Motor gráfico:

- Construcción y configuración del proyecto con Maven2. Dependencia de JOGL. DLLs para depuración. El navegador descargará las que correspondan en producción.
- Patrón singleton para acceder a los objetos propios de GL, GLU y GLUT. Excepción GLSingletonNotInitializedException.
- Patrón observador para notificar a la interfaz contenedora del motor gráfico los eventos producidos en los modelos gráficos del mismo.
- Patrón fachada para permitir que la interfaz contenedora del motor gráfico pueda configurar sus parámetros de entrada de un modo sencillo, abstrayendo a los programadores de toda la lógica de dominio subyacente.
- Uso de JSON para establecer el formato de los parámetros de entrada del motor gráfico.
- Patrón factoría para instanciar los gestores de vistas o escenas de un modo sencillo.
- Arquitectura extensible y reusable de los gestores de vistas/escenas basada en herencia y polimorfismo.
- Jerarquía de modelos gráficos basada en herencia comenzando en características básicas para modelos 2D, después características más genéricas para modelos 3D, y por último, características específicas de cada modelo.
- Cámara (operaciones con vectores), foco de luz y sombras.
- Clase GLDrawer. Inicialización de OpenGL y bucle de control.
- Cálculo en tiempo real de la distancia de cada elemento a la cámara para renderizar las transparencias de un modo correcto.
- Dibujo del skybox
- Leyendas. Cómo dibujarlas correctamente en base al cálculo de píxeles.
- Método de selección de modelos de una escena.
- Cambio de proyeccion Perspectiva -> Ortogonal, y viceversa.
- Captura de eventos del teclado y del ratón (navegabilidad y selección de ítems).
- Clase GLUtils con métodos auxiliares que proporcionan distintas funcionalidades de visualización: cambios de perspectiva, cálculo y dibujo de sombras, renderezado de textos, cálculo de píxeles a coordenadas y viceversa.

- Clase GLLogger para propósitos de depuración.
- Cómo pintar un objeto (incluir código de una torre o una bola con antenas).
- Texturas (cómo leer el fichero en disco, cómo crear la textura y cómo mapearla en el modelo).
- Clase Synchronizer para mantener un control sobre el hilo que carga el applet y el hilo de ejecución de OpenGL.
- Clases City y Neighborhood para establecer el posicionamiento de la ciudad en base a barrios. Los elementos que forman un barrio son objetos de un modelo gráfico.
- Java API Reflection. Anotación GLDimension (con sus propiedades) para poder especificar las dimensiones mapeables de un modelo, y su correspondiente Analizador (parser), que usa el método de introspección.
- Claset AppletMain que extiende de Applet e implementa las interfaces del observador notificador de eventos de click y doubleClick, y de la fachada de acceso al motor.
- Conversión de colores de formato hexadecimal a RGB.
- Diagramas de casos de uso.
- Diagramas de clases de análisis.
- Diagramas de clases de diseño.
- Diagramas de interacción.
- Diagramas de secuencia.

Aplicación Web:

- Construcción y configuración del proyecto con Maven2.
- Control de Acceso Basado en Grupos. Configuración de Spring Security con una base de datos personalizada y autenticación basada en Hibernate.
- Integración del motor gráfico como Applet (código HTML y fichero JNLP; firma de JOGAMP; descarga automática de librerías; etc.). Comunicación de la lógica de negocio de la aplicación web con el motor gráfico mediante cadenas de texto con formato JSON.
- Clase ApplicationContextProvider, que implementa el interfaz ApplicationContextAware de Spring, para proveer acceso a los beans instanciados mediante Spring.

- JAXB API para serializar jerarquías de objetos Java en ficheros XML, y viceversa. Clase genérica XMLAgent que permite serializar y deserializar cualquier tipo de objeto y fichero xml.
- Jerarquía de objetos para serializar y deserializar perfiles de visualización.
- Java API Reflection. Anotación Property (con sus propiedades), encapsulada mediante la clase PropertyWrapper, para poder especificar los atributos mapeables de las clases de dominio que representan la lógica de negocio del desarrollo global, y su correspondiente Analizador (parser), que usa el método de introspección. Explicar los dos modos que hay para realizar cálculos y obtener indicadores.
- Instanciación de acciones y DAOs mediante beans de Spring.
- Implementación de los casos de uso mediante acciones de Struts2:
    - Gestión de compañías (crear, editar y eliminar).
    - Gestión de factorías (crear, editar y eliminar).
    - Gestión de proyectos (crear, editar y eliminar).
    - Gestión de subproyectos (crear, editar y eliminar).
    - Validadores.
    - Autenticación de usuarios (login y logout mediante delegando en el framework Spring Security).
    - Configuración de perfiles (interfaz adaptable gracias al uso de las anotaciones de los modelos gráficos y de las clases de lógica de negocio; generación de ficheros XML para hacer los perfiles persistentes).
    - Visualización:
        - Google Maps.
        - Java API Reflection y método de introspección -clase GLObjectManager- para generar los objetos en tiempo de ejecución a partir del perfil seleccionado.
        - JSON para generar la cadena de texto de entrada del motor gráfico a partir de los objetos generados por GLObjectManager.
- Implementación de los DAOs implementando el interface HibernateDaoSupport de Spring.
- Persistencia mediante Hibernate y anotaciones JPA.

- Jerarquía de DAOs extensible para permitir el uso de otras tecnologías alternativas a Hibernate.
- Interceptor RedirectMessageInterceptor para permitir una mejor integración entre acciones y poder hacer paso de mensajes entre ellas.
- Interceptor UserRoleAuthorizationInterceptor para definir la política de qué roles podrán ejecutar cada acción de Struts2.
- Internacionalización.
- Struts-menu: Menús dinámicos configurables con ficheros XML.
- Displaytag: paginación y tablas dinámicas.
- JQuery para incorporar AJAX.
- Sitemesh para decoración y look&feel consistente.
- JSColorPicker para hacer los selectores de color.
- Diagramas de casos de uso.
- Diagramas de clases de análisis.
- Diagramas de clases de diseño.
- Diagramas de interacción.
- Diagramas de secuencia.
- Pruebas.