Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Programa de ingeniería de sistemas
Ingeniería de Software II Grupos A y B
Laboratorio de Ingeniería de Software II Grupos A, B y C
W. Libardo Pantoja Y, Brayan Daniel Perdomo
2025.2

Ingeniería de Software II

Teoría y Laboratorio

Proyecto de clase 2025.2: Gestión del Proceso de Trabajo de Grado – FIET

Fecha de creación: Agosto 4 de 2025
Última modificación: Agosto 4 de 2025

Descripción del problema

Actualmente, el proceso de trabajo de grado en la Facultad de Ingeniería Electrónica y Telecomunicaciones (FIET) de la Universidad del Cauca se gestiona principalmente mediante formatos PDF, correos electrónicos y validaciones manuales. Esto genera dificultades como:

- Desorganización o pérdida de versiones de los formatos (Formato A, anteproyecto, actas de sustentación, etc.)
- Retrasos en la comunicación entre estudiantes, directores, jurados y secretaría.
- Falta de trazabilidad sobre en qué etapa se encuentra cada trabajo de grado.
- Ausencia de un historial sistematizado y consultas en línea del avance por parte de las autoridades académicas.

Lo anterior impacta negativamente en la eficiencia, transparencia y seguimiento del proceso de titulación de los estudiantes.

El coordinador del Programa de Ingeniería de Sistemas, Dr. Julio Ariel Hurtado, necesita un sistema software que automatice y facilite el seguimiento del proceso de trabajo de grado del programa de Ingeniería de Sistemas (A futuro se usará para todos los programas de FIET), desde la presentación del Formato A hasta la sustentación y emisión del paz y salvo académico, para las modalidades de Práctica Profesional y Trabajo de Investigación.

La siguiente carpeta contiene la reglamentación y los formatos de las distintas modalidades de trabajo de grado de la FIET: enlace.

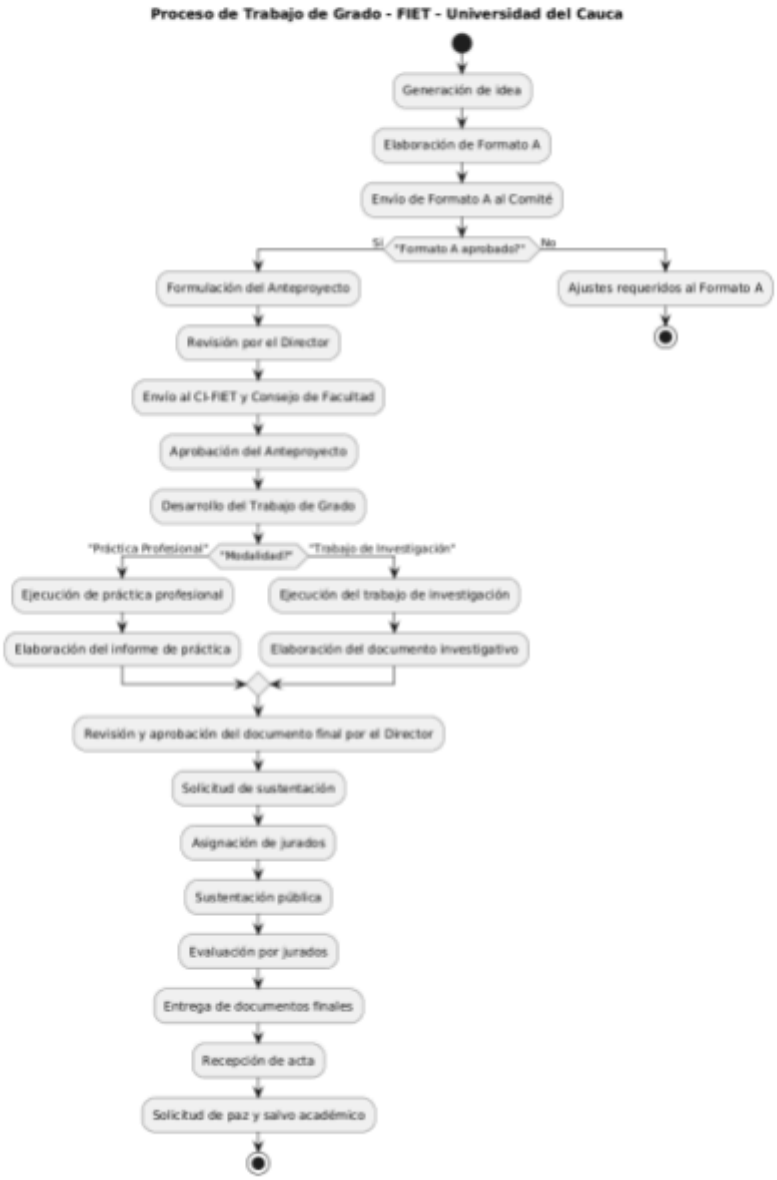
Requisitos

Requerimientos funcionales

1. **Registro y autenticación de usuarios** (estudiantes, directores, jurados, secretaría, comité).
2. **Presentación del Formato A** por parte del estudiante.
3. **Revisión y aprobación del Formato A** por parte del Comité de cada Programa.
4. **Elaboración y envío del anteproyecto** al jefe de cada departamento, incluyendo subida de documentos con la selección de modalidad: Práctica Profesional o Investigación..
5. **Asignación de los dos evaluadores** por parte de la jefe de departamento o persona delegada.
6. **Evaluación del anteproyecto** por parte de dos evaluadores.
7. **Aprobación y Resolución** del anteproyecto por el Consejo de Facultad.
8. **Gestión del desarrollo del trabajo de grado**, con registro de avances y observaciones.
9. **Subida del documento final**.
10. **Solicitud y programación de sustentación**.
11. **Asignación de jurados** por parte de secretaría de la decanatura.
12. **Registro de calificación y acta de sustentación**.
13. **Generación de historial del proceso** para cada estudiante.
14. **Notificaciones y alertas automáticas** por correo o dentro del sistema.
15. **Panel de seguimiento** para coordinador o secretaría.
16. **Exportación de documentos** (actas, paz y salvo, etc.) en formato PDF.
17. **Repositorio de necesidades de las empresas** para ser trabajados como proyectos de grado.

La Figura 1, muestra el proceso de negocio de los trabajos de grado.

Figura 1: Proceso de Negocio de los Trabajos de Grado



Requerimientos no funcionales

1. Autenticación segura: implementación de login con contraseñas cifradas.
2. Trazabilidad: registro de acciones (quién hizo qué y cuándo).
3. Interfaz intuitiva: amigable para usuarios no técnicos.
4. Soporte multiusuario simultáneo sin pérdida de información.
5. Disponibilidad mínima: 99% durante los periodos académicos activos.
6. Portabilidad: el sistema debe poder desplegarse en servidores Linux y Windows.
7. Escalabilidad: debe permitir incorporar otras modalidades de grado en el futuro (plan co-terminal, homologación, etc.).
8. Auditoría: el sistema debe permitir la revisión posterior del historial de cambios por parte de un administrador.

Trabajo en equipo

Se debe trabajar en grupos de tres a cinco personas. Es importante que cada equipo trabaje de forma colaborativa, donde haya buenas relaciones entre todos sus integrantes, tareas individuales y metas comunes. El equipo debe buscar herramientas que favorezcan la comunicación durante el proyecto y la coordinación de tareas. Se recomiendan: Taiga.io, Trello o Jira.

Entregables

Para los tres cortes se debe entregar:

1. Especificación de **requisitos funcionales** (de los más importantes, de los que generen valor al cliente) ya sea mediante escenarios de casos de uso, o Historias de Usuarios y criterios de aceptación. Los requisitos deben estar acompañados de prototipos de la interfaz de usuario mediante wireframes y algún método de validación de usabilidad como thinking aloud.
2. Especificación de requisitos **NO funcionales** mediante escenarios de calidad.
3. Elegir el **tipo de aplicación** a construir: de escritorio, web tradicional, web responsive, web design, web single page, móvil nativa, móvil híbrida y servicios. Se debe justificar por qué.
4. Elegir un **estilo arquitectónico** acorde a los requisitos funcionales y no funcionales. Se debe justificar por qué.
5. Elegir la **tecnología** (lenguaje, framework de desarrollo, motor de base de datos, etc.). En resumen, las preguntas que tenemos que responder son: ¿Qué tecnologías

ayudan a implementar los estilos arquitecturales seleccionados? ¿Qué tecnologías ayudan a implementar el tipo de aplicación seleccionada? ¿Qué tecnologías ayudan a cumplir con los requisitos no funcionales especificados?
6. Documentación de la arquitectura, ya sea con el modelo C4 o UML.
7. Prototipo funcional que compile, ejecute, pase las pruebas y potencialmente sea usado por el cliente.

Estos puntos deben estar en un documento en PDF. En el punto 7, debe colocarse la URL del repositorio GitHub del proyecto. A continuación un formato para este documento:

1. Portada
2. Resumen
3. Requisitos funcionales en forma de historias épicas, historias de usuario y criterios de aceptación (Referenciar la hoja de cálculo de la plantilla de historias de usuario).
4. Prototipos y test de usabilidad (en caso de haberlos realizado)
5. Requisitos no funcionales en forma de escenarios de atributos de calidad.
6. Diagrama entidad relación
7. Arquitectura del sistema usando el modelo C4 más algunos diagramas del modelo 4+1 (los más relevantes).
8. Decisiones de arquitectura y su justificación (Selección del tipo de aplicación, Selección del estilo arquitectónico, Selección de tecnologías)
9. URL del repositorio de GitHub.

Corte 1 **- Primera iteración:** hacer una implementación monolítica en capas (MVC). Utilizar principios de diseño y patrones de diseño para garantizar la **modificabilidad** de la aplicación. Se debe implementar al menos un conjunto de historias de usuario de alto valor para el cliente (que serán definidas en otro archivo).

Corte 2 **- Segunda iteración:** hacer una refactorización de la iteración 1 implementando una solución distribuida con arquitecturas microservicios y orientada a eventos para garantizar el la **escalabilidad**. Utilizar los patrones de diseño vistos.

Corte 3 **- Tercera iteración:** seguir con la implementación distribuida y utilizar una **arquitectura hexagonal,** abarcando tres requisitos funcionales nuevos de alta prioridad para el cliente. Utilizar más patrones de diseño GoF. Tener en cuenta aspectos de **seguridad** para la autenticación y la autorización.