1. Análisis

1.1 Revisión de especificación de requisitos.
1.1.1 Norma IEEE830.
1.1.2 Trazabilidad de requisitos.
1.2 Descripción de procesos actuales.
1.3 Diagramas UML.
1.4 Estudio de Factibilidad.
1.5 Análisis Costo-Beneficio.

¿Qué es UML?

El **Lenguaje Unificado de Modelado** (UML - Unified Modeling Language) creado para tener una opción visual común, semántica, sintácticamente para la arquitectura, el diseño, en la implementación de sistemas, definiendo estructuras y comportamientos.

UML es una combinación de varias notaciones orientadas a objetos:

- Diseño orientado a objetos.
- Técnica de modelado de objetos.
- Ingeniería de procesos automatizados orientada a objetos.

- UML usa las fortalezas de los tres enfoques anteriores mencionados para establecer una metodología uniforme sencilla de usar.

- UML representa buenas prácticas para la construcción y documentación de diferentes aspectos del modelado en ingeniería de procesos automatizados.

Object Management Group (OMG)

Consorcio internacional sin fines de lucro y de membresía abierta para estándares tecnológicos, fundado en 1989.

OMG supervisa la definición y el mantenimiento de las especificaciones de UML. Esta supervisión ofrece la capacidad de usar un lenguaje para varios propósitos durante todas las etapas del ciclo de vida de sistemas.

- Establecer una definición formal de un metamodelo común basado en el estándar **MOF (Meta-Object Facility)** que especifique la sintaxis abstracta del UML.

- La sintaxis abstracta define el conjunto de conceptos de modelado UML, sus atributos y sus relaciones, así como las reglas de combinación de estos conceptos.

- La semántica define, de manera independiente a la tecnología, cómo los conceptos UML se habrán de desarrollar por las computadoras.

- Especificar los elementos de notación de lectura humana para representar los conceptos individuales de modelado UML, así como las reglas para combinarlos en una variedad de diferentes tipos de diagramas que corresponden a diferentes aspectos de los sistemas modelados.

- Definir formas que permitan hacer que las herramientas UML cumplan con esta especificación. Esto se apoya (en una especificación independiente) con una especificación basada en **XML (eXtensible Markup Language)**, de formatos de intercambio de modelos correspondientes (XMI) que deben ser concretados por herramientas compatibles.

Metamodelo de almacén común (CWM) Interfaces estándares que se usan para permitir el intercambio de metadatos de almacén e inteligencia de negocios entre herramientas de almacén, plataformas de almacén y repositorios de metadatos de almacén en entornos heterogéneos distribuidos.

Núcleo en el contexto de UML, el núcleo comúnmente se refiere al "paquete central", que es un metamodelo completo particularmente diseñado para una alta reutilización.

Nivel 0 (L0) Nivel de cumplimiento inferior para la infraestructura UML - una sola unidad de lenguaje que hace posible el modelado de tipos de estructuras basadas en clases que se encuentran en los lenguajes más populares de programación orientados a objetos.

Meta Object Facility (MOF) Una especificación de modelado de OMG que brinda la base para las definiciones de metamodelos en la familia de lenguajes MDA de OMG.

Metamodelo Define el lenguaje y los procesos a partir de los cuales formar un modelo.

Construcciones de metamodelos (LM) Segundo nivel de cumplimiento en la infraestructura UML - una unidad adicional de lenguaje para estructuras más avanzadas basadas en clases, usadas para construir metamodelos (por medio de CMOF), tales como el UML mismo. UML solo tiene dos niveles de cumplimiento.

Arquitectura dirigida por modelos (MDA) Un enfoque y un plan para lograr un conjunto coherente de especificaciones de tecnología dirigida por modelos.

Lenguaje de restricciones para objetos (OCL) Un lenguaje declarativo para describir reglas que se aplican al Lenguaje Unificado de Modelado. OCL complementa a UML proporcionando términos y símbolos de diagramas de flujo.

Conceptos de modelado especificados por UML

- **Funcionales**: Se trata de diagramas de casos de uso que describen la funcionalidad del sistema desde el punto de vista del usuario.
- **De objetos**: Se trata de diagramas de clases que describen la estructura del sistema en términos de objetos, atributos, asociaciones y operaciones.
- **Dinámicos**: Los diagramas de interacción, los diagramas de máquina de estados y los diagramas de actividades se usan para describir el comportamiento interno del sistema.

Estos modelos de sistemas se visualizan a través de dos tipos diferentes de diagramas: estructurales y de comportamiento.

Conceptos orientados a objetos en UML

- **Objetos** Representan una entidad y el componente básico.
- **Clase** Plano de un objeto.
- **Abstracción** Comportamiento de una entidad del mundo real.
- **Encapsulación** Mecanismo para enlazar los datos y ocultarlos del mundo exterior.
- **Herencia** Mecanismo para crear nuevas clases a partir de una existente.
- **Polimorfismo** Define el mecanismo para salidas en diferentes formas

Tipos de diagramas UML

UML usa elementos y los asocia de diferentes formas para formar diagramas que representan aspectos estáticos o estructurales de un sistema, y diagramas de comportamiento, que captan los aspectos dinámicos de un sistema.

Diagramas UML estructurales

- Diagrama **de clases** El diagrama UML más comúnmente usado, y la base principal de toda solución orientada a objetos. Las clases dentro de un sistema, atributos y operaciones, y la relación entre cada clase. Las clases se agrupan para crear diagramas de clases al crear diagramas de sistemas grandes.
- Diagrama **de componentes** Muestra la relación estructural de los elementos del sistema de software, muy frecuentemente empleados al trabajar con sistemas complejos con componentes múltiples. Los componentes se comunican por medio de interfaces.
- Diagrama **de estructura compuesta** Los diagramas de estructura compuesta se usan para mostrar la estructura interna de una clase.
- Diagrama **de implementación** Ilustra el hardware del sistema y su software. Útil cuando se implementa una solución de software en múltiples máquinas con configuraciones únicas.
- Diagrama **de objetos** Muestra la relación entre objetos por medio de ejemplos del mundo real e ilustra cómo se verá un sistema en un momento dado. Dado que los datos están disponibles dentro de los objetos, estos pueden usarse para clarificar relaciones entre objetos.
- Diagrama **de paquetes** Hay dos tipos especiales de dependencias que se definen entre paquetes: la importación de paquetes y la fusión de paquetes. Los paquetes pueden representar los diferentes niveles de un sistema para revelar la arquitectura. Se pueden marcar las dependencias de paquetes para mostrar el mecanismo de comunicación entre niveles.

Diagramas UML de comportamiento

- Diagramas **de actividades** Flujos de trabajo de negocios u operativos representados gráficamente para mostrar la actividad de alguna parte o componente del sistema. Los diagramas de actividades se usan como una alternativa a los diagramas de máquina de estados.
- Diagrama **de comunicación** Similar a los diagramas de secuencia, pero el enfoque está en los mensajes que se pasan entre objetos. La misma información se puede representar usando un diagrama de secuencia y objetos diferentes.
- Diagrama **de panorama de interacciones** Hay siete tipos de diagramas de interacciones. Este diagrama muestra la secuencia en la cual actúan.
- Diagrama **de secuencia** Muestra cómo los objetos interactúan entre sí y el orden de la ocurrencia. Representan interacciones para un escenario concreto.
- Diagrama **de máquina de estados** Similar a los diagramas de actividades, describen el comportamiento de objetos que se comportan de diversas formas en su estado actual.
- Diagrama **de temporización** Al igual que en los diagramas de secuencia, se representa el comportamiento de los objetos en un período de tiempo dado. Si hay un solo objeto, el diagrama es simple. Si hay más de un objeto, las interacciones de los objetos se muestran durante ese período de tiempo particular.
- Diagrama **de caso de uso** Representa una funcionalidad particular de un sistema. Se crea para ilustrar cómo se relacionan las funcionalidades con sus controladores (actores) internos/externos.

Diagrama de clases (ejemplo).

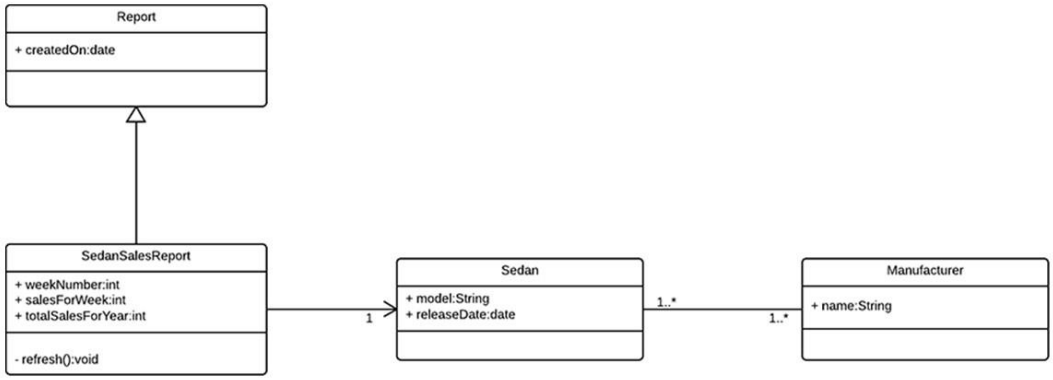


Diagrama de componentes (ejemplo).

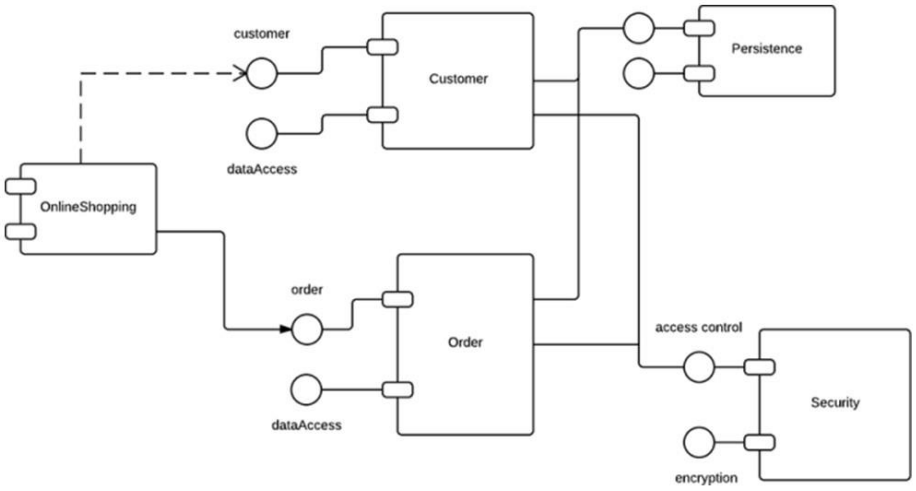


Diagrama de implementación (ejemplo).

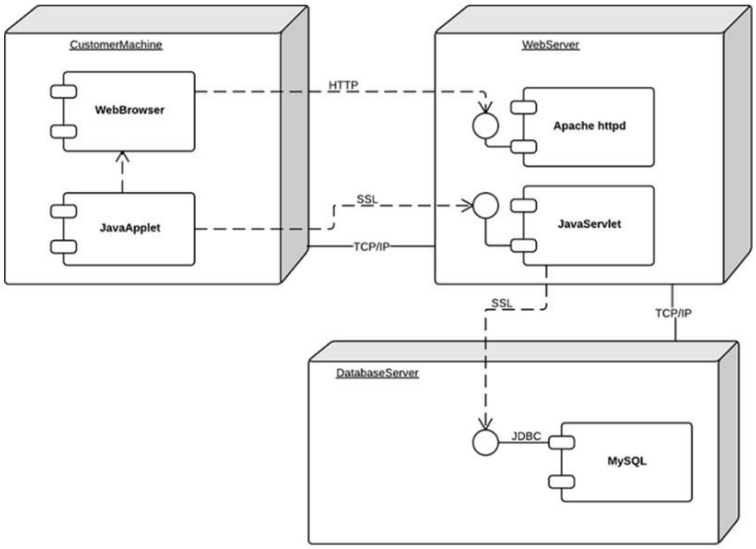


Diagrama de actividades (ejemplo).

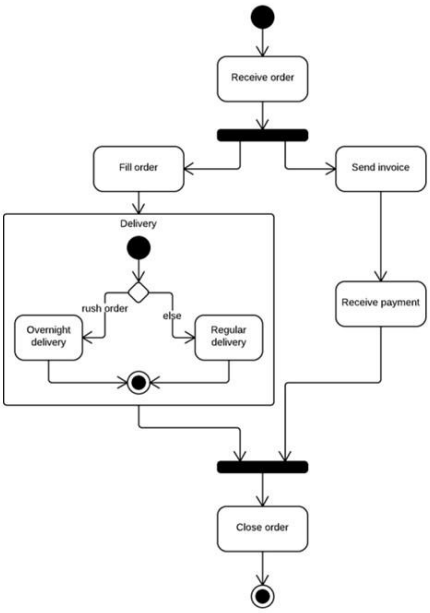


Diagrama de casos de uso (ejemplo).

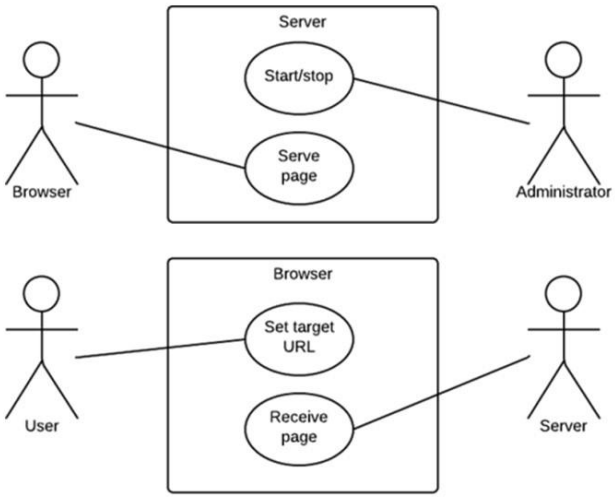
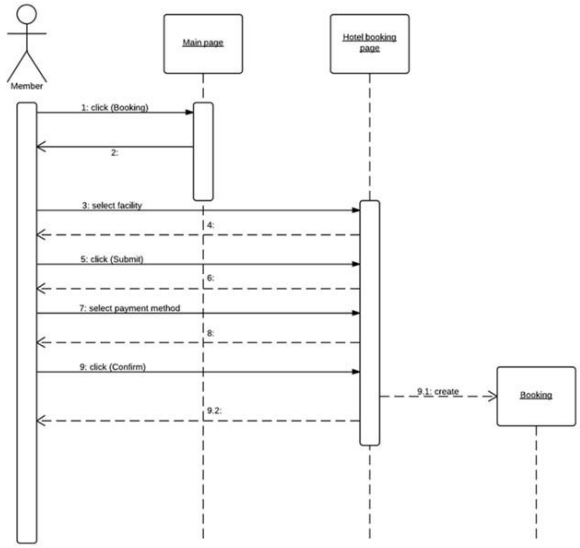


Diagrama de secuencia (ejemplo).



Referencia arbitrada.

OMG Unified Modeling Language (OMG UML), Version 2.5.1
https/www.omg.org/spec/UML/

- Componentes
- Infraestructura
- Diagramas

https://moodle.ucags.edu.mx/course/view.php?id=2082&section=6

Referencias.

- Aplicaciones, Uml & Java, En & ++, C & C++, Y & Jiménez de Parga, Carlos. (2015). UML: Java and C++ applications.
- Kamiński, Tomasz & Kamiński, Piotr. (2023). Application of Use Cases and the UML language in the design of Intelligent Transportation Systems. Journal of Civil Engineering and Transport. 5. 39-48. 10.24136/tren.2023.007.
- Murti, KCS. (2022). UML for Embedded Systems. 10.1007/978-981-16-3293-8_5.
- Zelinski, Jaroslaw. (2023). Diagrams in UML Notation.
- Lano, Kevin. (2023). The Agile UML Manual.
- Houndji, Vinasetan & Akotenou, Genereux. (2023). UMLDesigner: An Automatic UML Diagram Design Tool. 10.1007/978-3-031-39059-3_23.
- Knapp, Alexander. (2023). An Intermediate Language-Based Approach to Implementing and Verifying Communicating UML State Machines. 10.1007/978-3-031-40132-9_18.

1. Análisis

1.1 Revisión de especificación de requisitos.
1.1.1 Norma IEEE830.
1.1.2 Trazabilidad de requisitos.
1.2 Descripción de procesos actuales.
1.3 Diagramas UML.
1.4 Estudio de Factibilidad.
1.5 Análisis Costo-Beneficio.

Factibilidad se refiere a la disponibilidad de los recursos necesarios para llevar a cabo los objetivos y metas propuestas.

"*El éxito de un proyecto está determinado por el grado de factibilidad que se presente en cada aspecto a evaluar (Técnico, Económico y Operativo*)".

¿Qué es?

El estudio de factibilidad es el análisis que lleva a cabo una empresa/institución/organización, para determinar si el proyecto que se propone será pertinente, considerando las estrategias que se deben desarrollar para que sea exitoso.

Objetivos

- Reducción de eventualidades y mayor precisión en los procesos.
- Reducción de costos mediante la optimización o eliminación de los recursos no necesarios.
- Integración de todas las áreas y subsistemas.
- Actualización y mejora continua de los servicios.
- Agilidad en la recopilación de los datos.
- Reducción en el tiempo de procesamiento y ejecución de los procesos.
- Automatización de procedimientos manuales.
- Disponibilidad de los recursos necesarios para llevar a cabo los objetivos.
- Medir el retorno de inversión.

Fases factibilidad técnica

- Planteamiento del proyecto.
- Arquitectura general del proyecto.
  - Consideraciones de hardware, software, firmware.
  - Benchmarking.
- WorkFlow del proyecto y actividades.
- Implementación de prototipo.

Fases factibilidad económica

- Costos generales.
- Costos del ambiente.
- Costos del personal - roles.
- Costos operativos.
- Costos dimensionados en la línea de tiempo.

Fases factibilidad operativa

- **Administración de riesgos (Risk Management).**
- **Retorno de inversión ROI (Return On Investment).**

Research

IEEE | **Reinforcing the DevOps approach with security and risk management**: an implementation experience in a data center of a Mexican organization. DOI: 10.1109/CIMPS.2017.8169957

1. Análisis

1.1 Revisión de especificación de requisitos.
1.1.1 Norma IEEE830.
1.1.2 Trazabilidad de requisitos.
1.2 Descripción de procesos actuales.
1.3 Diagramas UML**.**
1.4 Estudio de Factibilidad.
1.5 Análisis Costo-Beneficio.

El **análisis de costo-beneficio** es una herramienta de toma de decisiones que permite evaluar si una acción o proyecto es viable desde el punto de vista económico.

Comparar diferentes alternativas de diseño, desarrollo, implementación o mantenimiento de un ecosistema informático.

El objetivo es elegir la opción que maximice el valor para el usuario y minimice los recursos necesarios.

Ejemplos:

- Costos: tiempo, presupuesto, personal, infraestructura, calidad, riesgos, entre otros.
- Beneficios: funcionalidad, usabilidad, seguridad, fiabilidad, rendimiento, ingresos, entre otros.

Nota: es una técnica útil para tomar decisiones racionales y basadas en evidencia en la ingeniería de datos. sin embargo, también tiene algunas limitaciones, como la dificultad para cuantificar algunos costos o beneficios intangibles, la incertidumbre sobre las estimaciones o las preferencias subjetivas de los interesados. Por eso, "es pertinente complementar con otras herramientas o métodos de evaluación".

Tips & Tricks

- Listar las alternativas de proyectos o soluciones.
- Identificar a los interesados o partes afectadas por la decisión.
- Seleccionar y medir todos los elementos de costo y beneficio relevantes para cada alternativa.
- Predecir el resultado del costo y los beneficios durante el ciclo de vida del proyecto.
- Convertir todos los costos y beneficios a una moneda común y aplicar una tasa de descuento si es necesario.
- Calcular el valor presente neto o la relación costo-beneficio de cada alternativa.
- Realizar un análisis de sensibilidad para evaluar cómo cambian los resultados ante diferentes escenarios o supuestos.
- Elegir la opción recomendada según el criterio establecido.

B/C = VAI / VAC

B/C: *relación costo-beneficio.*
VAI: *valor actual de los ingresos totales netos o beneficios netos.*
VAC: *valor actual de los costos de inversión o costos totales.*

- B/C **mayor a 1**: *quiere decir que los ingresos son superiores a los costos, por lo que el proyecto es rentable.*
- B/C **igual a 1**: *significa que no hay ni ganancias ni pérdidas, ya que uno absorbe al otro, así el proyecto no es viable.*
- B/C **menor a 1**: *indica que los costos sobrepasan a los beneficios por lo que el proyecto no es rentable.*