Ingeniería de software

Elementos de Informática

¿Dónde está presente el software?

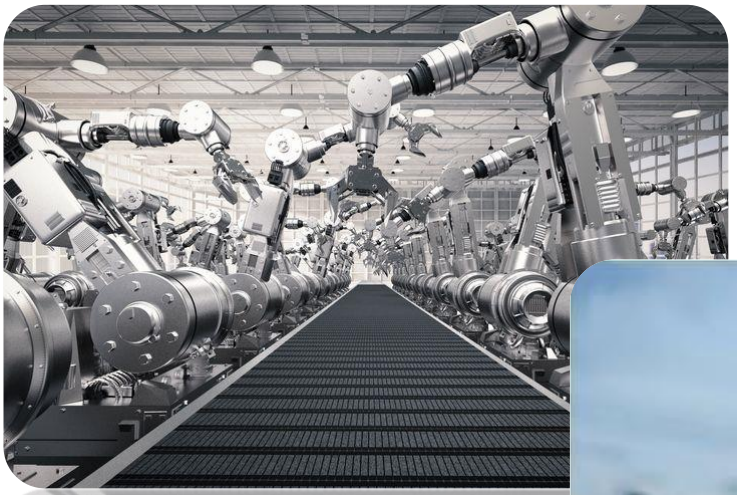
Infraestructuras
Nacionales Y
Servicios Públicos

RENTAS
Dirección Provincial de Rentas

AFIP

Productos
Electrónicos

¿Dónde está presente el software?



Fabricación Industrial

Sistema Financiero

Entretenimiento

Ingeniería de Software: Introducción

Hay muchos tipos diferentes de sistemas de software que necesitan diferentes técnicas de ingeniería de software para evitar fallas de software producto de las demandas crecientes de sistemas más grandes y complejos que deben construirse y distribuirse rápidamente y de las bajas expectativas desarrollando sistemas más costosos y menos confiables.

Ingeniería de Software: Historia

El concepto “ingeniería de software” se propuso originalmente en 1968, en una conferencia realizada para discutir lo que entonces se llamaba la “crisis del software” (Naur y Randell, 1969). Se volvió claro que los enfoques individuales al desarrollo de programas no escalaban hacia los grandes y complejos sistemas de software. Éstos no eran confiables, costaban más de lo esperado y se distribuían con demora.

A lo largo de las décadas de 1970 y 1980 se desarrolló una variedad de nuevas técnicas y métodos de ingeniería de software, tales como la programación estructurada, el encubrimiento de información y el desarrollo orientado a objetos. Se perfeccionaron herramientas y notaciones estándar y ahora se usan de manera extensa.

Ingeniería de Software: Definición 1

La ingeniería del software, según la definición de la IEEE en 1993, es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software.

La ingeniería del software ofrece métodos o técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo, y trata áreas muy diversas de la informática y de las ciencias computacionales.

IEEE
Advancing Technology for Humanity

Institute of Electrical and Electronics Engineers

Ingeniería de Software: Definición 2

Richard Fairley: Disciplina tecnológica y de administración que se ocupa de la producción y evolución sistemática de productos de software que son desarrollados y modificados dentro de los tiempos y costos estimados.

Ingeniería de Software: Definición 2

Barry Boehm: Es la aplicación práctica de conocimiento científico al diseño y construcción de programas... y la documentación asociada requerida para su desarrollo, operación y mantenimiento.

Software Profesional

Un sistema de software desarrollado profesionalmente es usualmente más que un solo programa. El sistema consta de un número de programas separados y archivos de configuración que se usan para instalar dichos programas. Puede incluir el código fuente; documentación del sistema que describe la estructura del sistema; documentación del usuario, que explica cómo usar el sistema y los sitios web para que los usuarios descarguen información reciente del producto

Productos de Software: Tipos

Existen dos tipos de productos de software:

- Productos genéricos: Son software producidos por una organización y vendidos en el mercado abierto. Por ejemplo: Software para bases de datos, procesadores de texto, paquetes de dibujo, sistemas de propósitos generales (facturación, contabilidad, historias clínicas, etc.)
- Productos Personalizados: son sistemas destinados a un cliente en particular en los que el desarrollador debe ajustarse a las especificaciones del cliente. Por ejemplo: sistemas de control de tráfico aéreo, sistemas de control para dispositivos electrónicos, sistemas de apoyo para cierto proceso empresarial, etc.

Productos de Software: Tipos

La distinción entre estos tipos de producto de sistemas se ha vuelto difusa.

Cada vez más sistemas se construyen con un producto genérico como base, que luego se adapta para ajustarse a los requerimientos de un cliente.

Los sistemas Enterprise Resource Planning (ERP, planeación de recursos empresariales), como los sistema creados por SAP (Systems, Applications, Products in Data Processing), son los mejores ejemplos de este enfoque.

En este caso, un sistema grande y complejo se adapta a una compañía al incorporar la información acerca de las reglas y los procesos empresariales, los reportes requeridos, etcétera.

Ingeniería de Software

La ingeniería de software es una disciplina de ingeniería que se interesa por todos los aspectos de la producción de software, desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema después de que se pone en operación.

Aplica teorías, métodos y herramientas de manera selectiva donde es adecuado buscar y encontrar soluciones a cada problema.

La ingeniería busca obtener resultados de la calidad requerida dentro de la fecha y del presupuesto.

Calidad de Software

La calidad del software profesional no sólo se debe a lo que hace el mismo. También se debe tener en cuenta el comportamiento del software mientras se ejecuta, y la estructura y organización de los programas del sistema y la documentación asociada.

Esto se refleja en los llamados atributos no funcionales del software o calidad, que dependen evidentemente de su aplicación. Por ejemplo un sistema bancario debe ser seguro, un juego interactivo debe tener capacidad de respuesta, un sistema de conmutación telefónica debe ser confiable.

Calidad del Software: atributos

Características del Producto	Descripción
Mantenimiento	El software debe escribirse de tal forma que pueda evolucionar para satisfacer las necesidades cambiantes de los clientes. Éste es un atributo crítico porque el cambio del software es un requerimiento inevitable de un entorno empresarial variable.
Confiabilidad y seguridad	La confiabilidad del software incluye un rango de características que abarcan fiabilidad, seguridad y protección. El software confiable no tiene que causar daño físico ni económico, en caso de falla del sistema. Los usuarios malintencionados no deben tener posibilidad de acceder al sistema o dañarlo

Calidad del Software: atributos

Características del Producto	Descripción
Eficiencia	El software no tiene que desperdiciar los recursos del sistema, como la memoria y los ciclos del procesador. Por lo tanto, la eficiencia incluye capacidad de respuesta, tiempo de procesamiento, utilización de memoria, etcétera.
Aceptabilidad	El software debe ser aceptable al tipo de usuarios para quienes se diseña. Esto significa que necesita ser comprensible, utilizable y compatible con otros sistemas que ellos usan.

Ingeniería de Software

La ingeniería de software es importante por dos razones:

1. Con mayor frecuencia, los individuos y la sociedad se apoyan en los avanzados sistemas de software. Por ende, se requiere producir económica y rápidamente sistem confiables

2. A veces resulta más barato a largo plazo usar métodos y técnicas de ingeniería de software para los sistemas de software, que sólo diseñar los programas como si fueran un proyecto de programación personal. Para muchos tipos de sistemas, la mayoría de los costos consisten en cambiar el software después de ponerlo en operación.

Ingeniería de Software: Enfoque sistémico

El enfoque sistemático que se usa en la ingeniería de software se conoce como proceso de software.

Un proceso de software es una secuencia de actividades que conducen a la elaboración de un producto de software.

Existen cuatro actividades fundamentales que son comunes a todos los procesos de software.

Ingeniería de Software: Enfoque sistémico

1. Especificación del software:

   Clientes e ingenieros definen el software que se producirá y las restricciones en su operación.

Ingeniería de Software:
Enfoque sistémico

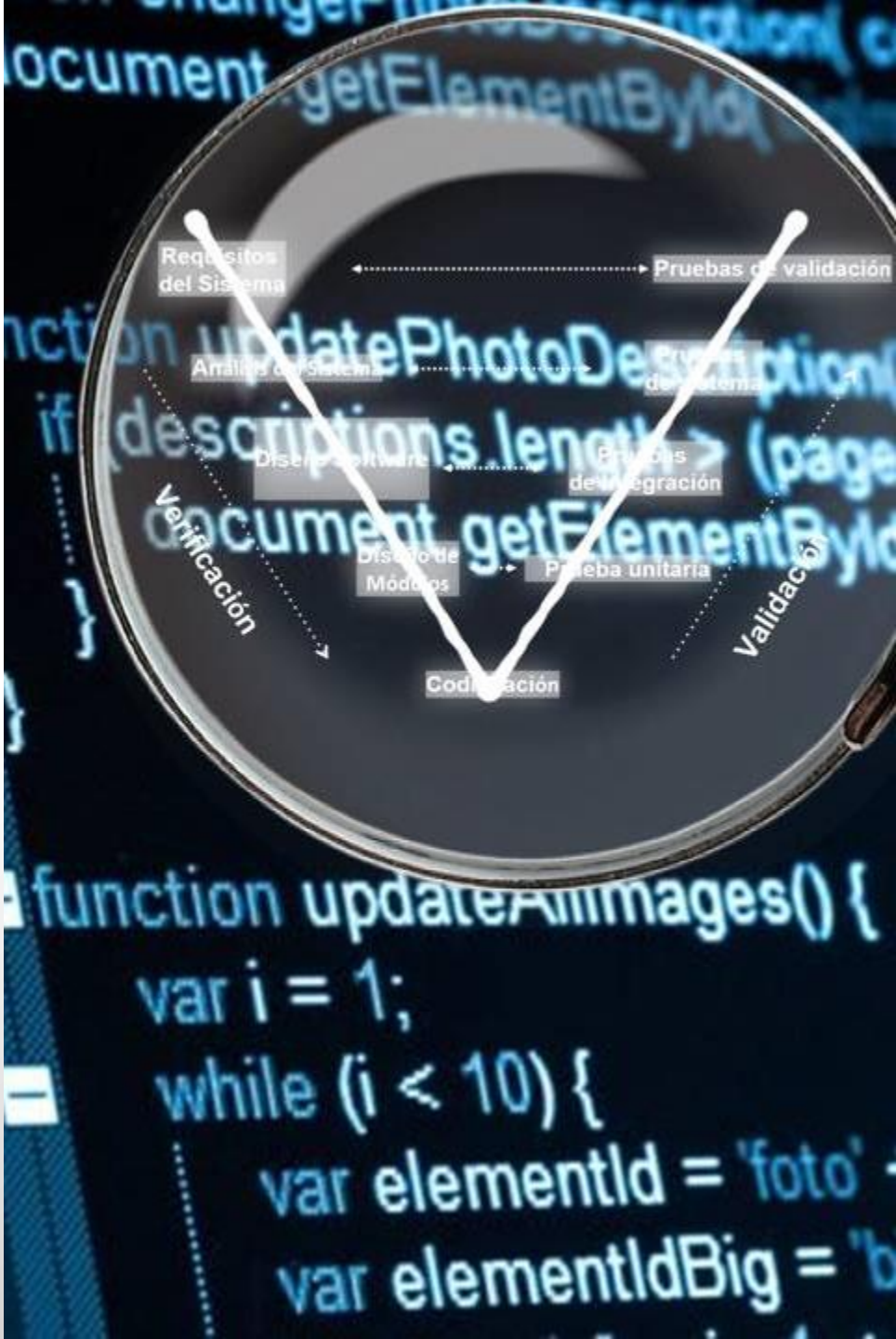
2. Desarrollo del software:

   Etapa en la que se diseña y programa el software.

Ingeniería de Software:
Enfoque sistémico

3. Validación del software:

   Etapa en la que se verifica el software para asegurar que sea lo que el cliente requiere.

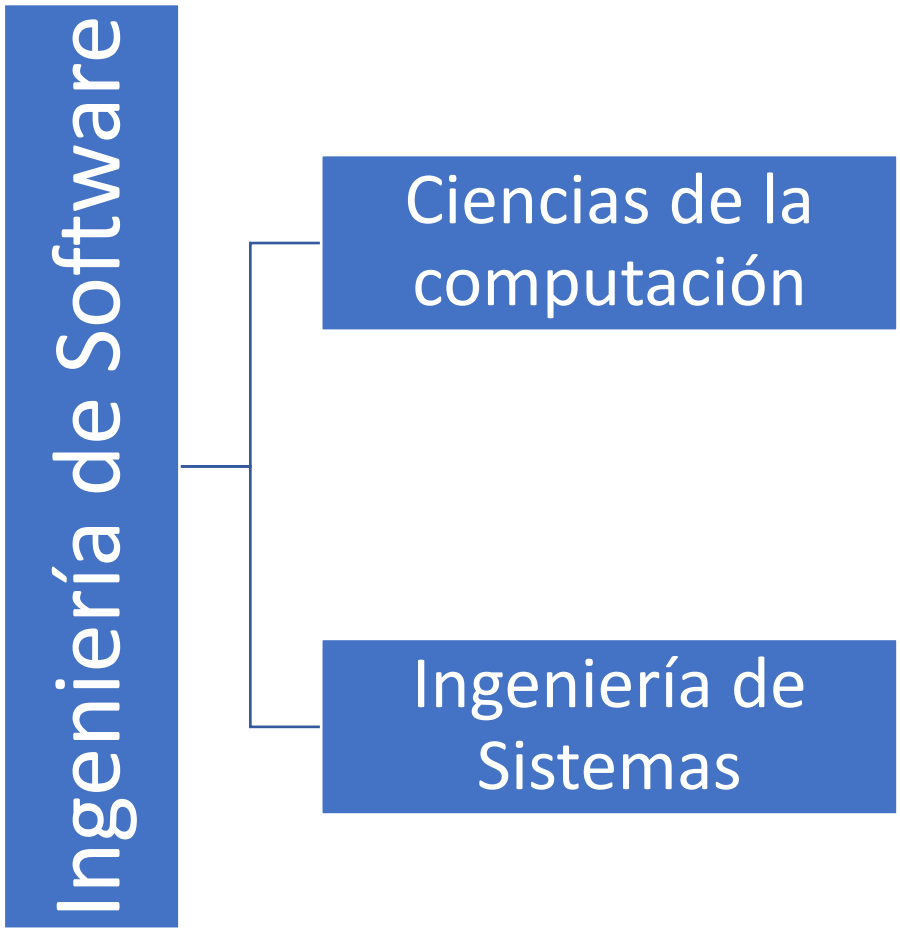


Ingeniería de Software:
Enfoque sistémico

4. Evolución del software

   Etapa en la que se modifica el software para reflejar los requerimientos cambiantes del cliente y del mercado.

Ingeniería de Software y sus relaciones



Se interesan por las teorías y los métodos que subyacen en las computadoras y los sistemas de software

La ingeniería de sistemas se interesa por todos los aspectos del desarrollo y la evolución de complejos sistemas, donde el software tiene un papel principal.

Retos que afronta la Ingeniería de Software

1. **Heterogeneidad**

   Cada vez con mayor frecuencia se requieren sistemas que operen como distribuidos a través de redes que incluyan diferentes tipos de computadoras y dispositivos móviles necesario desarrollar técnicas para construir software confiable que sea suficientemente flexible para enfrentar esa heterogeneidad

Retos que afronta la Ingeniería de Software

2. **Cambio empresarial y social**

   Los negocios y la sociedad cambian de manera increíblemente rápida, conforme se desarrollan las economías emergentes, nuevas tecnologías están a la disposición. Ambos necesitan tener la posibilidad de cambiar su software existente y desarrollar rápidamente uno nuevo.

Retos que afronta la Ingeniería de Software

3. **Seguridad y confianza**

El software está vinculado con todos los aspectos de la vida, por lo que es esencial confiar en dicho software. En especial para los sistemas de software remoto a los que se accede a través de una página Web.

Es necesario asegurarse de que usuarios malintencionados no puedan atacar el software y que se conserve la seguridad de la información.

Diversidad en la Ingeniería de Software

La ingeniería de software es un enfoque sistemático para la producción de software que toma en cuenta los costos, la fecha y la confiabilidad, así como las necesidades de clientes y fabricantes de software.

Este enfoque sistemático varía dependiendo de la organización que desarrolla el software, el tipo de software y los individuos que intervienen en el proceso de desarrollo.

Por lo tanto no existen métodos y técnicas universales de ingeniería de software que sean adecuados para todos los sistemas

Tipos de aplicaciones

- Aplicaciones independientes

  Son aplicaciones que corren en una computadora local, incluyen toda la funcionalidad necesaria y no requieren conectarse a una red. Ejemplos de tales aplicaciones son las de oficina en una PC, programas CAD, software de manipulación de fotografías.

- Aplicaciones interactivas basadas en transacción

  Consisten en aplicaciones que se ejecutan en una computadora remota y a las que los usuarios acceden desde sus propias PC o terminales. Evidentemente, en ellas se incluyen aplicaciones Web como las de comercio electrónico, servicios basados en la nube, como correo electrónico y compartición de archivos

Tipos de aplicaciones

- Sistemas de control embebido

  Son los sistemas de control de software que regulan y gestionan dispositivos de hardware. Algunos ejemplos de sistemas embebidos incluyen el software en un teléfono móvil (celular), el software que controla los frenos antibloqueo de un automóvil, el software en un horno de microondas, de un lavarropas

- Sistemas de procesamiento en lotes

  Son sistemas empresariales que se diseñan para procesar datos en grandes lotes (batch). Procesan gran cantidad de entradas individuales para crear salidas correspondientes. Por ejemplo los sistemas de facturación periódica, los sistemas de facturación telefónica, los sistemas de pago de salario.

Tipos de aplicaciones

- Sistemas de entretenimiento

  Son sistemas para uso personal, con el propósito de entretener al usuario. La mayoría de estos sistemas son juegos de uno u otro tipo. La calidad de interacción ofrecida al usuario es la característica más importante de los sistemas de entretenimiento.

- Sistemas para modelado y simulación

  Éstos son sistemas que desarrollan científicos e ingenieros para modelar procesos o situaciones físicas, que incluyen muchos objetos separados interactuantes. Dichos sistemas a menudo son computacionalmente intensivos y para su ejecución requieren sistemas paralelos de alto desempeño.

Tipos de aplicaciones

- Sistemas de adquisición de datos

  Son sistemas que desde su entorno recopilan datos usando un conjunto de sensores, y envían dichos datos para su procesamiento a otros sistemas.

- Sistemas de sistemas

  Son sistemas compuestos de un cierto número de sistemas de software. Algunos de ellos son producto del software genérico, como un programa de hoja de cálculo.

Ciclo de vida de un sistema

Se define como Ciclo de vida al periodo de tiempo que comienza al concebir la idea de un nuevo sistema de software, y termina cuando este se retira y deja de funcionar.

Un proceso es un conjunto de actividades y tareas relacionadas, que al ejecutarse de forma conjunta transforman una entrada en una salida.

Modelos de Proceso de Software

Un modelo de proceso, o paradigma de Ingeniería de Software, es una plantilla, patrón o marco que define el proceso a través del cual se crea software.

Los modelos de desarrollo definen la estructura de un proceso de desarrollo racional y controlable.

No existe un modelo universal, son una guía respecto al orden en que deben adelantarse las actividades, se basa en el reconocimiento que el software tiene un “ciclo de vida”.

Modelos de Proceso de Software

Modelos secuenciales

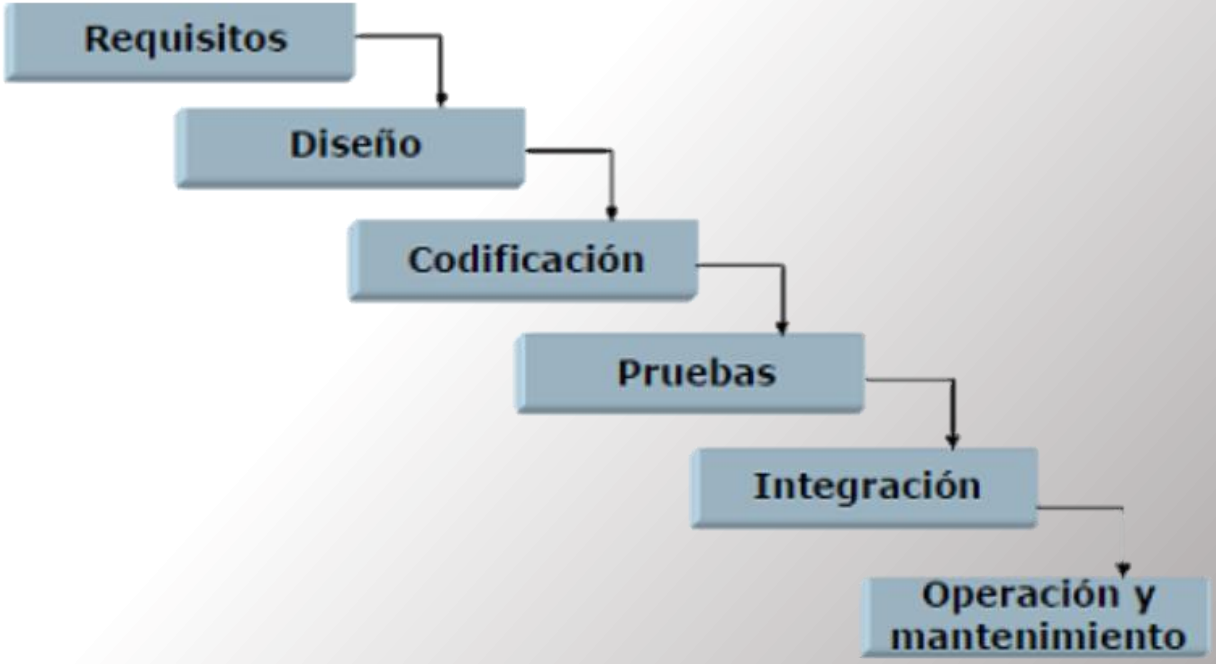
- Lineal
- Cascada
- RAD

Modelos evolutivos

- Incremental
- Espiral
- basado en reuso
- Prototipos.

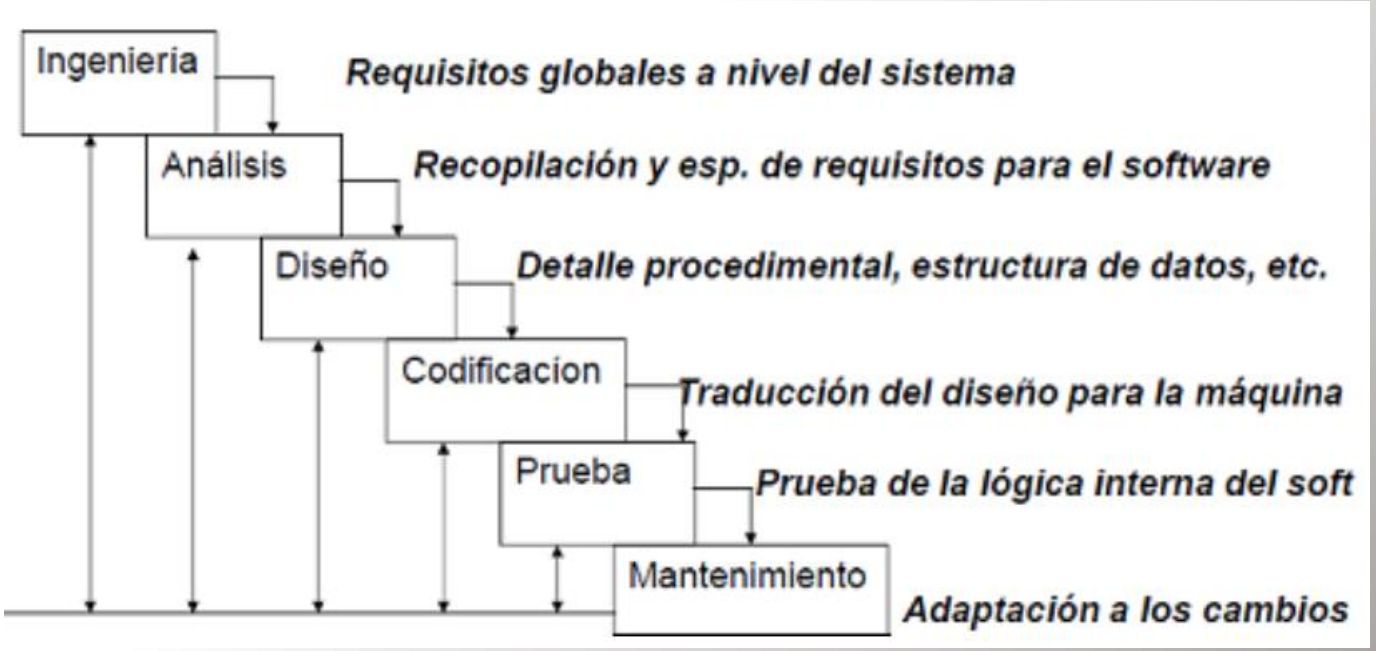
El modelo Lineal

El modelo establece una sucesión escalonada de las etapas que lo componen. Para comenzar una nueva etapa en es necesario terminar la etapa anterior dado que cada fase requiere como entrada el resultado de la anterior.



El modelo en cascada

Este modelo sigue un enfoque ascendente y secuencial. Toma las actividades fundamentales del proceso de especificación, desarrollo, validación y evolución y, luego, los representa como fases separadas del proceso, tal como especificación de requerimientos, diseño de software, implementación, pruebas, etcétera.



El modelo en cascada: Etapas

Ingeniería: en esta etapa se establecen los requisitos generales teniendo en cuenta que el sistema se debe interrelacionar con otros elementos tales como el hardware, los usuarios y las bases de datos.

Análisis: la construcción de un sistema requiere conocer el ámbito de la información del software, la función que debe cumplir y la interfaz. deben ser documentados y revisados con el cliente. Estos requisitos deben ser documentados y revisados con el cliente.

El modelo en cascada: Etapas

Diseño: en esta etapa se definen las estructuras de datos, la arquitectura del softare, los procedimientos y las características generales de la interfaz.

Codificación: es la etapa en la que el diseño de ser legible por una máquina, es decir en la que se programa.

Prueba: es la etapa en la que se realizan las pruebas sobre el software con el propósito de identificar errores.

Mantenimiento: etapa que atiende las modificaciones del sistema

Enfoque ascendente: Dificultades

- El usuario debe esperar a que el sistema esté totalmente terminado.
- Las fallas más complejas se encuentran al final del proceso de prueba y su eliminación es costosa
- En la etapa de prueba se incrementa la necesidad de realizar pruebas con la computadora a medida que se acerca la entrega del software.

Enfoque secuencial: Dificultades

- El deseo de progreso ordenado de las etapas no es nada realista.
- El usuario puede cambiar de parecer respecto a lo que debe hacer el sistema.
- Se apoya en técnicas anticuadas.
- Resulta apropiado para sistemas pequeños.

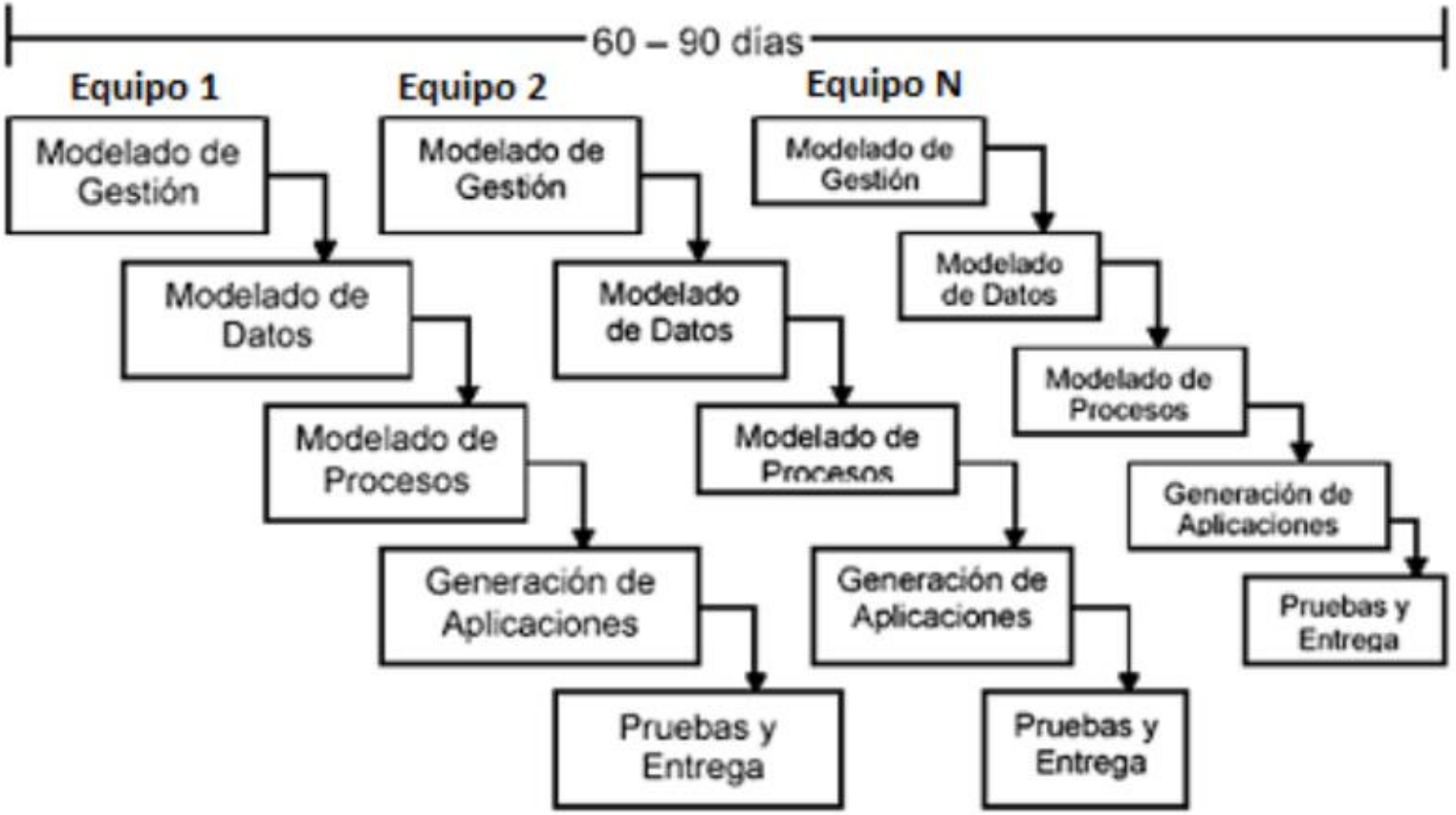
Modelo RAD (rapid application develoment)

Este modelo es una mejora del modelo lineal en el que los equipos de trabajo lo hacen en forma paralela.

Se utiliza para desarrollar grandes proyectos en el que el tiempo de espera es corto por lo que es necesario contar con los recursos suficientes.

No es recomendable para sistemas que requieren mantenimiento constante o que tienen un grado alto de interopertividad con otros programas

Modelo RAD (rapid application develoment)



Modelos evolutivos

Este modelo se caracteriza por gestionar en forma adecuada bien la naturaleza evolutiva del software.

Se adaptan fácilmente a los cambios a lo largo del desarrollo (requisitos, fechas de entrega, especificaciones parciales) construyendo versiones del software cada vez más completas.

Modelo en espiral

En este paradigma se definen cuatro actividades:

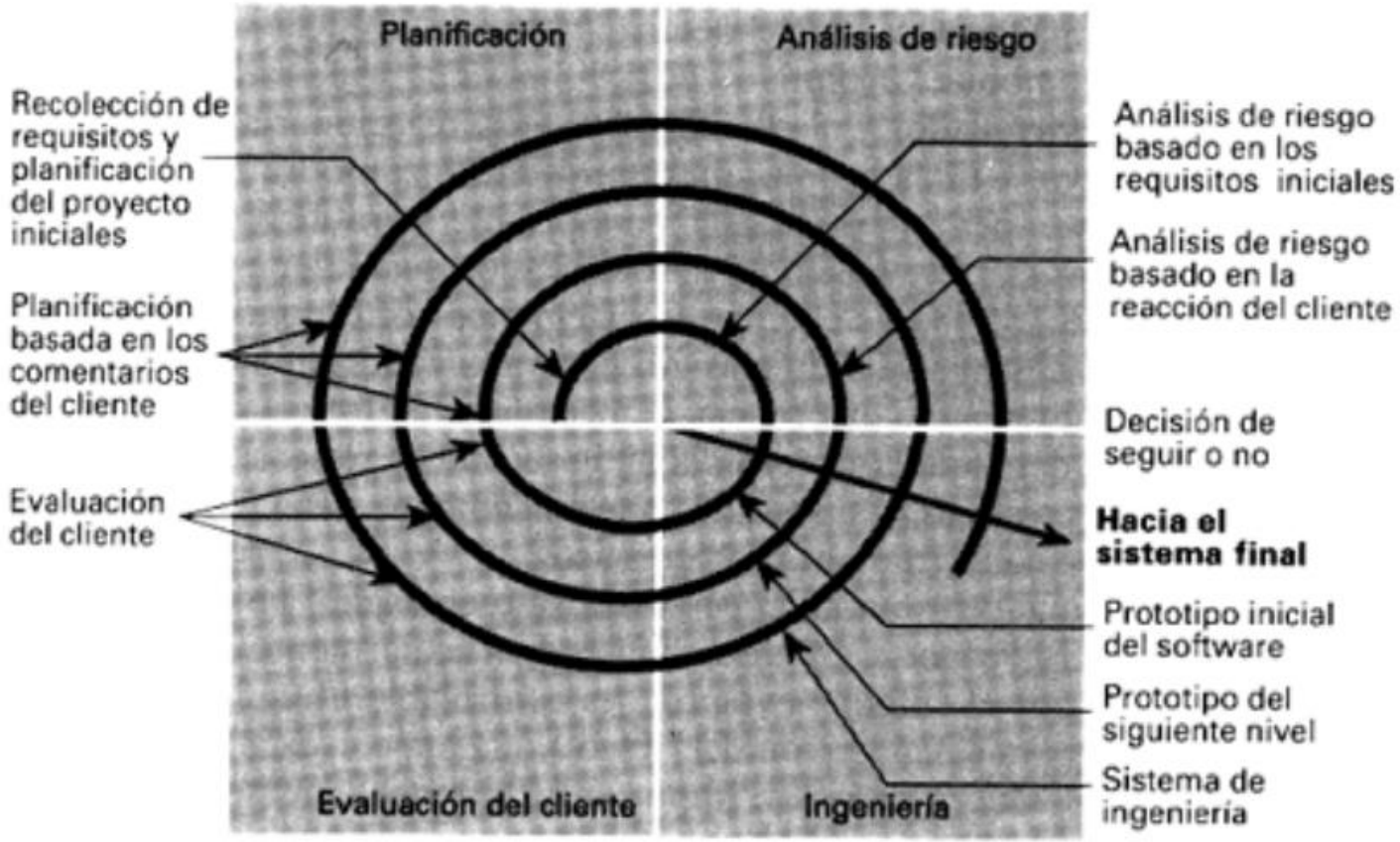
Planificación: Se determinan los objetivos, se recolectan los requisitos y se planifica una primera instancia a partir de los comentarios de los clientes.

Análisis de riesgos: se analizan diferentes alternativas analizando los riesgos a partir de los requisitos iniciales y de la reacción del cliente.

Ingeniería: se construye un prototipo cero y se refina en niveles superiores.

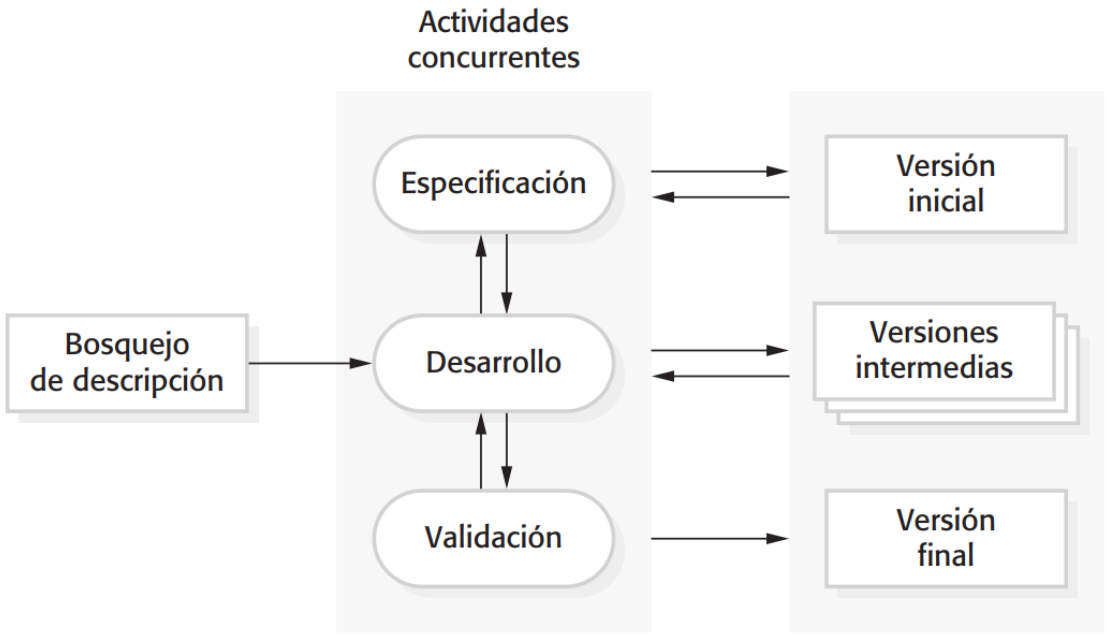
Evaluación del Cliente: se valoran los resultados de cada una de las etapas.

Modelo en espiral



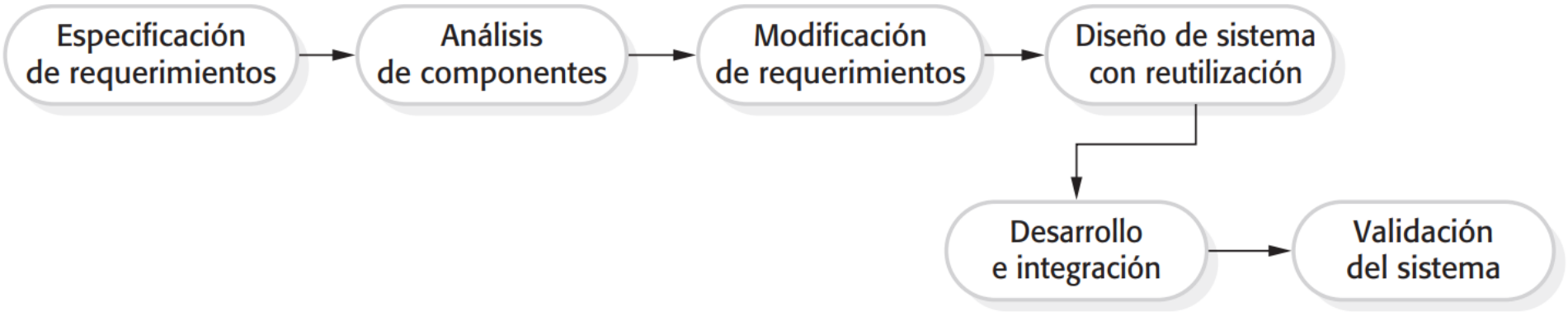
Desarrollo incremental

Este enfoque vincula las actividades de especificación, desarrollo y validación. El sistema se desarrolla como una serie de versiones (incrementos), y cada versión añade funcionalidad a la versión anterior.



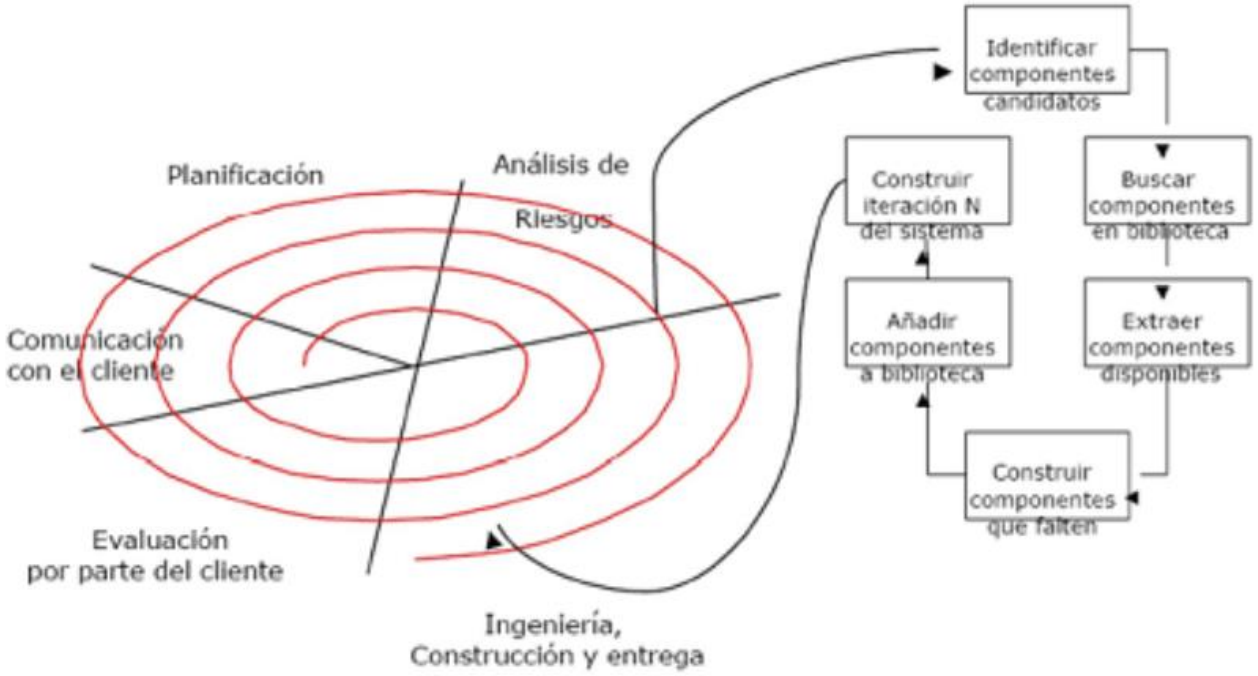
Ingeniería de software orientada a la reutilización

Este enfoque se basa en la existencia de un número significativo de componentes reutilizables. El proceso de desarrollo del sistema se enfoca en la integración de estos componentes en un sistema, en vez de desarrollarlo desde cero.



Ingeniería de software orientada a la reutilización

Este modelo también puede avanzar en forma espiralada buscando componentes para reutilizar y creando nuevos componentes.



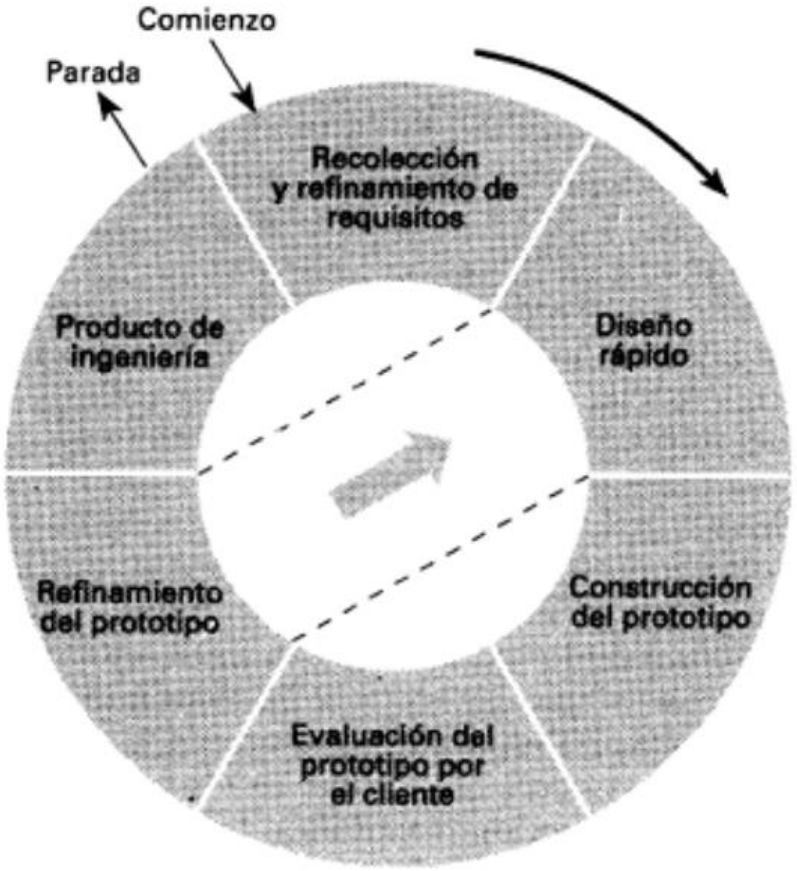
Modelo de Prototipos

En este modelo se construyen modelos de prueba sobre los que interactúa el usuario para obtener un feedback hasta lograr obtener los requisitos.

Los prototipos son pantallas que permiten simular como funcionará el sistema y cual será la interfaz.

El usuario evalúa el prototipo para avanzar con el refinamiento de los requisitos del software.

Modelo de Prototipos



Modelo Transformacional

Este modelo se basa en especificaciones formales de los requisitos con la finalidad de reducir errores.

Se requiere una formación especializada para aplicar la técnica.

Se aplica a sistemas en los que la seguridad y la fiabilidad deben asegurarse antes de ponerlos en funcionamiento.

Modelo TransformacionalS