El futuro digital es de todos

MinTIC

Universidad Industrial de Santander

MisiónTIC 2022

FUNDAMENTOS DEL LENGUAJE PYTHON

TEMA 2: MODELOS DE DESARROLLO DE SOFTWARE

2.1. ¿Cuál es el ciclo de vida del software?

Un ciclo de vida de desarrollo de *software* (SDLC) incluye los procesos de *software* utilizados para especificar y transformar los requisitos de *software* en un producto de *software* entregable. Está dividido en 4 categorías como se muestra a continuación:

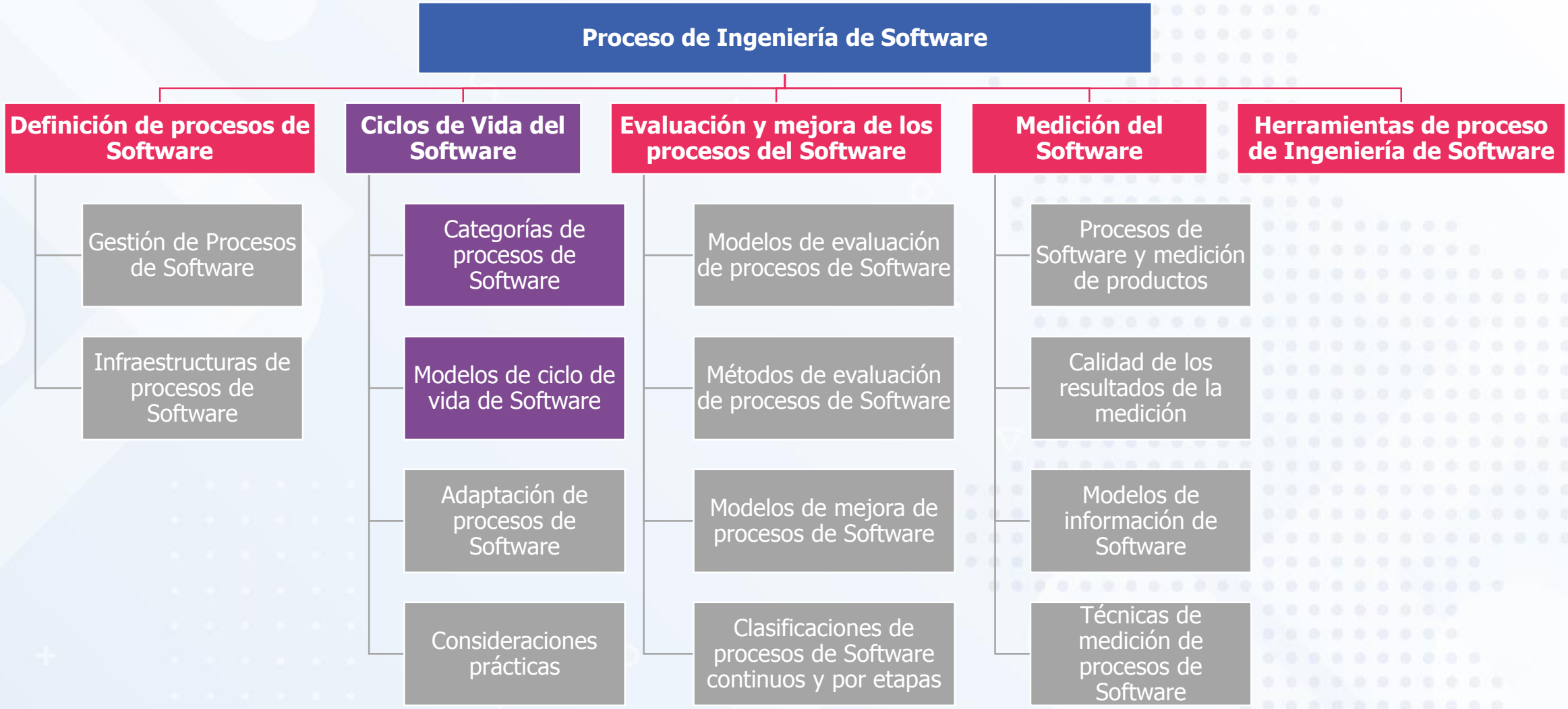
1. Procesos primarios
Incluye los procesos para análisis, diseño, desarrollo, operación y mantenimiento del *Software*.

2. Procesos de soporte
Son aplicados de forma intermitente o continua a lo largo del ciclo de vida de un producto *software* para respaldar los procesos primarios, incluyendo procesos de gestión de la configuración, aseguramiento de calidad, verificación y validación.

Ciclo de vida del software

3. Procesos organizacionales
Proporcionan soporte en el proceso de ingeniería de *software* incluyendo, capacitación, análisis de medición de proceso, gestión de infraestructura, reutilización, mejora de procesos organizacionales y gestión de modelos de ciclo de vida de *software*.

4. Procesos de proyectos cruzados
Como reutilización, línea de productos de *software*; pueden implicar más de un proyecto de *software* en una organización.



Proceso de ingeniería de software, Fuente: adaptado de [1]

2.2. Fases de desarrollo de software

Las fases en el desarrollo del *software* varían en la forma como son aplicadas según el modelo de ciclo de vida de *software* utilizado, pero de manera general comienzan con un proceso de elicitación o levantamiento de requisitos para realizar el diseño del producto *software* con base en un problema o contexto; posteriormente, pasan a procesos de desarrollo que, una vez finalizados son puestos a prueba en la implementación. Cuando un producto *software* es entregado a un cliente, queda con un vínculo constante de soporte o mantenimiento para incluir mejoras o corregir posibles eventualidades que surjan por errores no previstos en las etapas de análisis, diseño y desarrollo. Algo importante a tener en cuenta es que prevenir esos errores en etapas tempranas del ciclo de vida es menos costoso que cuando son detectados en etapas finales, esto puede generar sobrecostos en los proyectos de desarrollo o incumplimientos en las fechas de entrega establecidas.

2.3. Ciclo de vida del desarrollo de software (SDLC)

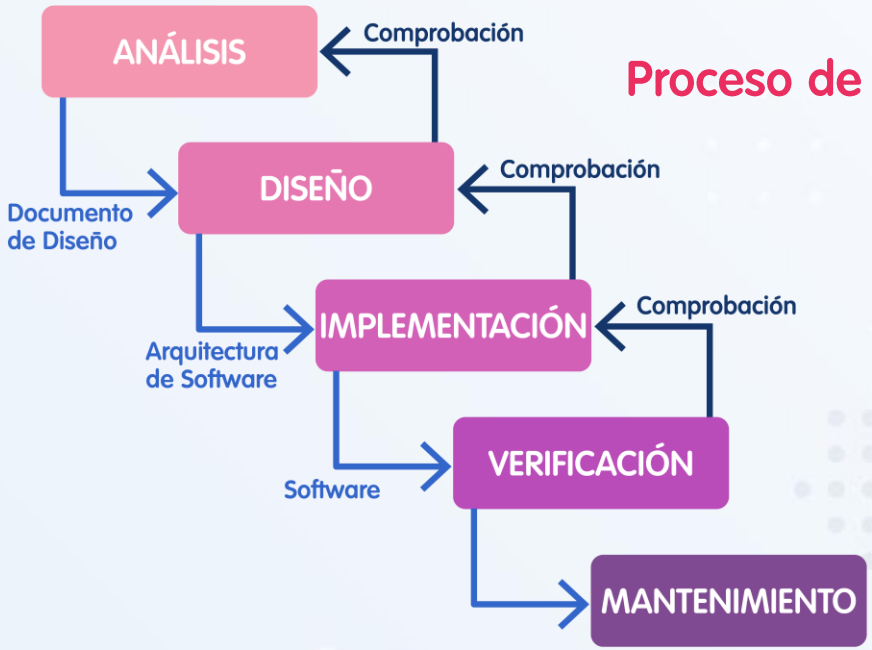
La naturaleza intangible y maleable del *software* permite una amplia variedad de modelos de ciclo de vida de desarrollo de *software* que van desde modelos lineales en los que las fases del desarrollo de *software* se logran secuencialmente con retroalimentación e iteración según sea necesario, seguidas de integración, prueba y entrega de un producto único; a modelos iterativos en los que el *software* se desarrolla en incrementos de funcionalidad creciente en ciclos iterativos; a modelos ágiles que normalmente implican demostraciones frecuentes de *software* en funcionamiento a un cliente o representante del usuario que dirige el desarrollo del *software* en ciclos iterativos cortos que producen pequeños incrementos de *software* que funciona y que se puede entregar. Los modelos incrementales, iterativos y ágiles pueden ofrecer subconjuntos tempranos de *software* de trabajo

Los modelos de SDLC lineales a veces se denominan modelos de ciclo de vida de desarrollo de *software* predictivo, mientras que los SDLC iterativos y ágiles se denominan modelos de ciclo de vida de desarrollo de *software* adaptativo. Cabe señalar que se pueden realizar diversas actividades de mantenimiento durante un ciclo de vida de producto *software* (SPLC) utilizando diferentes modelos de SDLC, según corresponda.

2.4. Modelos básicos de ciclos de vida de software (SDLC)

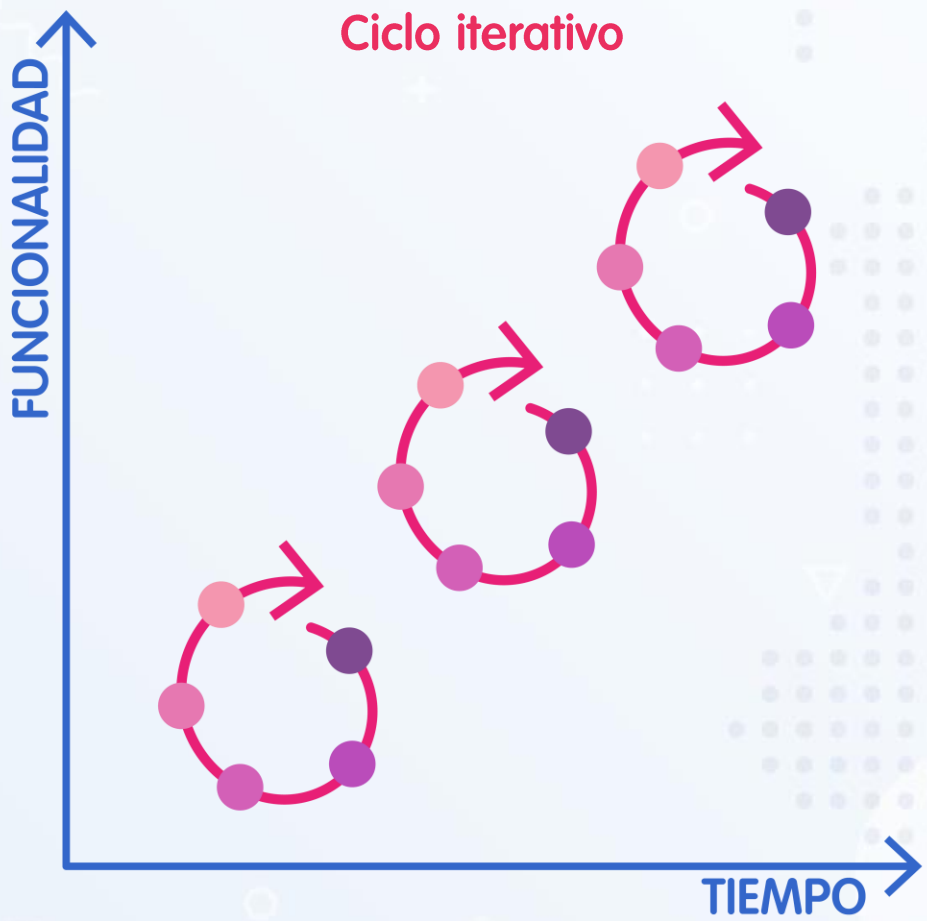
Adaptada de Modelos de desarrollo de software - Modelo V (K2 - entender, explicar , razonar) (2013). Recuperado de *http://scrum-qa.blogspot.com/2013/03/modelos-de-desarrollo-de-software.html*

Ciclo en cascada

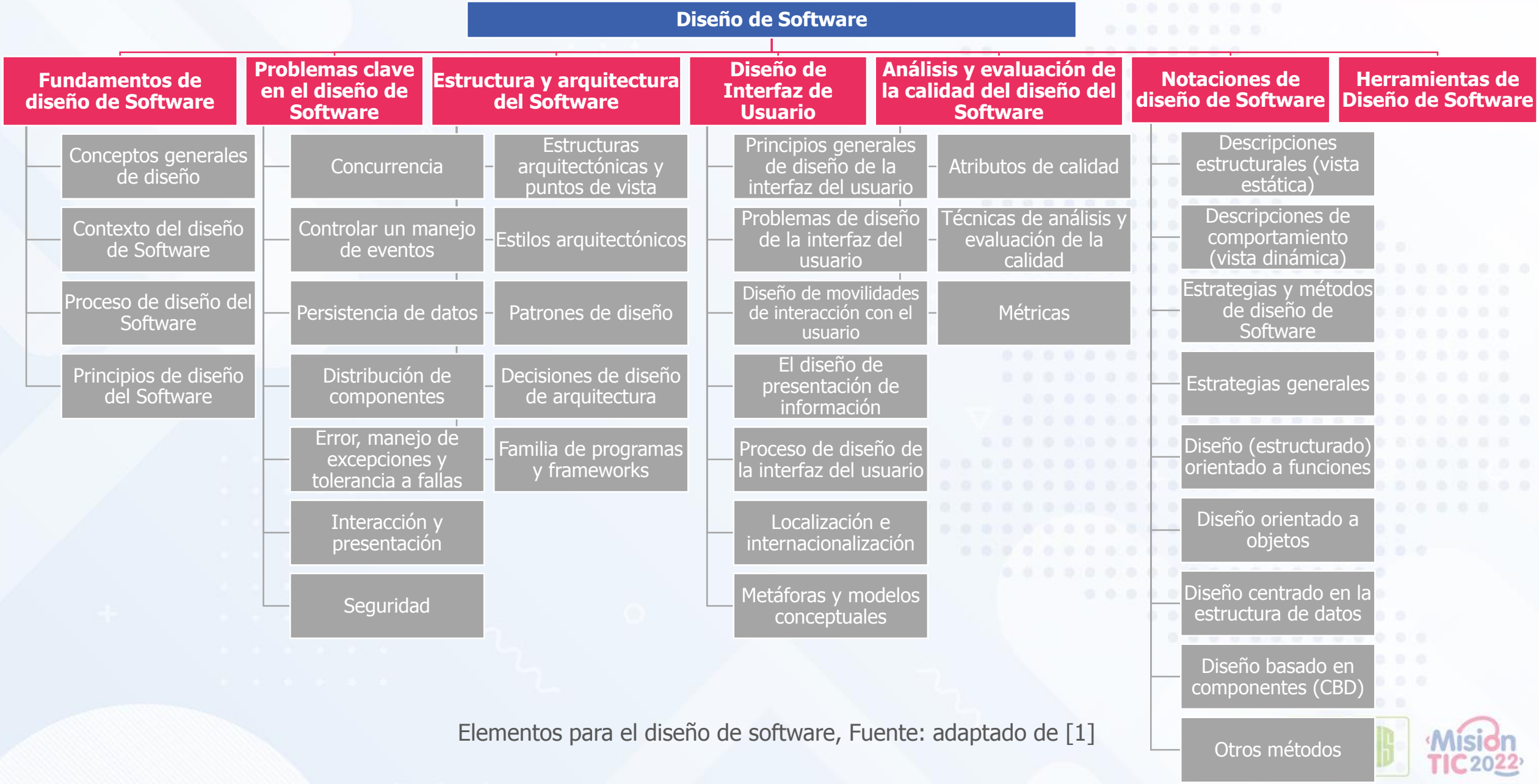


Proceso de desarrollo

Adaptada de: El modelo en cascada: desarrollo secuencial de software (2019). Recuperado de https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/



SARCO, (2012). ISTQB – CAP 2 – TESTING A TRAVÉS DEL CICLO DE VIDA DEL SOFTWARE – I Recuperado de: https://josepablosarco.wordpress.com/2012/03/24/istqb-cap-2-testing-a-traves-del-ciclo-de-vida-del-software-i/



Elementos para el diseño de software, Fuente: adaptado de [1]

El diseño de *software* contiene múltiples elementos para que sea llevado de manera correcta, pero centraremos nuestra atención al seleccionar 3 capas: persistencia, lógica y presentación, que permitirán elaborar al final del curso un programa funcional en el cual podamos poner en práctica lo aprendido.

CAPA	DESCRIPCIÓN
Presentación	Interface mostrada al usuario
Lógica	Funcionalidades del programa
Persistencia	Lectura y escritura de datos

Material complementario

Se recomienda la Lectura del capítulo 2 EVOLUTION OF COMPUTING EDUCATION del reporte de la Association for computing machinery (ACM), para identificar las múltiples disciplinas que se pueden abordar desde la computación en general. Hacer énfasis en la sección 2.3.6: *Software Engineering*, es donde se sitúa de mejor manera el tema expuesto en este capítulo.
De manera general, se motiva al estudiante a ver que existe una gran variedad de posibilidades para continuar aprendiendo o especializándose en una o varias disciplinas.

https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf

Referencias Bibliográfica

Hilliard, R. (noviembre de 2017). SWEBoK Evolution: una sesión de panel de CSEE & T 2017. En *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE & T)* (págs. 234-235). IEEE.