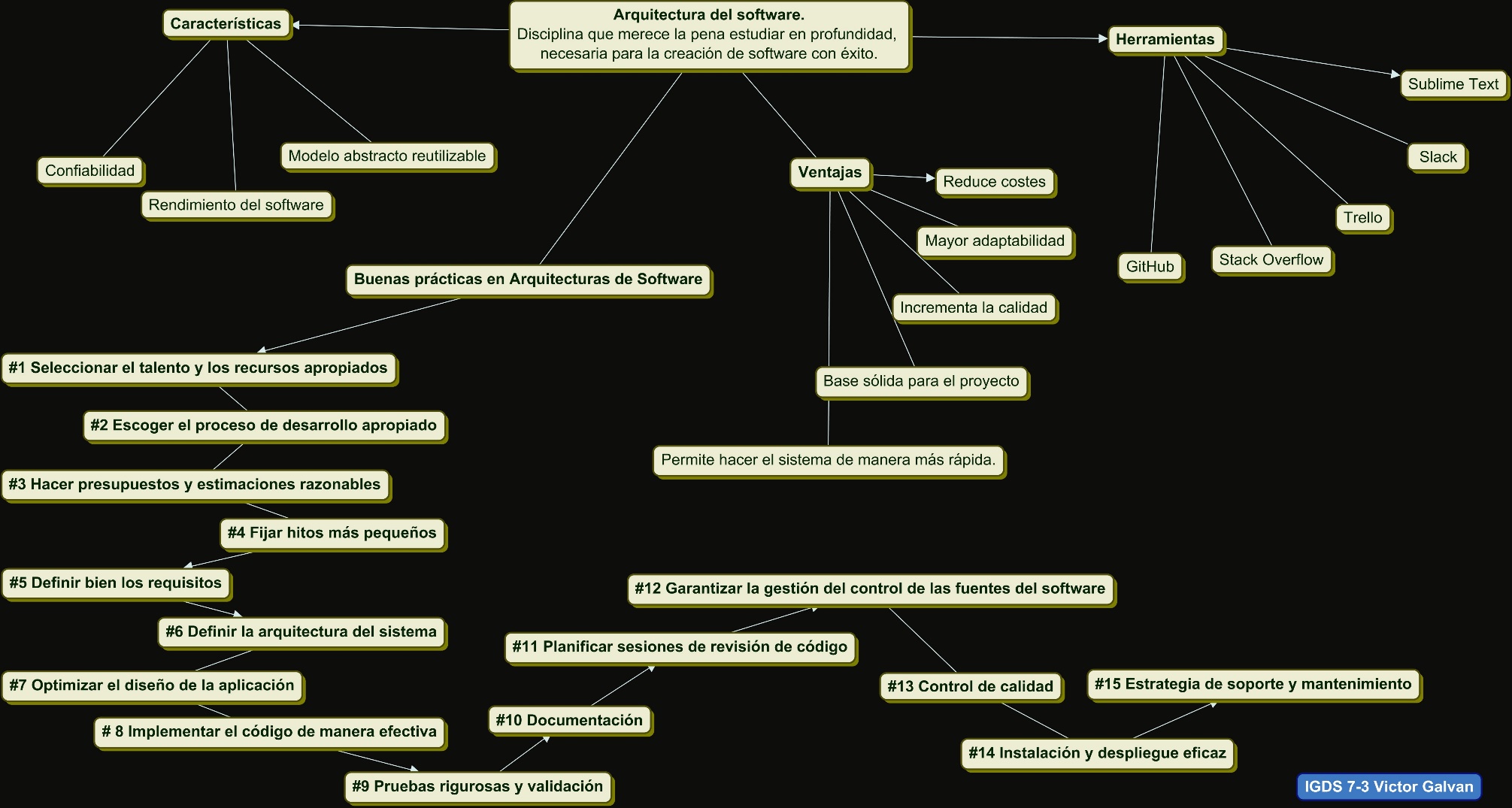
**Guía para el examen**

**Arquitecturas de software (Concepto, ventajas y desventajas)**

****

**Tipos de arquitecturas (Conceptos de cada tipo y ejemplos)**

**Cliente-Servidor**

Modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes.

**Ventajas**

Escalabilidad

Fácil Mantenimiento

**Desventajas**

Bajo rendimiento

Costos elevados

**Microservicios**

Construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros.

**Ventajas**

Facil de implementar

Reusabilidad del código

Idependencia de la aplicacion

**Desventajas**

Memoria

Difícil gestionar

Difícil de hacer tests

**N capas**

Objetivo primordial es la separación de las partes que componen un sistema software o también una arquitectura cliente-servidor.

**Ventajas**

Portable

Curva de aprendizaje reducida

**Desventajas**

Dificultar para diseñar las capas

Menor eficiencia

**Dirigida a eventos**

Patrón de arquitectura software que promueve la producción, detección, consumo de, y reacción a eventos. Un evento puede ser definido como "un cambio significativo en un estado".

**Ventajas**

Muy escalable

Sin dependencia entre servicios

**Desventajas**

El servidor no estará al tanto si un servicio se cayo

Sin comprobación ante la atención de eventos

**Microkernel**

También conocido como arquitectura de Plug-in, permite crear aplicaciones extensibles, mediante la cual es posible agregar nueva funcionalidad mediante la adición de pequeños plugins que extienden la funcionalidad inicial del sistema.

**Ventajas**

Buen despliegue

Buena testibilidad

Mayor rendimiento

**Desventajas**

No son fáciles de desarrollar

El Core debe estipular muy claramente cómo se desarrollarán los plugings

**Serverless**

Manera de crear y ejecutar aplicaciones y servicios sin tener que administrar infraestructura. Su aplicación continúa ejecutándose en servidores, pero terceros se encargan de toda la administración de los servidores.

**Ventajas**

Costos menores

Atención a lo importante

Menor administración de recursos

**Desventajas**

Complejas

Tiempos de carga y latencia

Muy difícil de controlar correctamente

**Requerimientos funcionales y no funcionales (Concepto de ambos y ejemplos)**

**FUNCIONALES**

Un requisito funcional define una función del sistema de software o sus componentes.

**NO FUNCIONALES**

Especifican criterios para evaluar la operación de un servicio de tecnología de información, en contraste con los requerimientos funcionales que especifican los comportamientos específicos.

