar el *software*, esto hace necesario gestionar el trabajo de un equipo de personas y nos enfrenta a la necesidad de resolver preguntas más específicas como:
  - ¿Cuáles y cuántos son los roles que se requieren? O dicho de otra manera ¿quién hace qué?
  - ¿Qué perfiles se requieren para cada rol? Es decir, de acuerdo con los roles definidos, que conocimientos, experiencias, actitudes, aptitudes y habilidades deben cumplir los perfiles que se definan.
  - ¿Cómo se van a organizar estos roles? Es decir, ¿cuál es la estructura orgánica que define las responsabilidades y autoridades de cada quién
  - ¿Cómo se van a comunicar?

- **¿Por qué? *(Why)*:** esta pregunta la podemos ver desde el punto de vista del producto esperado, definiendo las razones o el propósito de construir el *software*, cuando tenemos claro por qué y para qué se hace el *software*; es decir, qué impacto se espera que tenga la implementación del *software*, será más fácil tomar decisiones de diseño del *software*. Sin embargo, también se puede ver desde el punto de vista del proceso de *software* en sí mismo, y define el por qué o las razones de gestionar el proceso de la manera en que se decida hacerlo. Como veremos más adelante, en la ingeniería del *software*, dar respuesta a cada una de las preguntas que surgen es un proceso de toma de decisiones constante en la que lo más importante es conocer las diferentes herramientas, metodologías y alternativas que tienen a disposición y use el criterio propio para determinar cuáles usar, cuándo y hasta dónde y para eso es importante que constantemente se formule las preguntas: ¿por qué? ¿por qué usar esta metodología y no otra? ¿por qué esta organización y no otra? ¿por qué en este momento y no en otro?

- **¿Dónde? *(Where)*:** esta pregunta, además de dar orientación con respecto a la ubicación geográfica, también puede dar orientación con respecto a la organización. Por ejemplo: nos permite hacernos preguntas sobre ¿dónde es más adecuado desarrollar el proceso de construir, mantener y evolucionar el *software*, dentro o fuera de la empresa? Es decir, que tenemos dos alternativas o combinaciones de ellas, cada una con sus propios desafíos, ventajas y desventajas a la hora de gestionarlos:
  - *In-house* (en casa): se desarrolla el *software* dentro de la organización.
  - *Outsourcing o freelance* (tercerización): delega la responsabilidad de desarrollo a un tercero.

- **¿Cuándo? *(When)*:** en este sentido se trata de las decisiones de gestión del tiempo, es decir, qué actividades realizar en qué momento, como veremos más adelante, las decisiones de qué actividades desarrollar antes, después o en paralelo con otras son muy importantes y definen el flujo del proceso de *software* más adecuado con respecto a las características del producto a desarrollar.
- **¿Cómo? *(How)*:** en este caso define específicamente la metodología que se va a utilizar para ejecutar el proceso de desarrollo, en la Unidad dos, veremos las diferencias que hay entre usar metodologías tradicionales y usar metodologías ágiles.

Por otro lado, la **Figura 5.** Estructura del Proceso de *Software*, muestra la organización esquemática de un proceso de *software* de manera jerárquica.

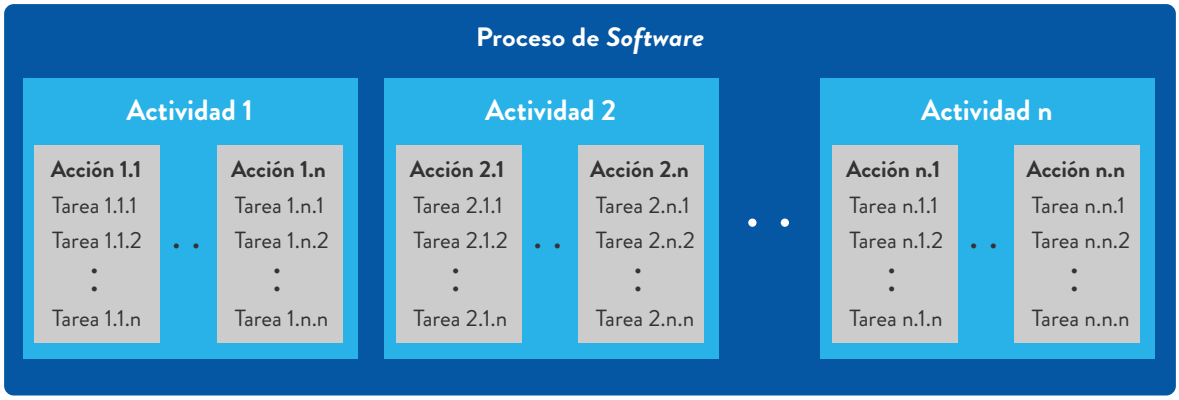

Figura 5. Estructura del Proceso de *Software*

Fuente: elaboración propia

De manera que un proceso está conformado por los siguientes elementos estructurales:

- Actividad: es un conjunto de acciones ordenadas que al ejecutarse generan un resultado o producto específico.
- Acción: es un conjunto ordenado de tareas que al ejecutarse generan el resultado esperado en la actividad.
- Tarea: son las asignaciones con el mayor grado de especificidad en términos de la complejidad de su ejecución.

A continuación, veremos los tipos de actividades que de acuerdo con Pressman (2010), comprenden un proceso de *software* estándar:

- Comunicación: este proceso se hace cargo de responder con la mayor especificidad posible las preguntas de ¿qué? y ¿por qué? que mencionamos anteriormente. Se trata de comunicarse y colaborar con todos los interesados o *stakeholders* en el desarrollo de *software* con el propósito de entender sus preocupaciones o *concerns* y reunir los requerimientos que define las funcionalidades y características del *software*.
- Planeación: en esta etapa de define el **plan de proyecto del *software***, es decir que se detalla al nivel de actividades, acciones y tareas el trabajo que se debe realizar, los riesgos probables, los recursos que se requieren para desarrollar estas actividades incluidos el recurso humano, los productos que se esperan obtener y la programación de las actividades. En otras palabras, la planeación se hace cargo de las preguntas ¿cómo? definiendo la estructura de actividades, acciones y tareas, ¿cuándo? al definir la programación de estas tareas, y ¿quién? al realizar la asignación de las mismas.
- Modelado: se trata de bosquejar, cuáles son las partes que componen en el *software*, cuáles son sus características, como se relacionan las partes que las componen y como este diseño propuesto da solución a los requerimientos identificados.
- Construcción: en esta actividad se genera el código y todos los artefactos que constituyen el *software*, que recordemos va más allá del código y comprende también las estructuras de datos y la documentación. En esta etapa se realización también las pruebas son necesarias para detectar y corregir errores que se puedan presentar.
- Despliegue: el *software* se entrega al cliente quien lo evalúa y da una retroalimentación.

En paralelo a estas actividades básicas se deben desarrollar entre otras las siguientes actividades:

- Seguimiento y control del proyecto
- Administración de riesgos
- Aseguramiento de la calidad
- Administración de la configuración

4.1. Flujos de Proceso

Ahora bien, la forma en que las actividades están organizadas con respecto al tiempo y a las secuencias define el flujo del proceso. A continuación se ven los distintos tipos de flujo de proceso:

4.1.1. Flujo de Proceso lineal

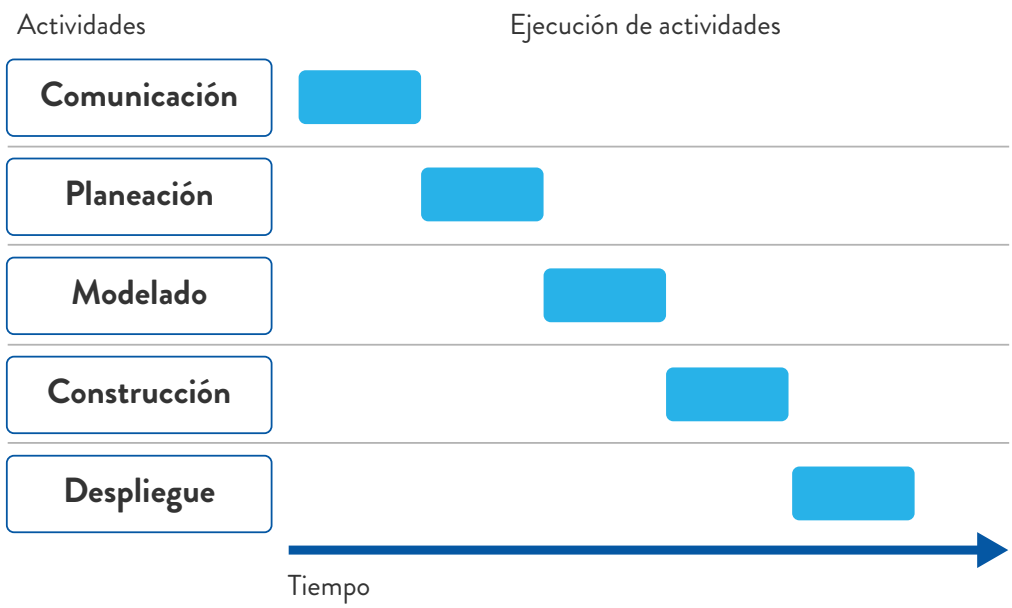

Figura 6. Flujo de proceso lineal

Fuente: elaboración propia

La Figura 6. Flujo de proceso lineal, ilustra un flujo de proceso que desarrolla las actividades de manera estrictamente secuencial; es decir, que solamente se desarrolla una actividad a la vez ajustándose rigurosamente al orden establecido empezando por la actividad de comunicación y terminando por el despliegue.

4.1.2. Flujo de Proceso Iterativo

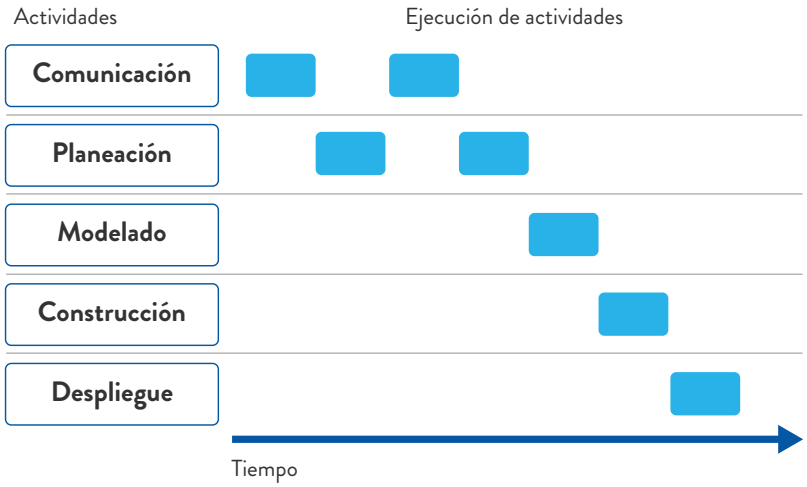

Figura 7. Flujo de proceso iterativo

Fuente: elaboración propia

La Figura 7. presenta una secuencia en la que una o más actividades pueden repetirse antes de pasar a la siguiente actividad, en el ejemplo vemos que las actividades de comunicación y planeación se repiten una vez antes de continuar con la actividad de modelado; sin embargo, se conserva la condición de que sólo la realiza una actividad a la vez.

4.1.3. Flujo de Proceso Evolutivo

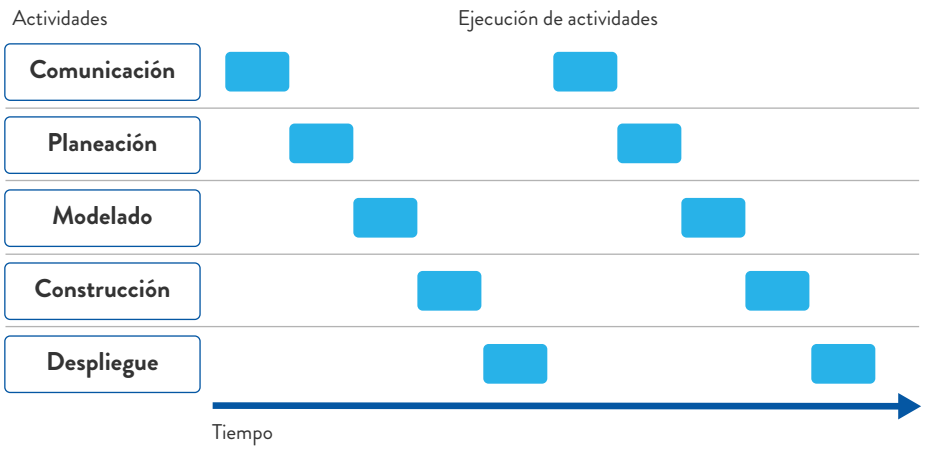

Figura 8. Flujo de proceso evolutivo

Fuente: elaboración propia

La **Figura 8.** presenta una secuencia donde se realizan varios ciclos del flujo lineal, la ejecución de cada ciclo genera una versión más completa o compleja del *software*.

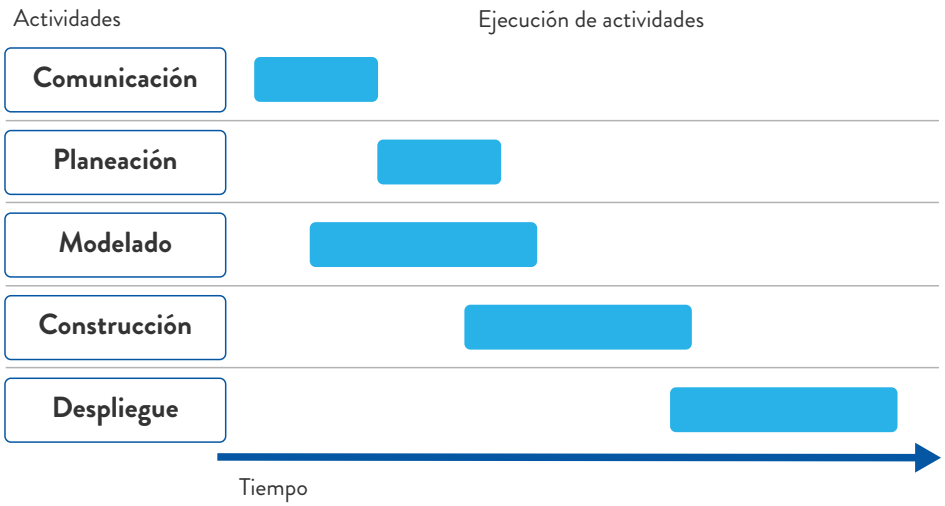

Figura 9. Flujo de proceso paralelo

Fuente: elaboración propia

La **Figura 9.** Flujo de proceso paralelo, muestra una secuencia en la que se ejecutan una o más actividades a la vez, es decir que no es un requisito indispensable haber terminado una actividad para iniciar la ejecución de otra.

En el siguiente Escenario ahonda en los detalles del proceso de *software* explorando los distintos modelos de procesos que se diferencian especialmente en el flujo de procesos seleccionado, los cuales tienen unas ventajas y desventajas específicas y que se deben elegir de acuerdo con las características del proyecto que se va a desarrollar.

Referencias

Lasswell, L. (1979) *Estructura y función de la comunicación en la sociedad, en: Sociología de la comunicación de masas*. Barcelona. p. 158-172

Pressman, R. (2010). *Ingeniería del software. Un enfoque práctico.* México, D. F, México: McGraw Hill p. 1,26)

INFORMACIÓN TÉCNICA

FACULTAD DE
INGENIERÍA, DISEÑO
E INNOVACIÓN

Módulo: Ingeniería del *Software* I

Unidad 1: Ingeniería del *software* conceptos e historia

Escenario 1: Conceptos generales de la Ingeniería del *Software*

Autor: Diana Angélica Cruz Ortega

Asesor Pedagógico: Luisa Esperanza Rincón Jiménez

Diseñador Gráfico: Cristian Navarro

Asistente: Ginna Quiroga