

• La importancia del Desarrollo de Software unificado

Autores:

Ing. Julio Alfonso De León Razo

Ing. Jacobo Díaz Martínez

Agenda

- Objetivo
- Introducción
- Desarrollo
 - Metodología propuesta
 - Arquitectura propuesta
 - Pruebas
- Conclusiones

Documento elaborado por el IINGEN UNAM

Documento elaborado por el IINGEN UNAM

Objetivo

- Proponer una metodología que enfatice la importancia de las buenas prácticas al desarrollar Sistemas de Software utilizando y partiendo del ciclo de vida básico de la Ingeniería de Software, así como de un framework estandarizado y de procesos Normalizados en su Desarrollo, con la finalidad inclusive, de poder integrar si es necesario diversos sistemas aún cuando se trate de plataformas diversas; así como de poder garantizar la transferencia de conocimiento e inclusive disminuir los tiempos de Desarrollo y creación de nuevos Sistemas de Información.

Documento elaborado por el IINGEN UNAM

Introducción

- El Desarrollo de Sistemas de Software es una disciplina que aún en la actualidad es poco valorada, si no es que hasta despreciada por una gran mayoría de áreas del conocimiento humano, inclusive hasta por nosotros mismos como conocedores del tema, y esto seguramente se debe a que el Software es Intangible, lo que resulta paradójico ya que en la mayoría de las disciplinas Sociales, de Salud, Económicas, Culturales, etc. Es una herramienta fundamental para la eficiencia, el proceso y control de datos. Sin embargo, derivado de lo intangible, es que aún cuando existe la Ingeniería de Software, ésta se considera como un obstáculo e inclusive como un proceso burocrático para la gestión y administración de los proyectos de Software.

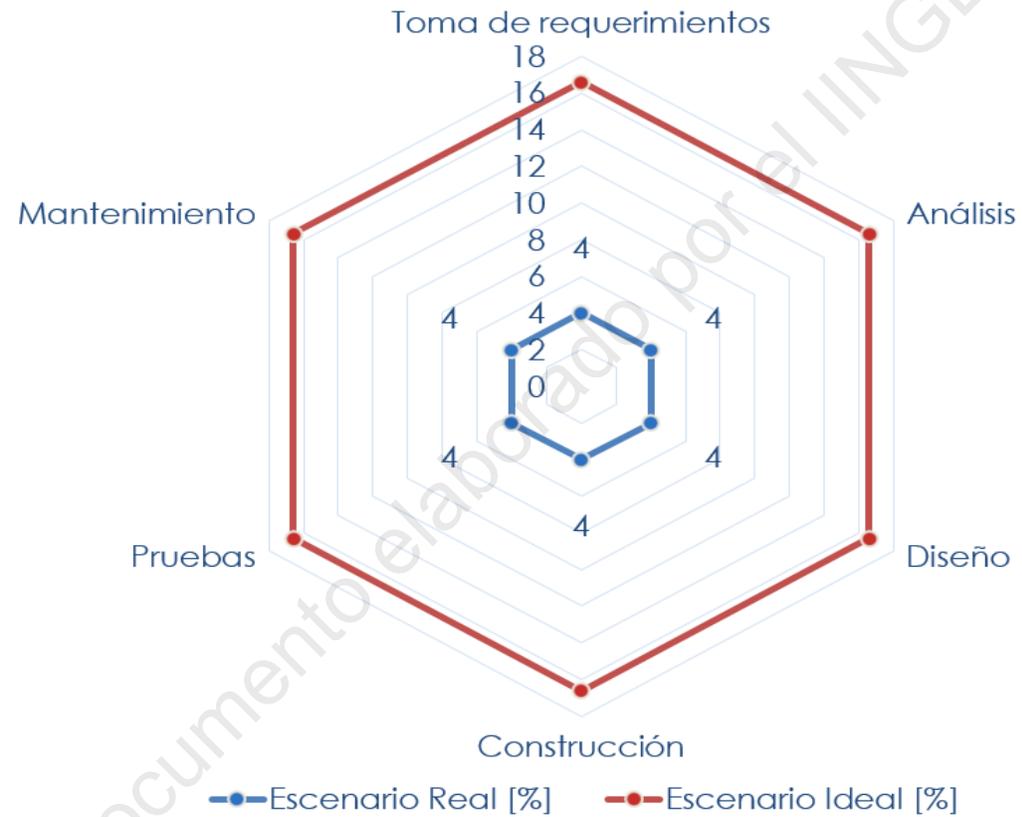
Documento elaborado por el IINGEN UNAM

Desarrollo

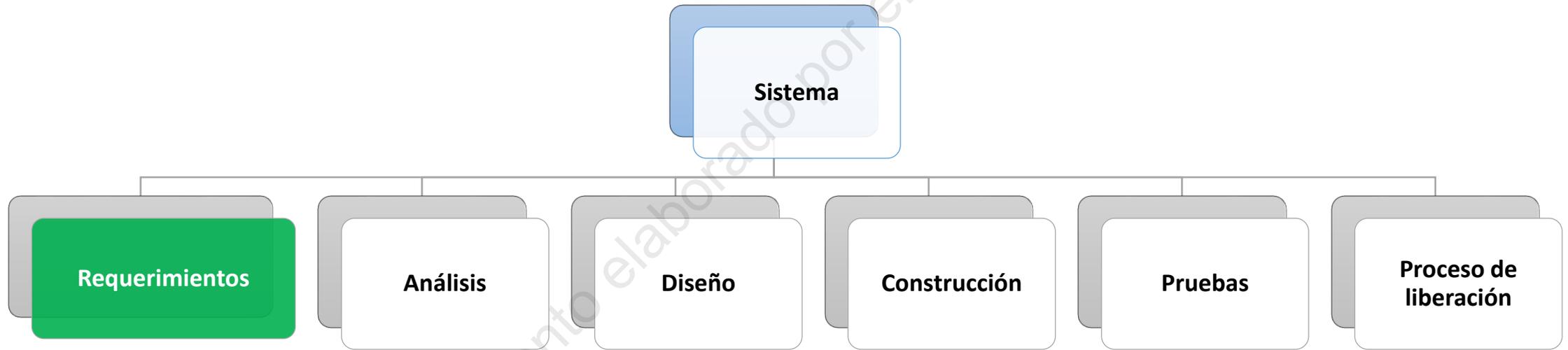
Metodología propuesta



1. Situación actual



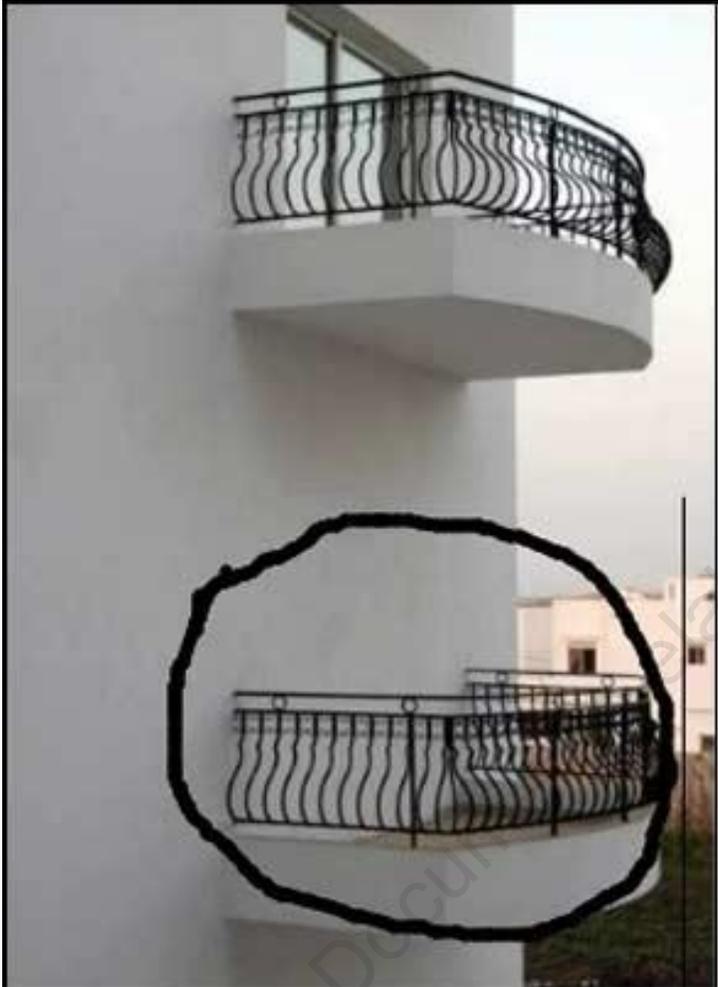
Ciclo de vida como desglose de actividades de Ingeniería de Software



2. Normalizar la toma de requerimientos



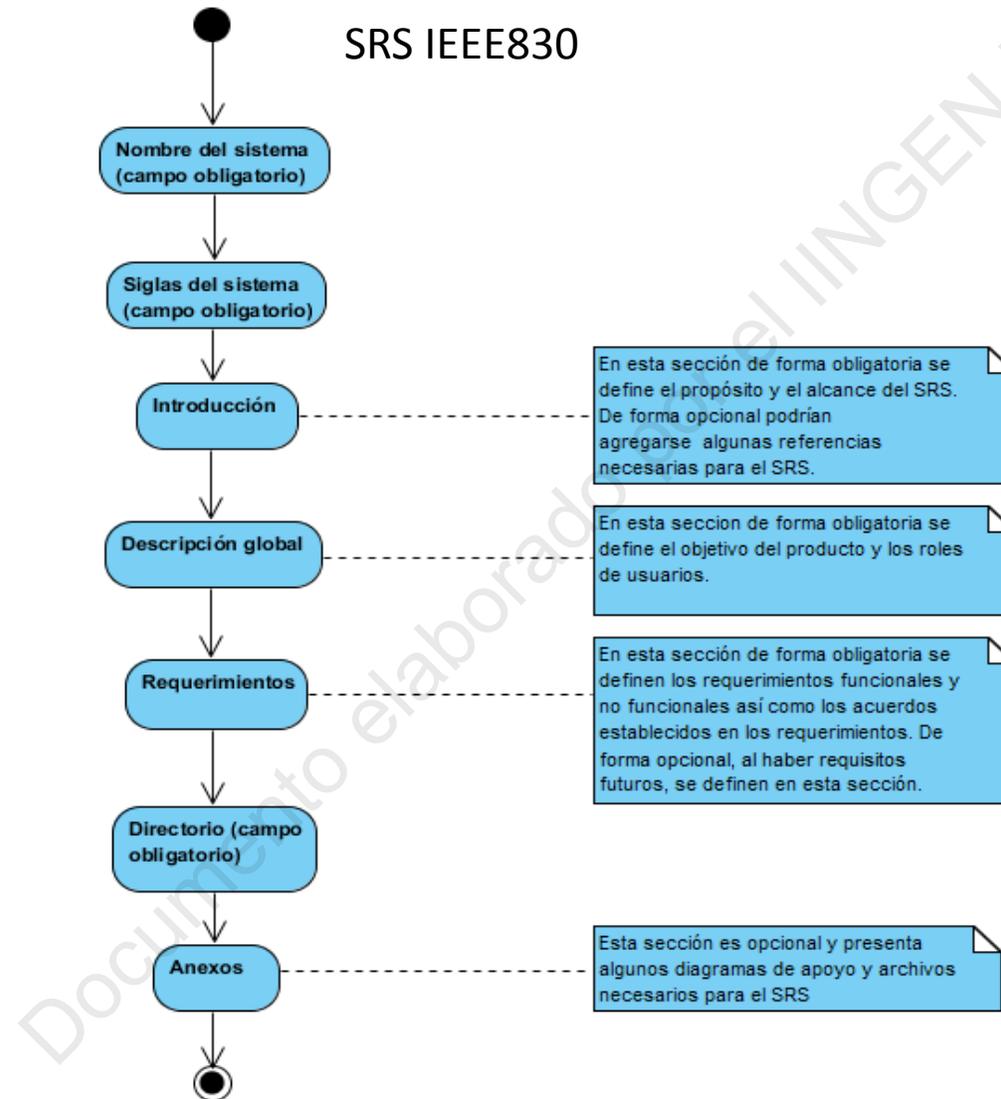
2. Normalizar la toma de requerimientos



2. Normalizar la toma de requerimientos



2. Normalizar la toma de requerimientos

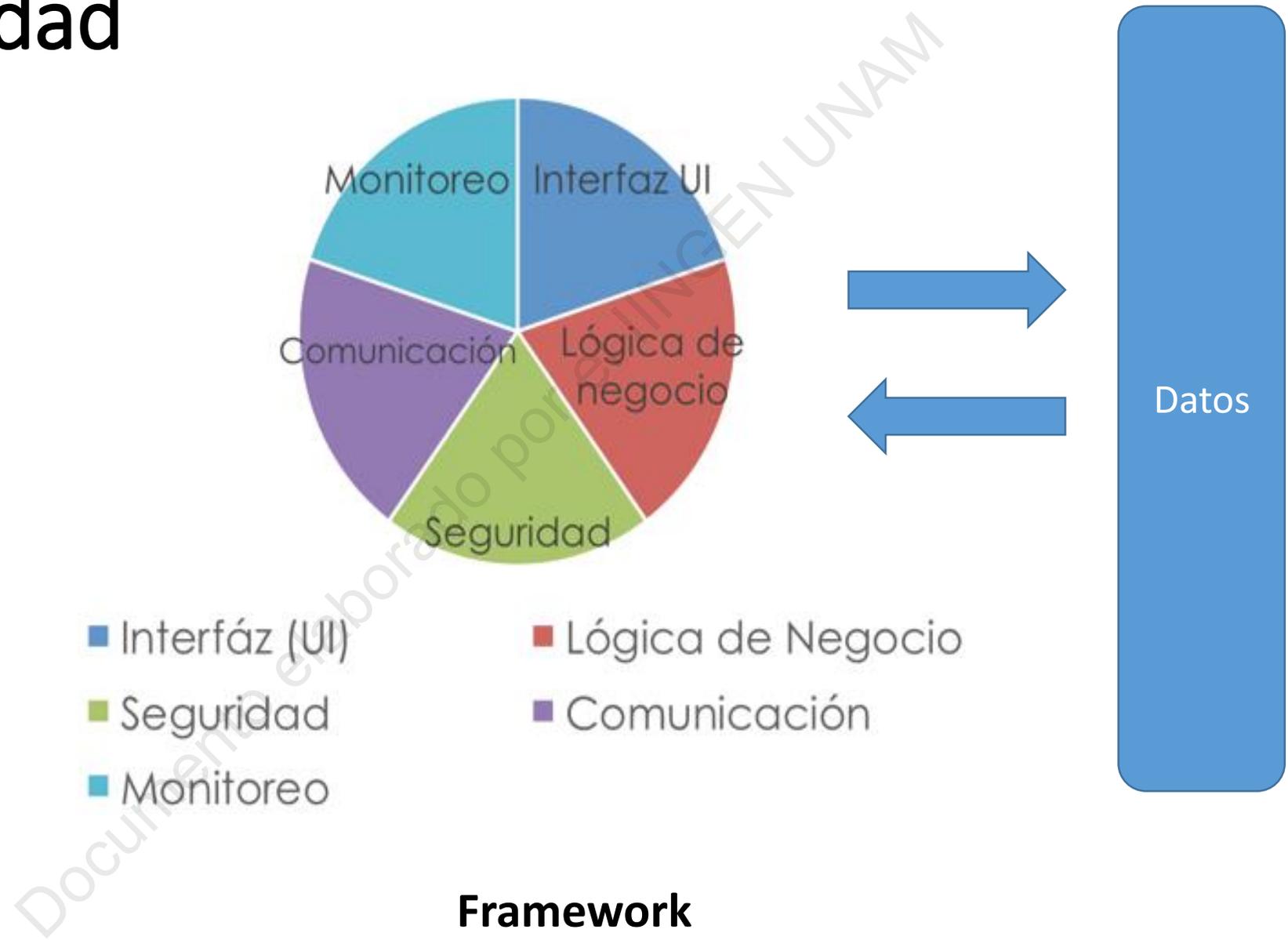


2. Normalizar la toma de requerimientos

Historia de usuario

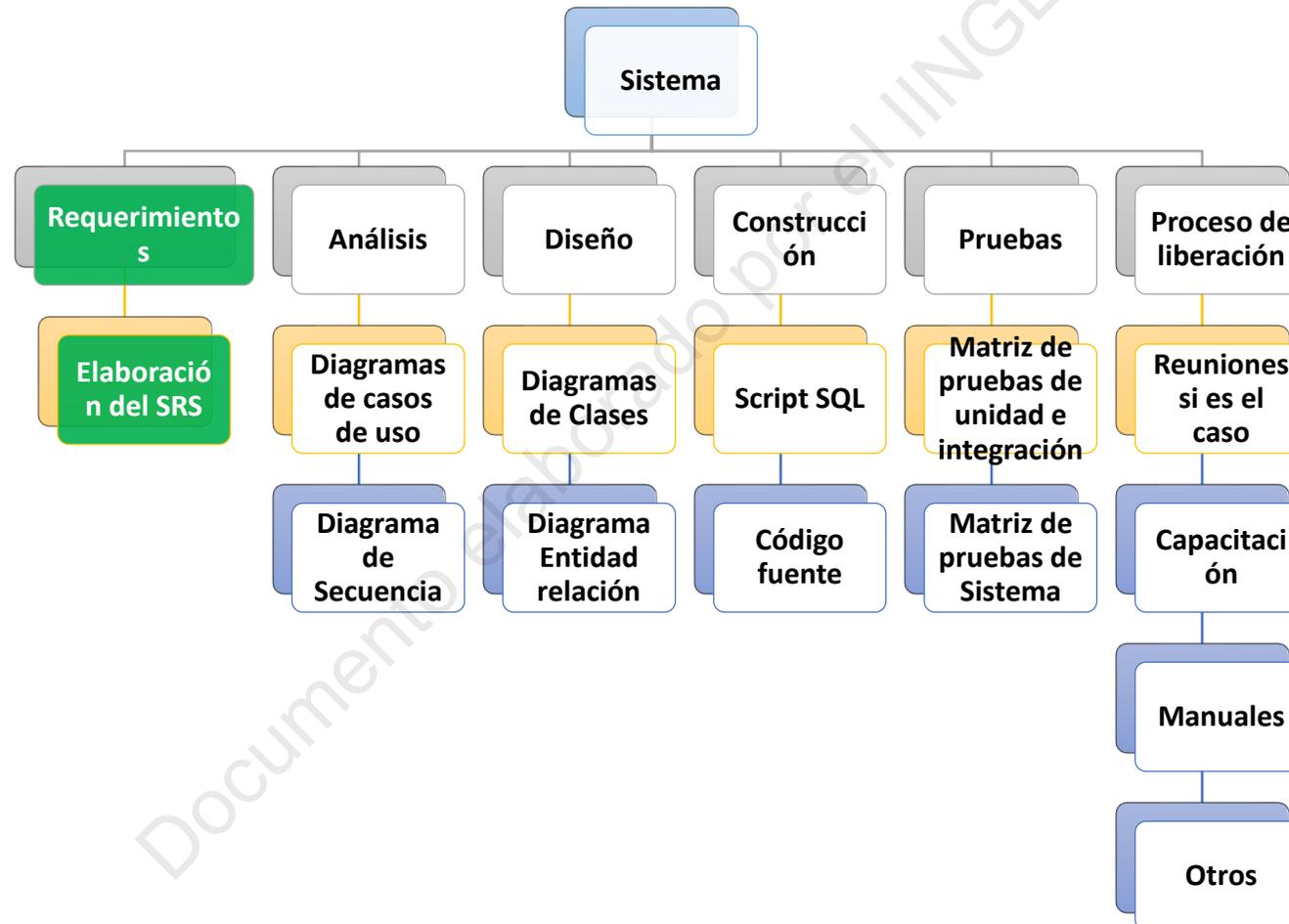
Historia de usuario	
Número: 1,2,...	
Usuario: solicitante	
Modificación de historia o requerimiento número:	
Prioridad en negocio (Alta, Media, Baja):	
Riesgo en desarrollo (Alto, Medio, Bajo):	
Descripción:	
Observaciones:	

3. Trazabilidad

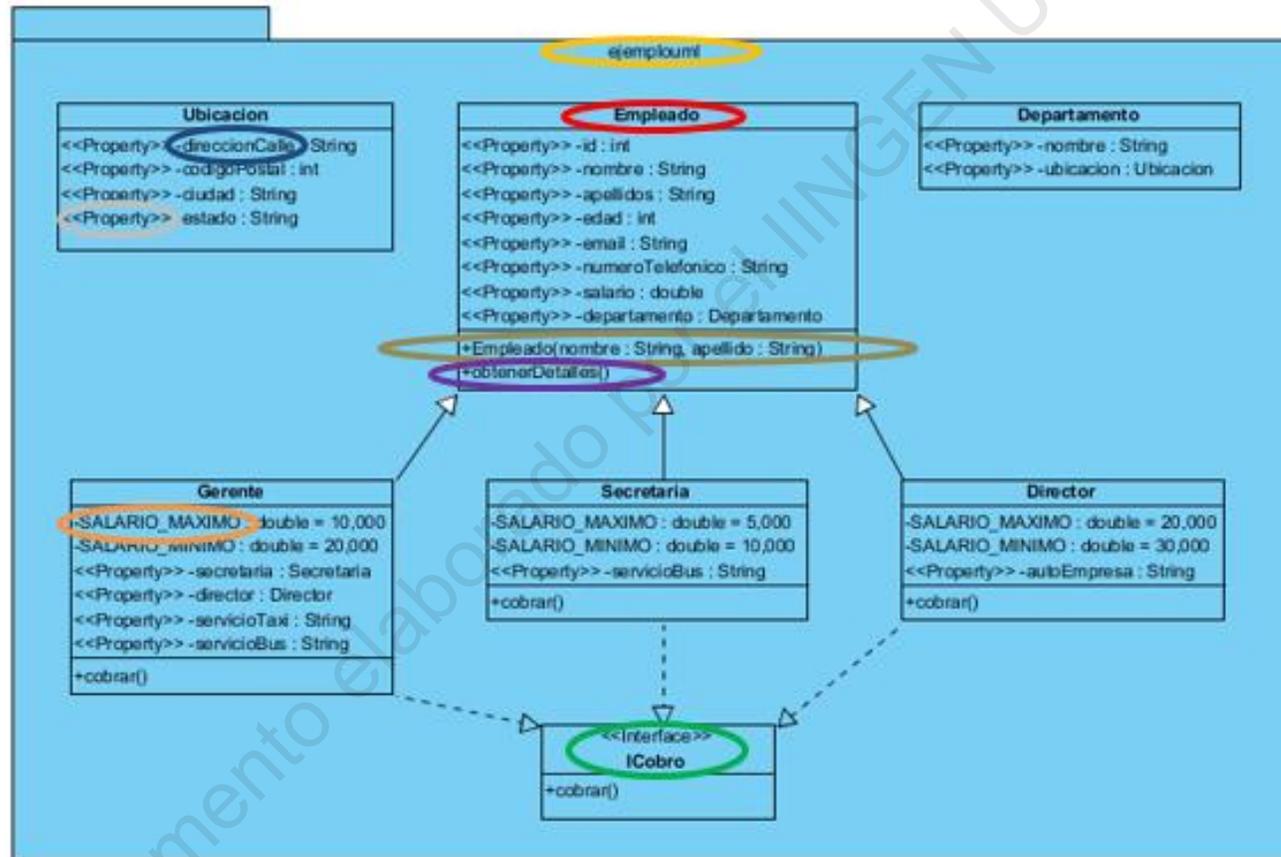


4. Documentación necesaria

Garantizar transferencia de conocimiento

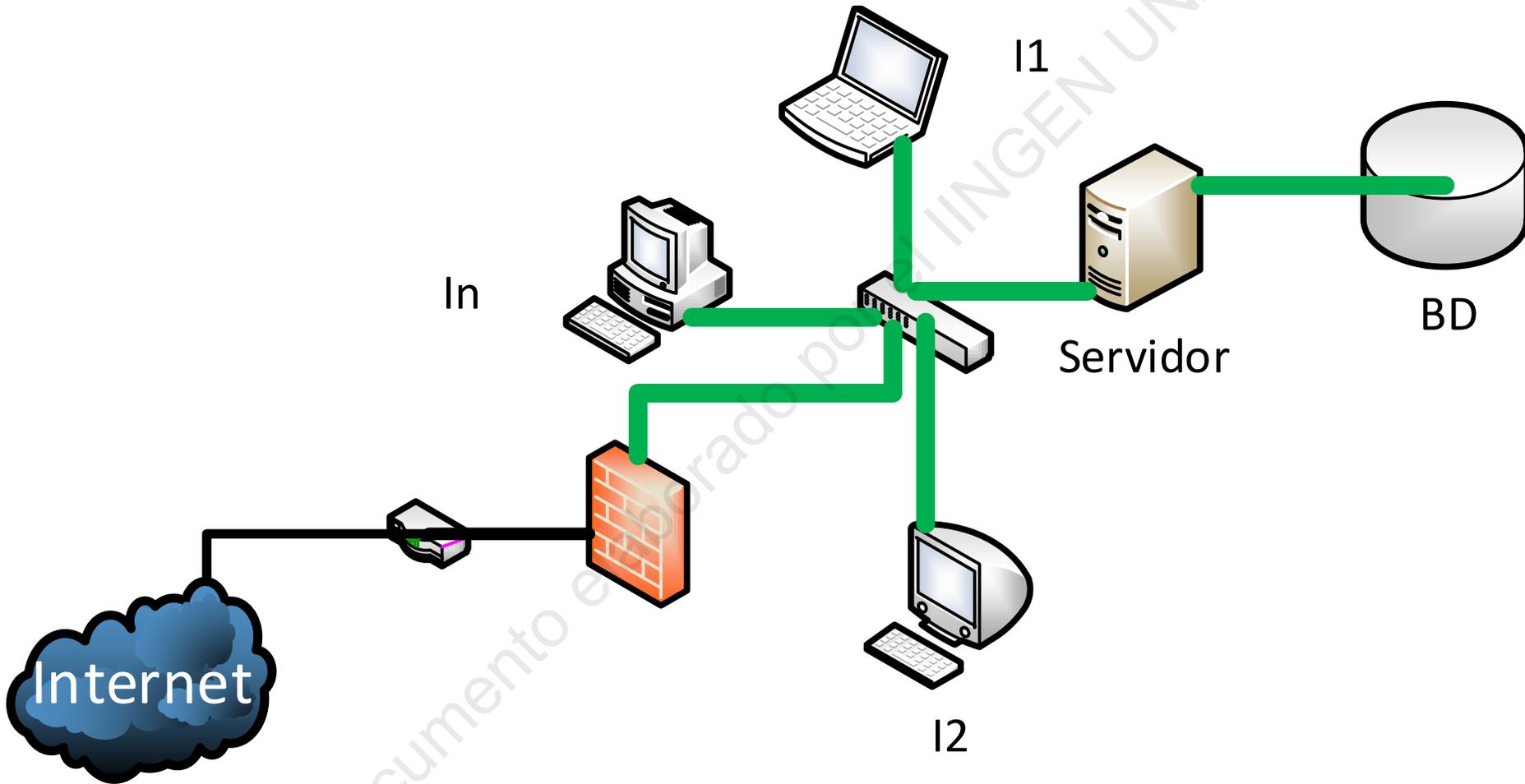


4.1 Normalizar estilos de código y nomenclatura

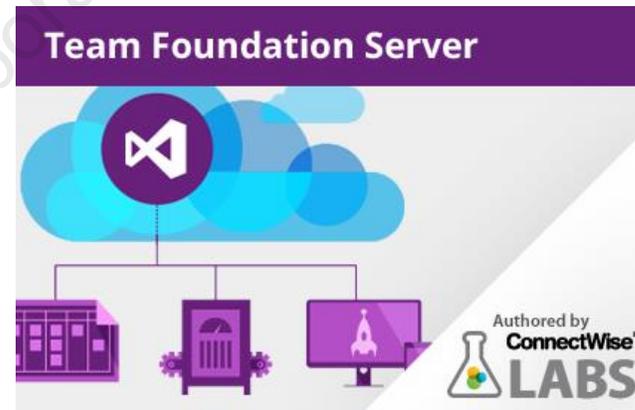


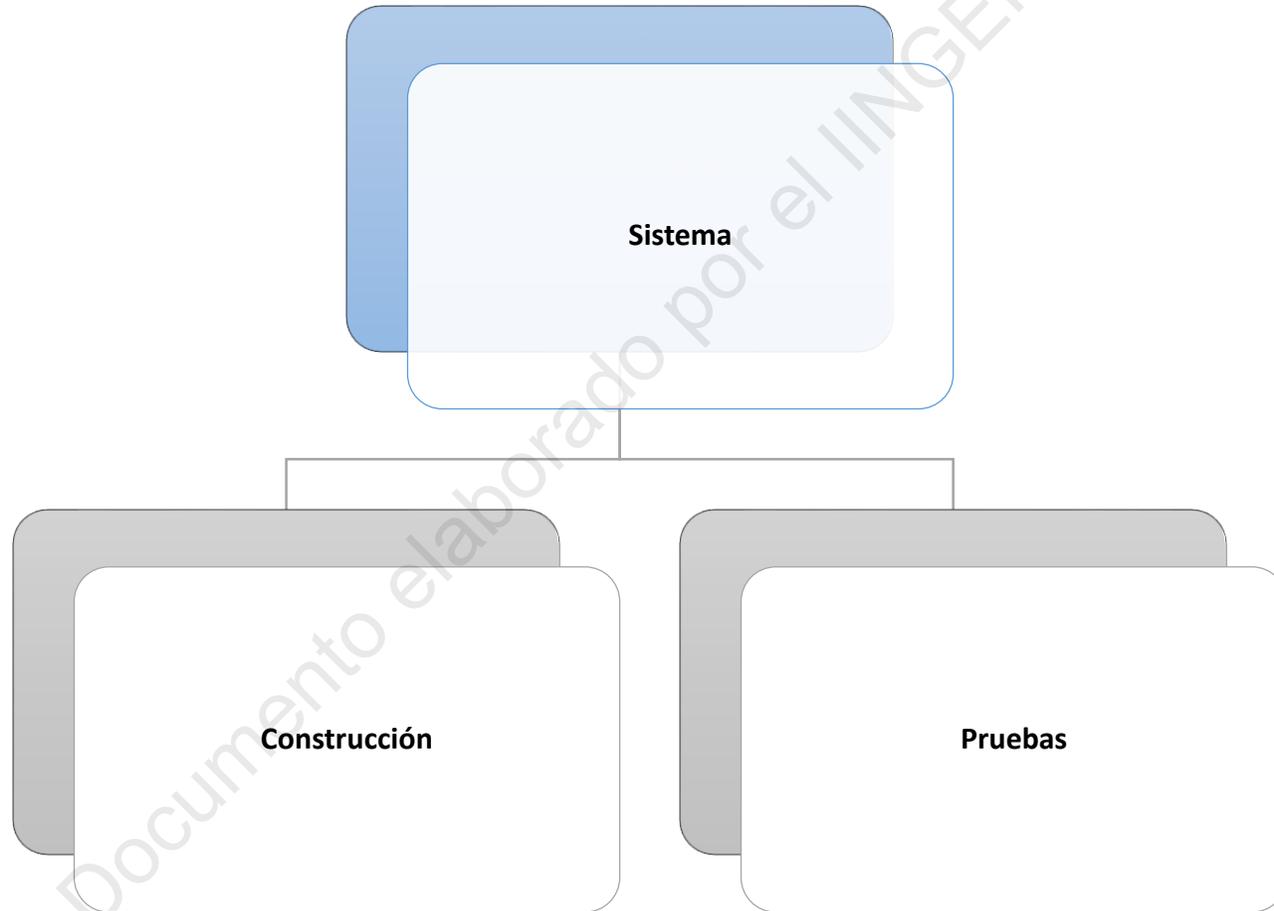
- Clase
- Paquete
- Propiedad
- Atributo (variable)
- Método
- Constante
- Constructor
- Interface

5. Repositorio



Algunos repositorios



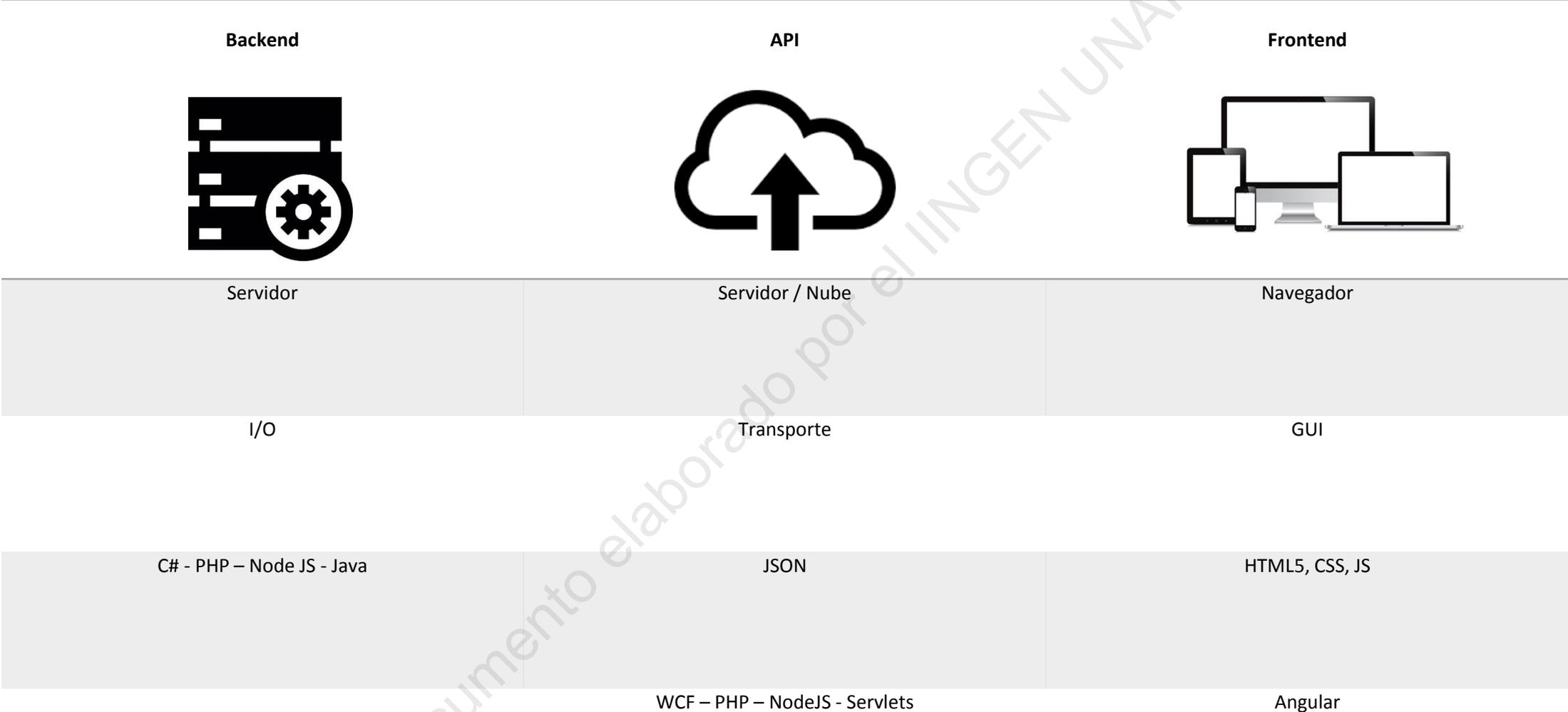


Arquitectura propuesta

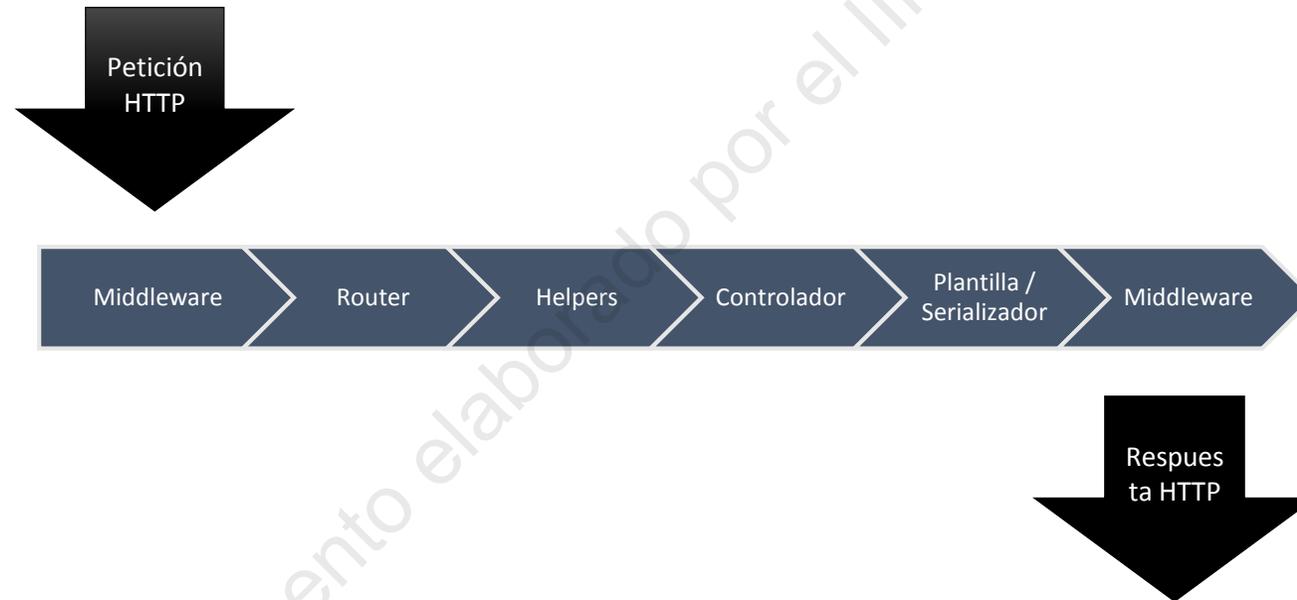
- Arquitectura de la aplicación
- Organización del código Back end
- Organización del código Front end
- API de comunicación
- Tecnología

Documento elaborado por el INGEN UNAM

Arquitectura



Back end



Back end

- Middleware - Procesa las peticiones entrantes y respuestas. (sessionMiddleware, cacheMiddleware)
- Router – Mapea las URL a sus controladores, contener rutas con parámetros por defecto, uso de expresiones regulares.
- Decorators / Helpers – Envolver las funciones del controlador con lógica adicional, pueden modificar peticiones de objetos o direccionar (ejemplo, login_requerido, permisos_requeridos, cache_page, seguridad)
- Controlador – Lógica de negocio, su función es recibir peticiones HTTP y dar respuestas HTTP, se comunica con la capa de modelo, sesión, etc.
- Modelo(ORM / SP) – Capa de abstracción de la base de datos, convierte datarows en objetos / Procedimientos almacenados / híbrido
- Template – Genera HTML, Convierte objetos en cadenas para el intercambio de datos (JSON)

Frontend



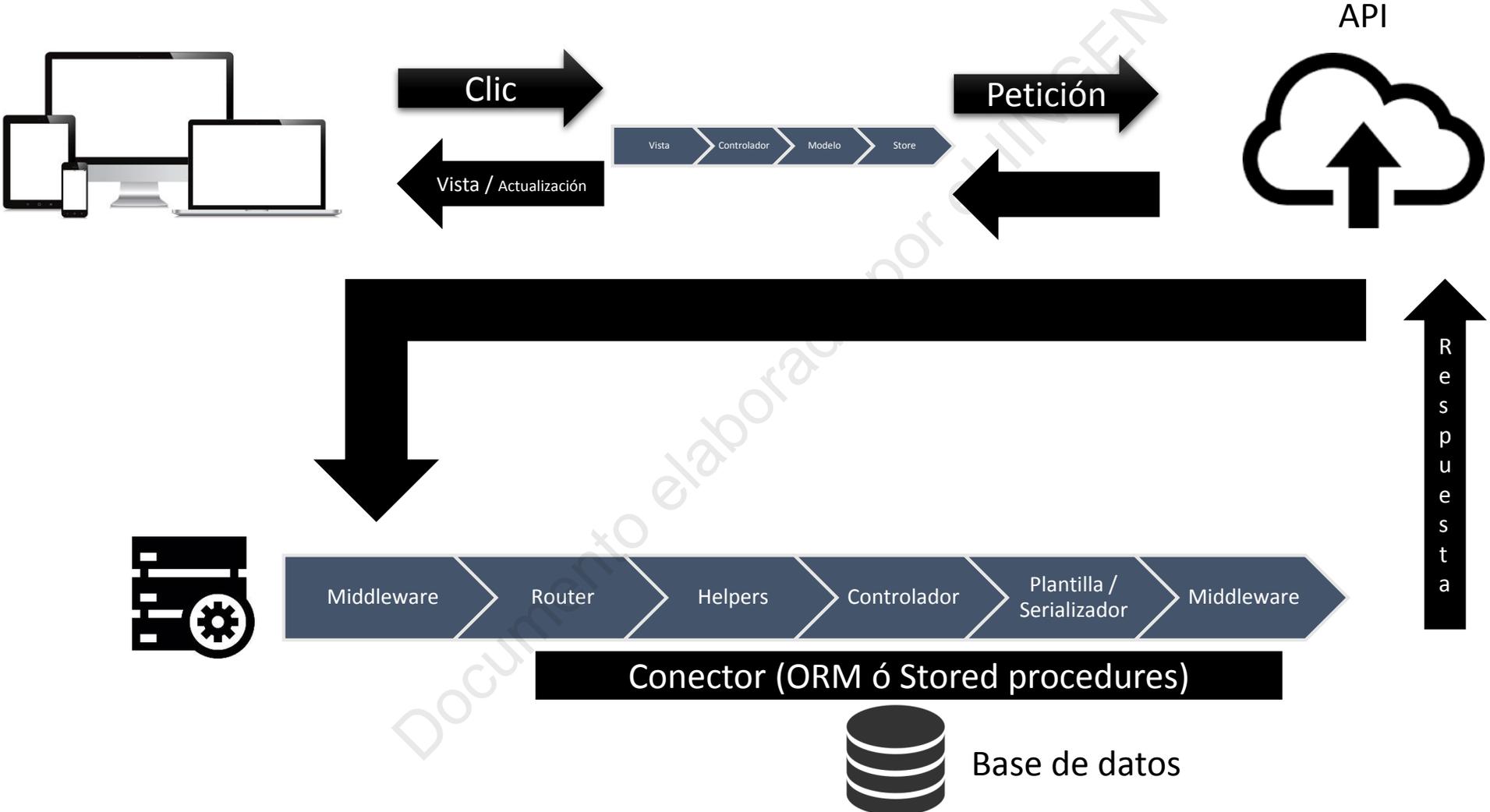
Vista – Establece elementos GUI visibles, enlaza elementos de la vista con datos

Controlador – Lógica de negocio, Escuchador de eventos, interactúa con el modelo, procesa respuestas, actualiza la vista

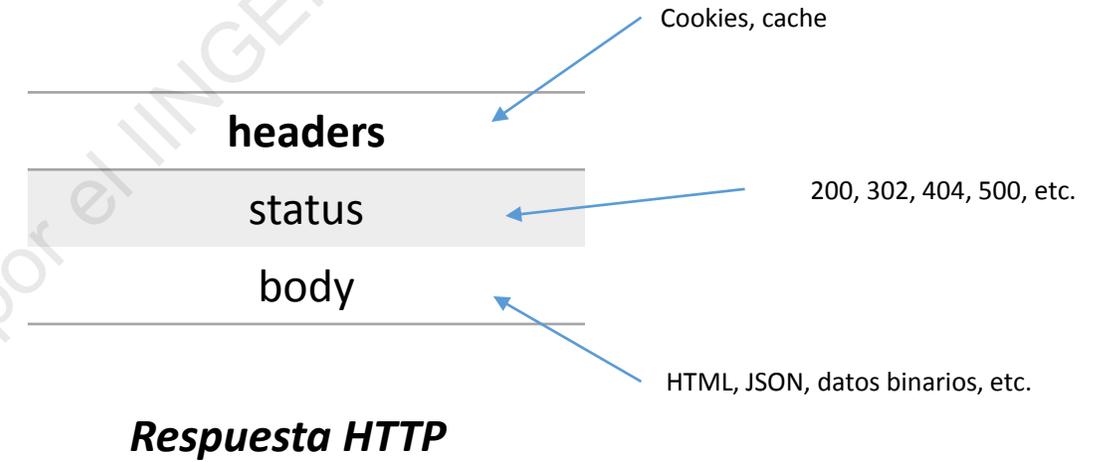
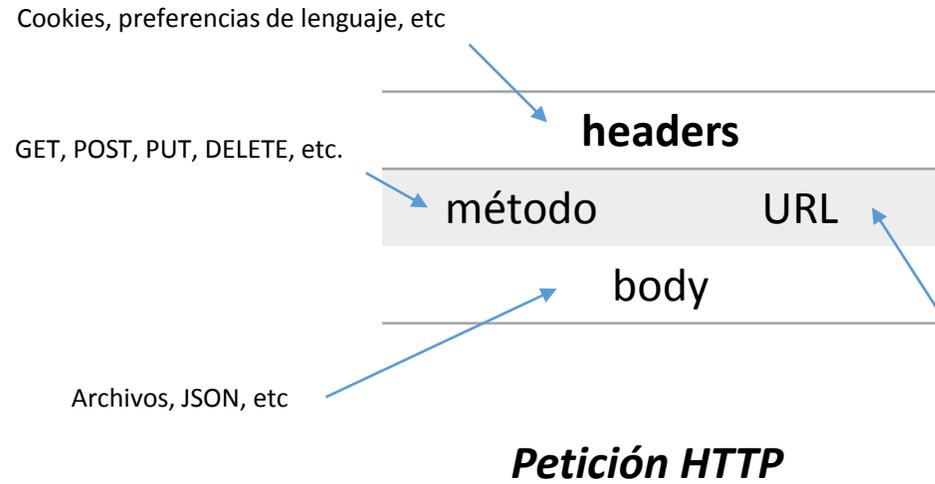
Modelo – Representa datos de la API en objetos javascript, representación del lado del cliente de los modelos del lado del servidor, realiza validación de datos.

Store – Se comunica con la API, permite el enlace de datos a la interfaz de usuario, convierte las respuestas de la API en instancias del modelo

Arquitectura de la aplicación

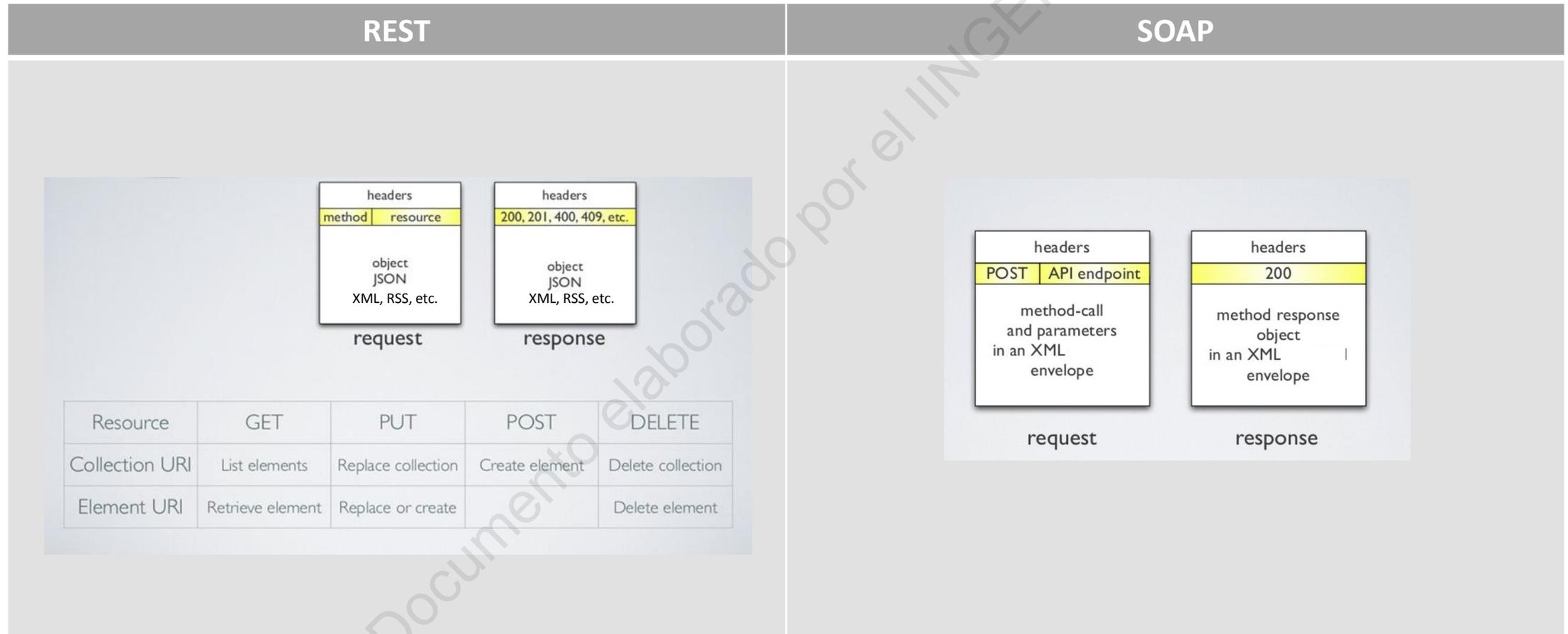


Diseño de la API



Protocolo, host, puerto, recurso, parámetros

Diseño de la API (Rest o SOAP)



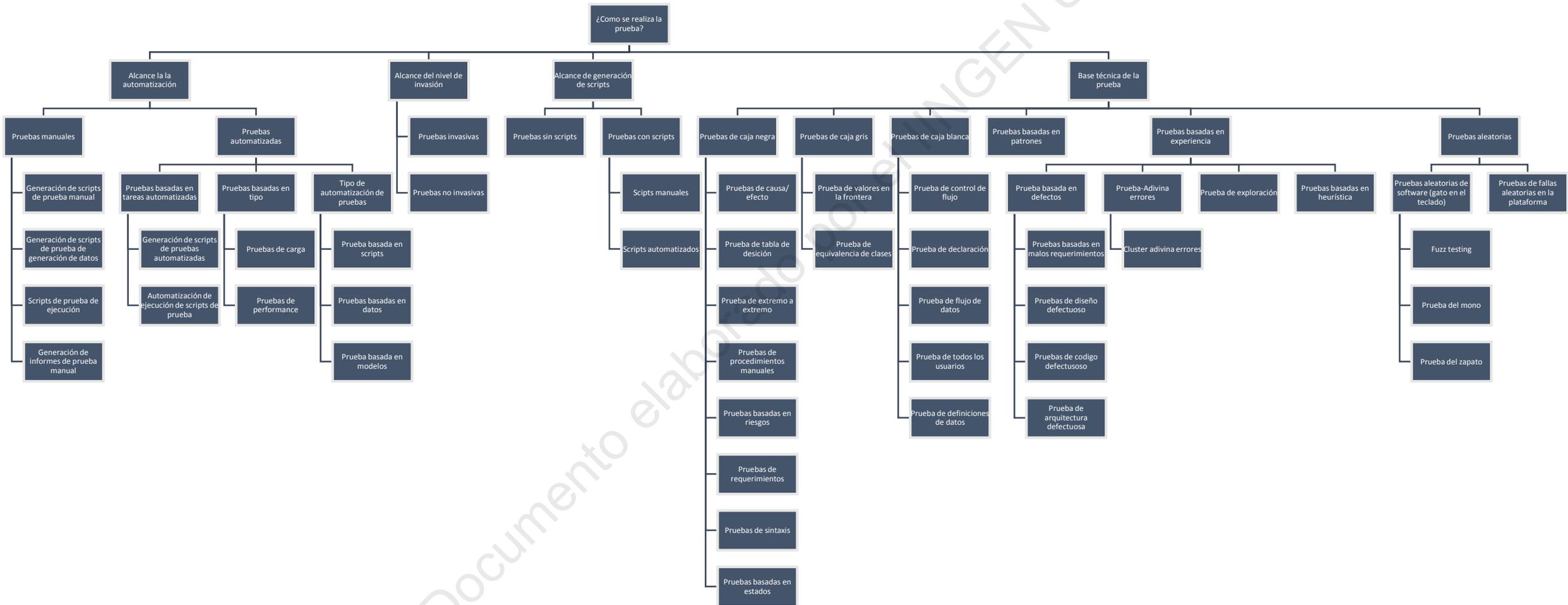
Herramientas (entorno multiplataforma)

Front end	Back end	Bases de datos
 <p>HTML5, JavaScript, CSS3, Angular 2.0</p>	 <p>Node.js, PHP, Redis</p>	 <p>MySQL, Oracle, MongoDB</p>

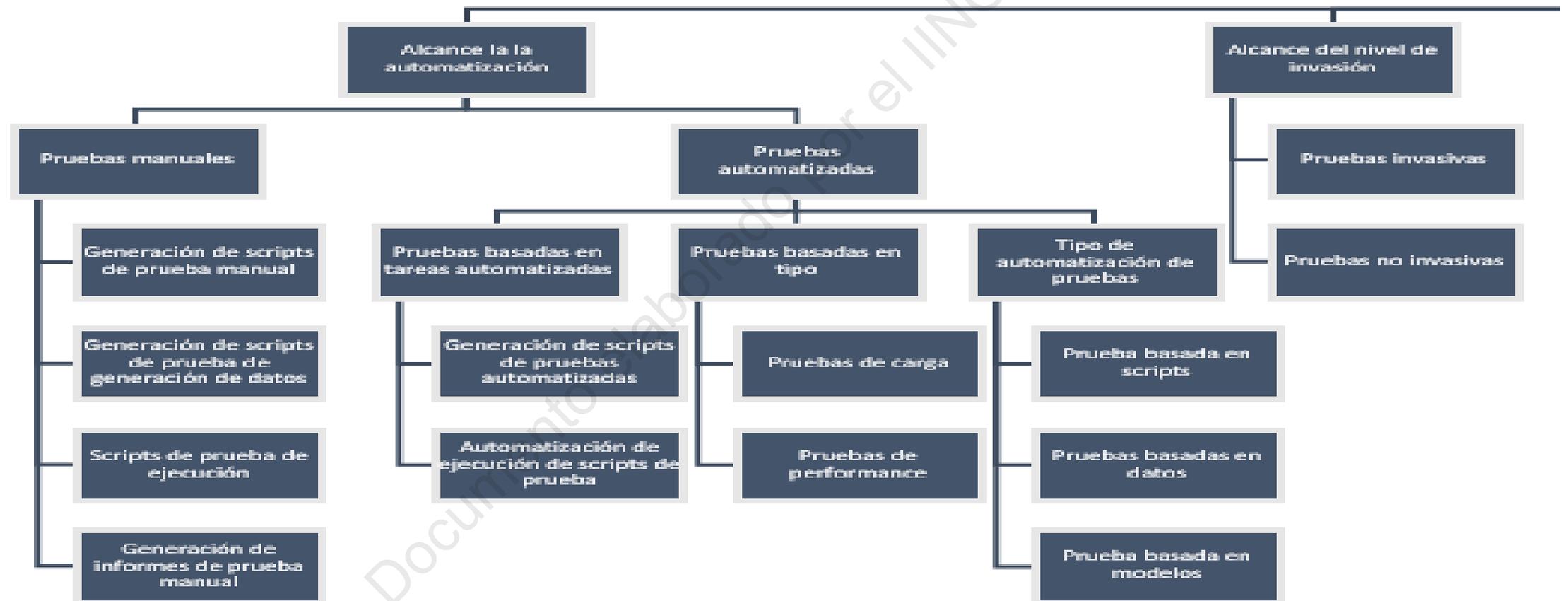
Pruebas

- Existe un gran número de pruebas que pueden aplicarse a un sistema. Algunos somos conscientes de una minoría de ellas.
- Para determinar qué pruebas son útiles a mi sistema existe una serie de preguntas que deben ser respondidas para obtener una guía de pruebas a utilizar.
- ¿Cómo se realiza la prueba?
- ¿Qué tan bien desempeña sus funciones el objeto bajo prueba?
- ¿Qué está siendo probado?
- ¿Cuándo se realiza la prueba?
- ¿Dónde se realiza la prueba ?
- ¿Qué o quien está realizando la prueba ?
- ¿Porque se realiza la prueba?

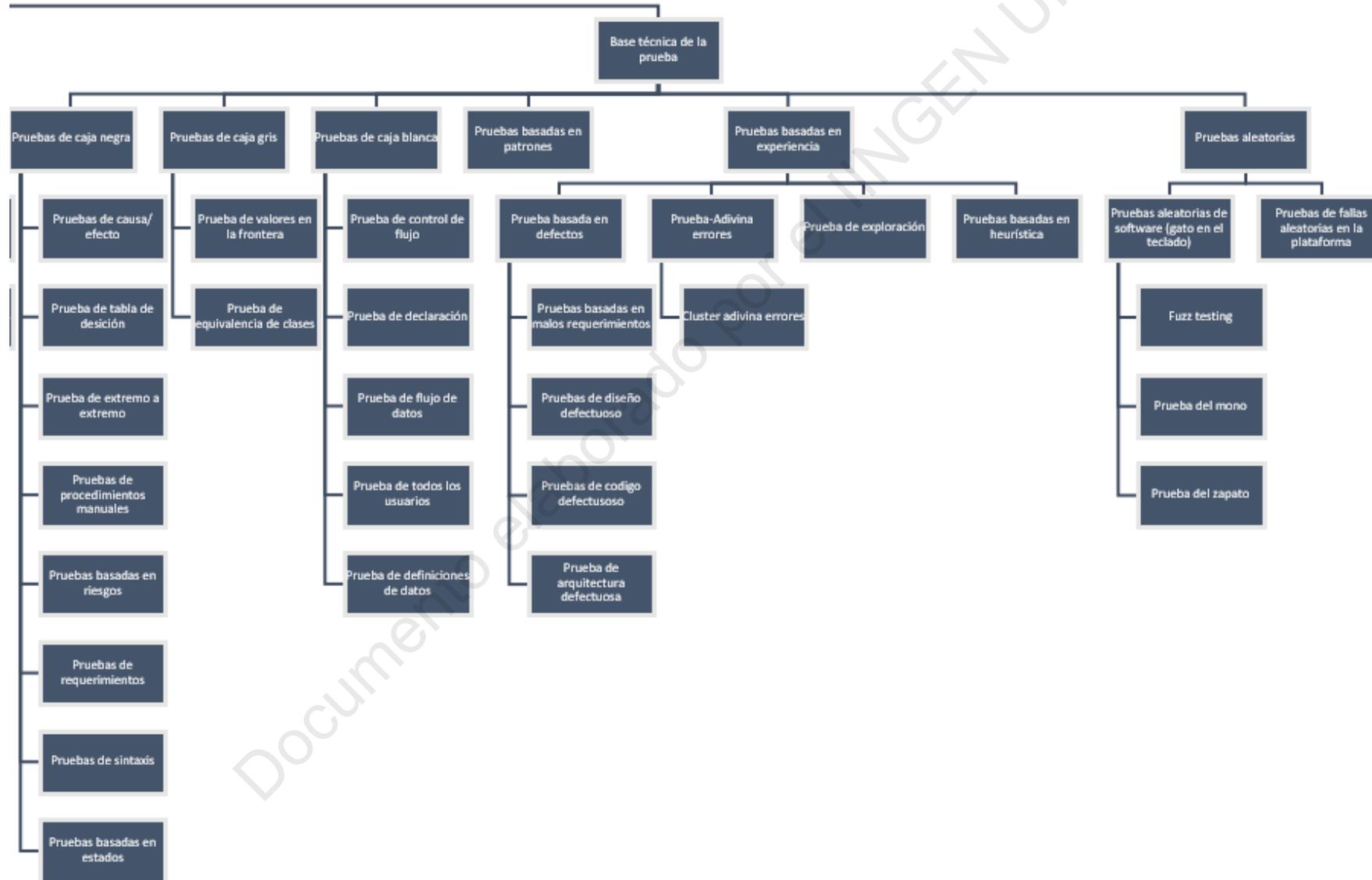
¿Cómo se realiza la prueba ?



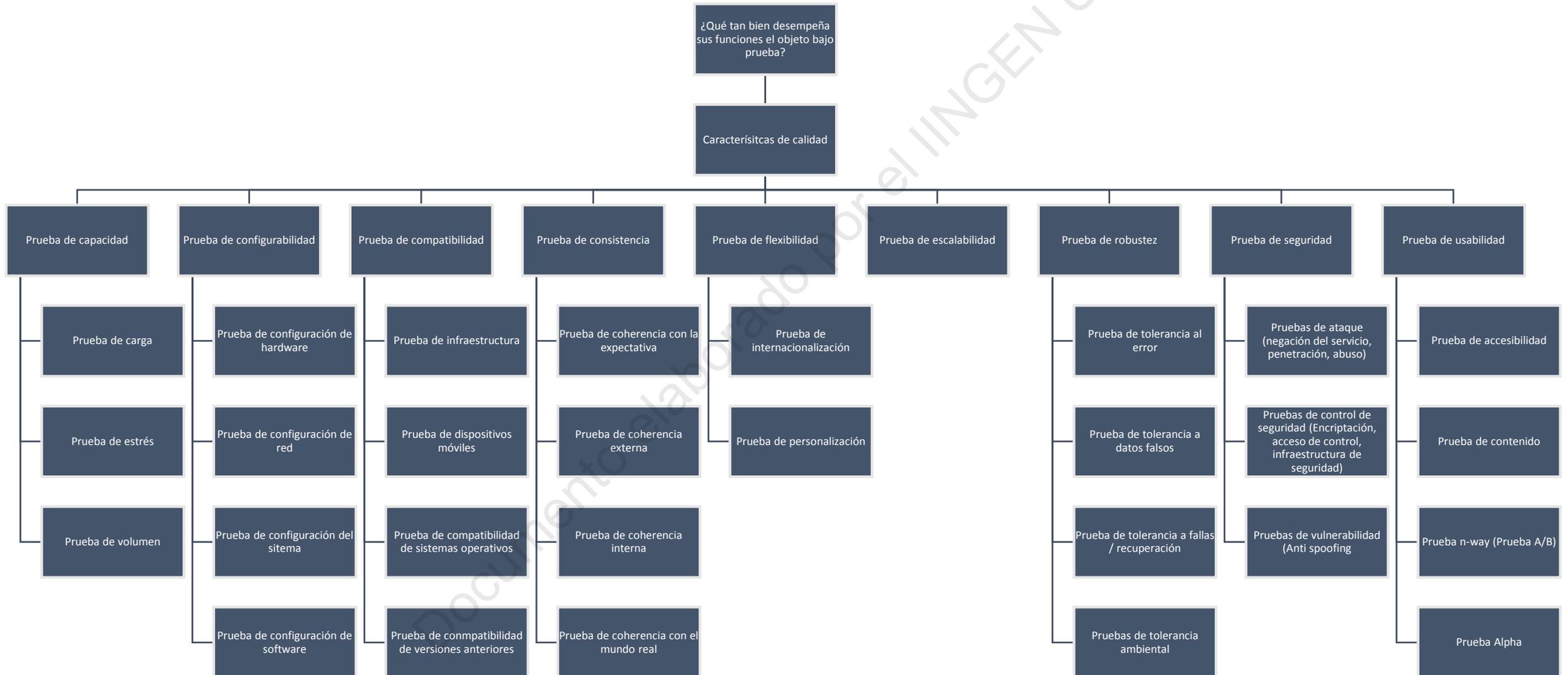
¿Cómo se realiza la prueba ?



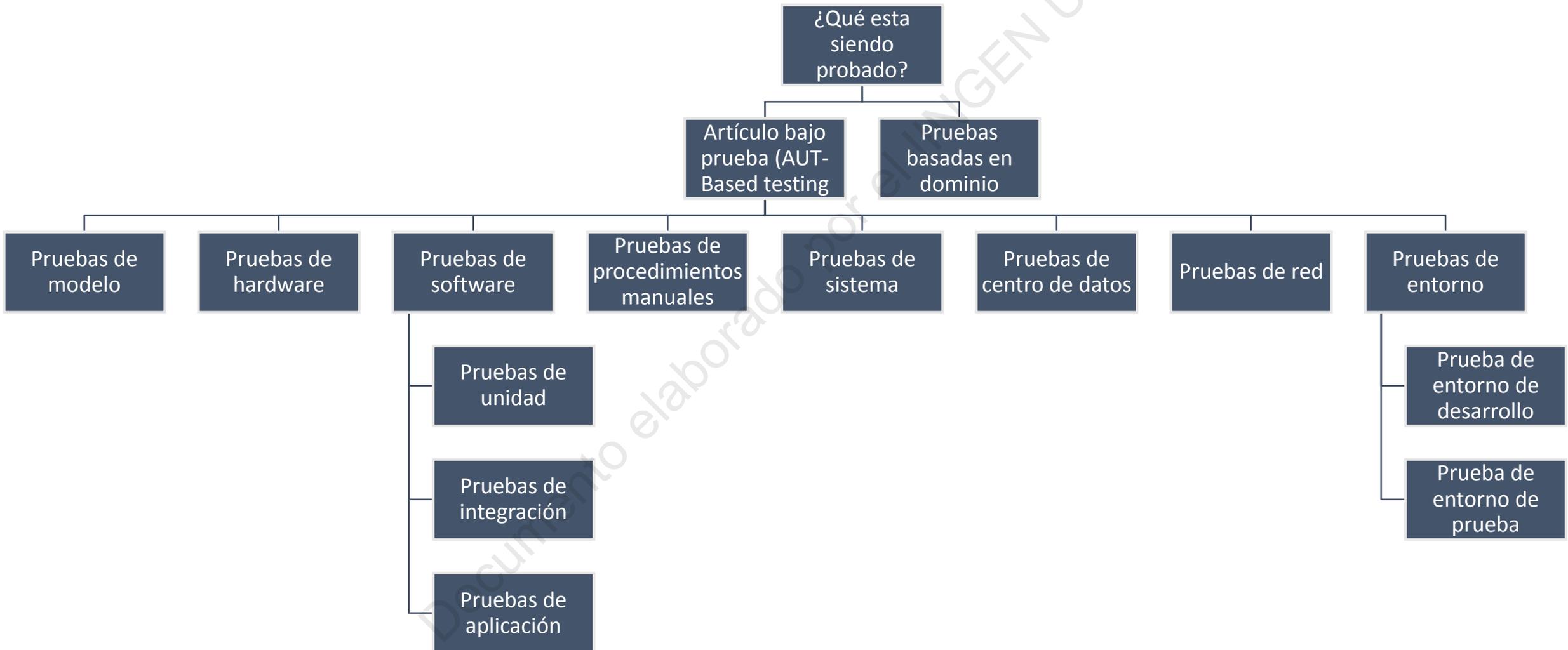
¿Cómo se realiza la prueba ?



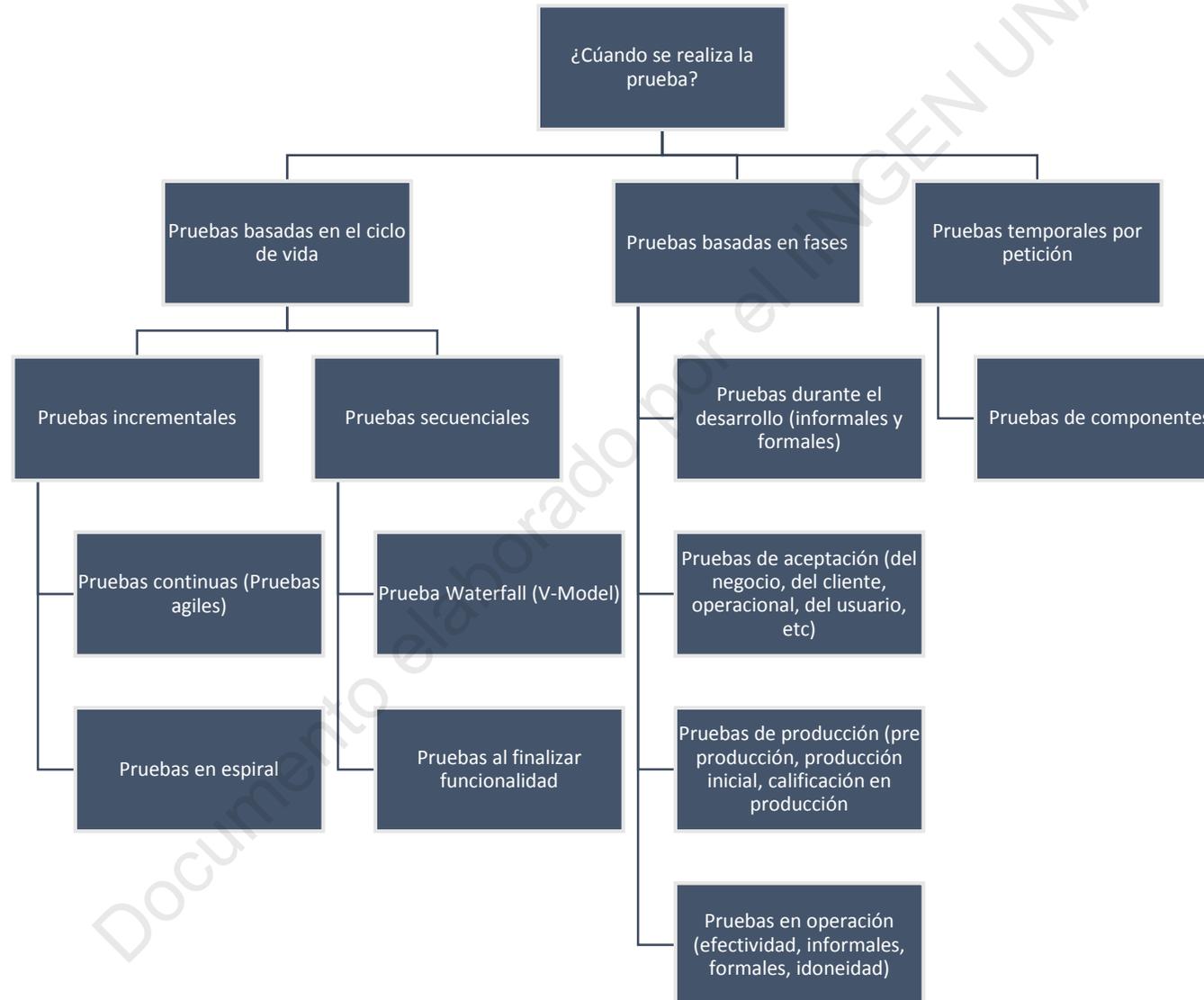
¿Qué tan bien desempeña sus funciones el objeto bajo prueba?



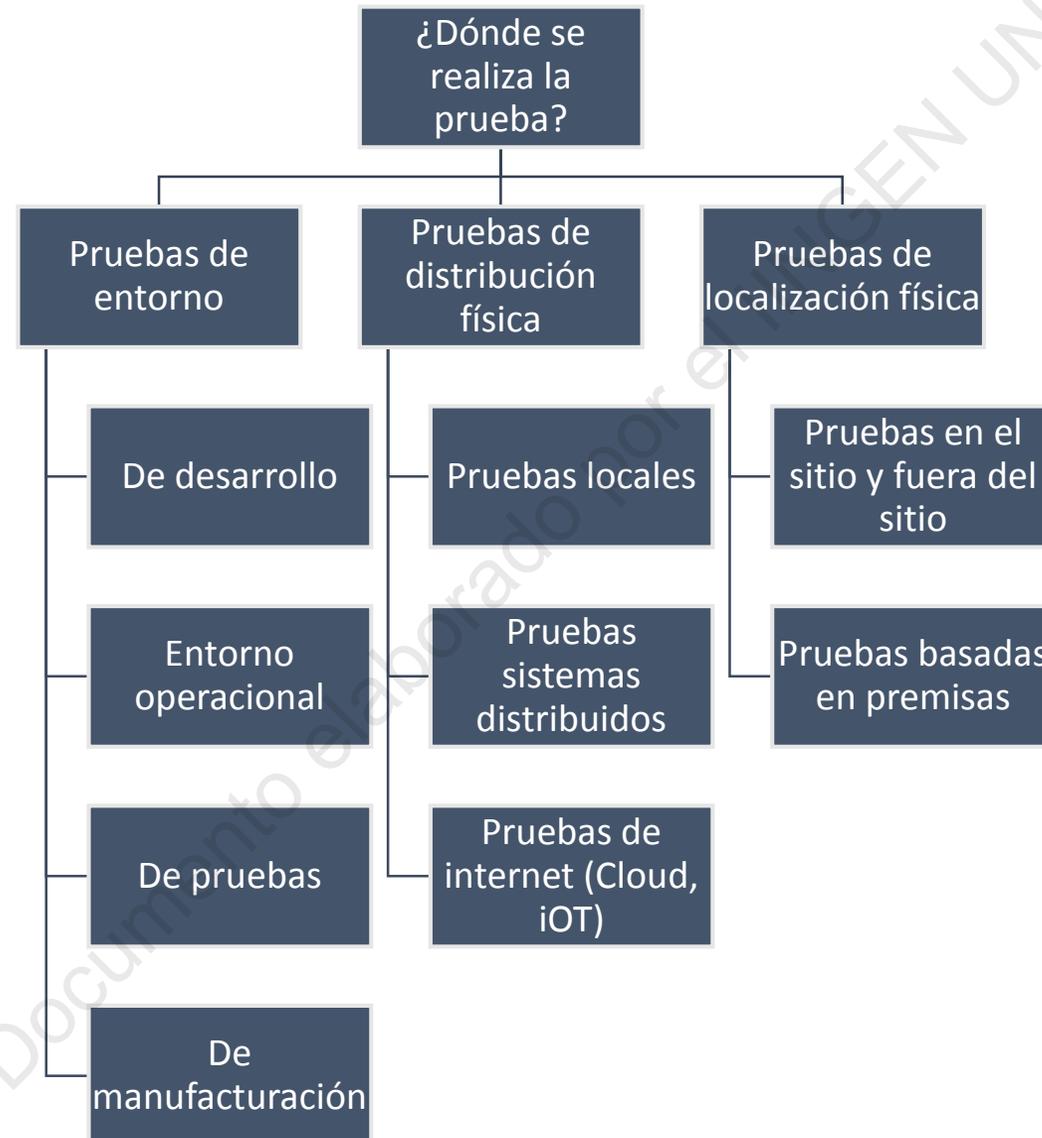
¿Qué está siendo probado ?



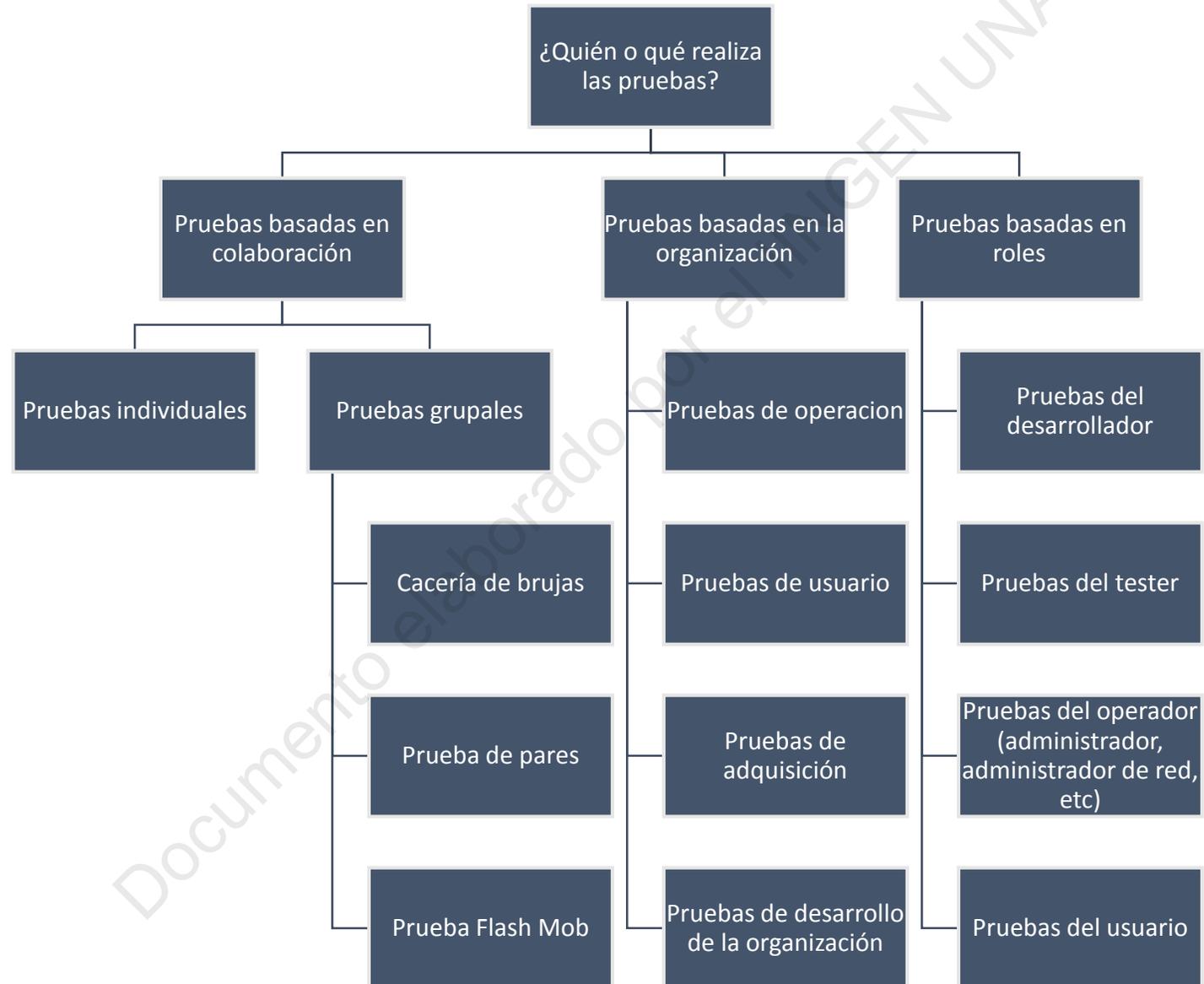
¿Cuándo se realiza la prueba ?



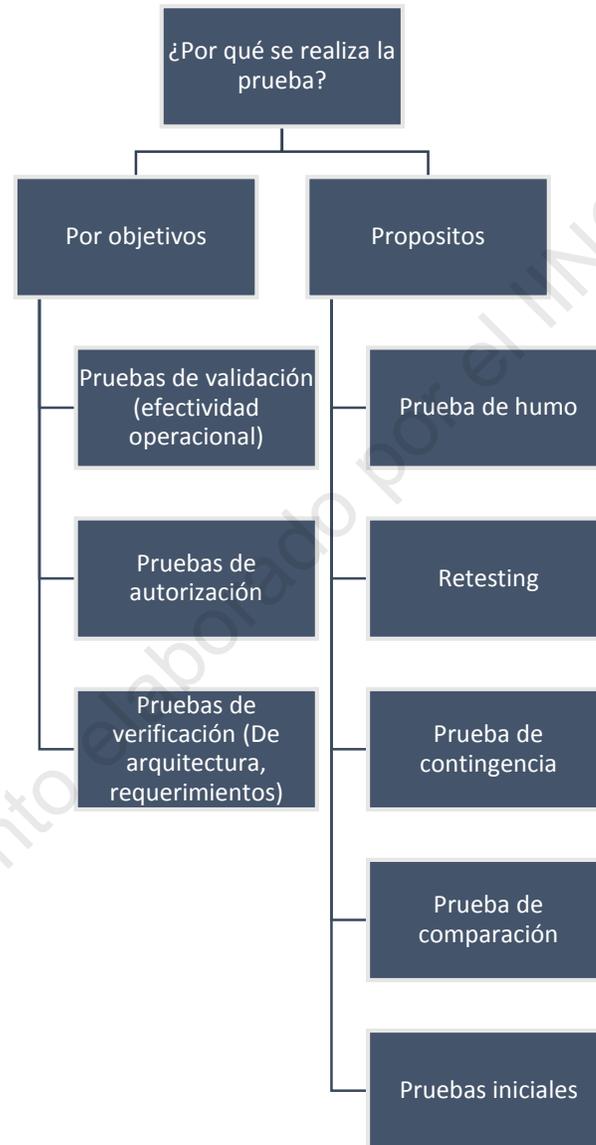
¿Dónde se realiza la prueba ?



¿Qué o quien está realizando la prueba ?



¿Por qué se realiza la prueba ?



- Los sistemas requieren diferentes tipos de pruebas. Las pruebas se pueden organizar en una taxonomía de las preguntas expuestas, de acuerdo a sus respuestas, puede ser más fácil determinar que pruebas son necesarias gracias a esta organización.



Documento elaborado por el IINGEN UNAM

Conclusiones

- Metodología Unificada
- Mejores prácticas
- Integración de Sistemas
- Disposición de Información
- Mejora continua
- Compartir conocimiento



Contacto

- Ing. Julio Alfonso De León Razo
 - Correo: jdeleonr@iingen.unam.mx
 - Tel: 56233600 ext.8893
- Ing. Jacobo Díaz Martínez
 - Correo: jdiazm@iingen.unam.mx
 - Tel: 56233600 ext.8893