



Trabajo n.º 4:

Estudio de herramientas CASE libres de soporte a UML

Juan Francisco Romero Lavirgen
Ingeniería de Requisitos. Curso 20/21

Una breve introducción...

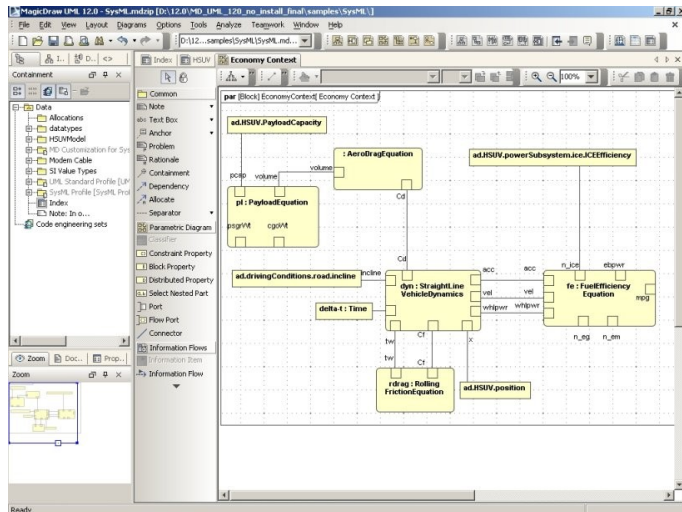
- Una herramienta CASE es una aplicación destinada a aumentar el balance en el desarrollo de software reduciendo el costo de tiempo y dinero.
- Programa que facilita las labores del desarrollador o diseñador en cualquier fase del proceso de desarrollo de un sistema, aumentando su productividad.
- Aumentador número de alternativas tanto a nivel personal como corporativo. ¿Cuál es la más apropiada para nosotros? ¿Y para un profesional?
- En este trabajo nos centraremos en aquellas de soporte a UML 2.

Herramientas de pago



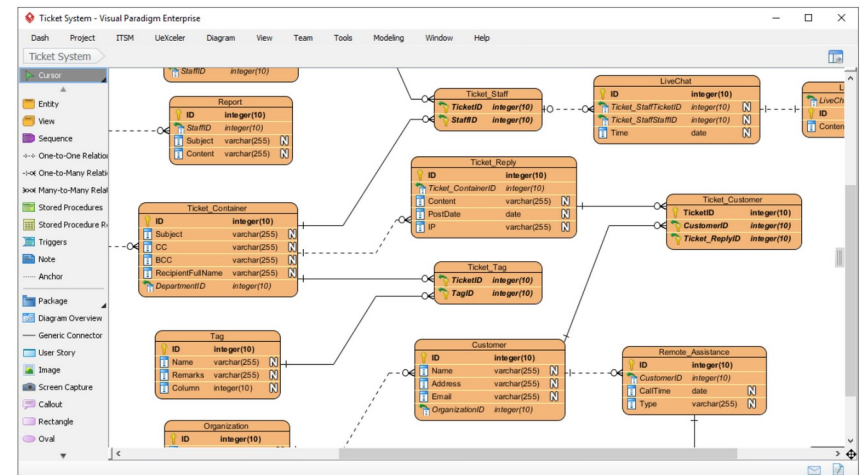
MagicDraw:

- Funcionalidad inmejorable
- Elevada complejidad dada sus vistas, secciones, etc
- Permite Plugins como MagicUWE (utilizado en IW)



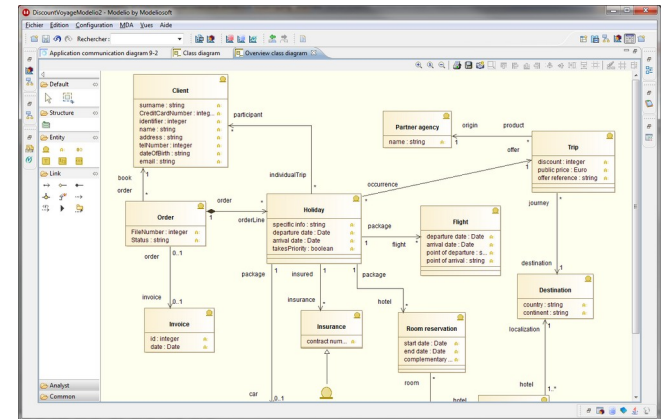
Visual Paradigm:

- Posibilidad de ingeniería inversa (obtener diagramas a partir de código)
- Soporte a BPMN



Modelio

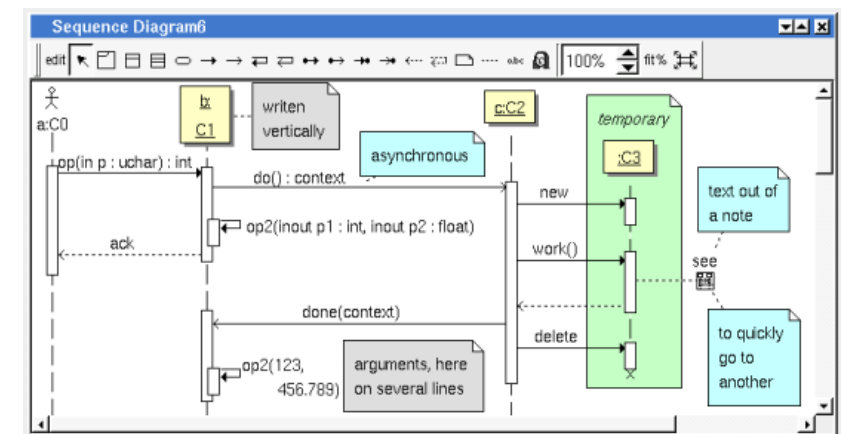
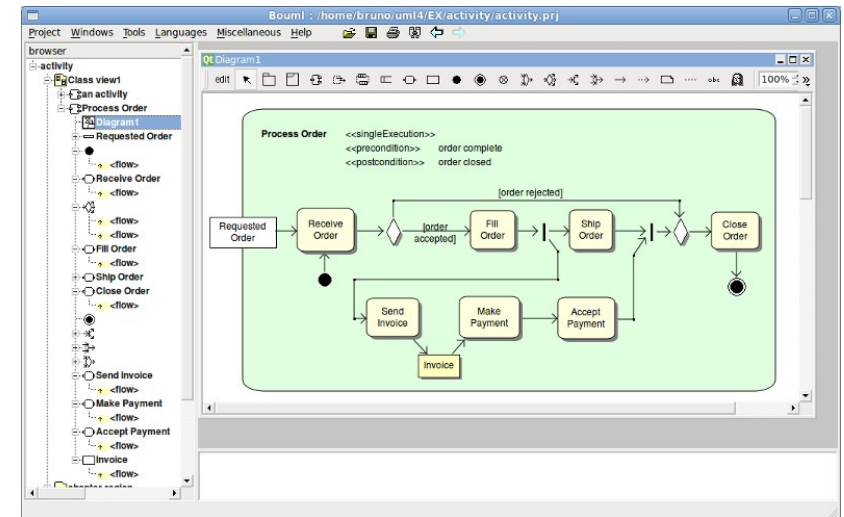
- Enorme extensibilidad: se trata de un programa altamente modular.
- El usuario puede comenzar desde abajo e ir ampliando cada vez más y más según sus necesidades.
- Tienda con muchos modelos.
- Algunos gratuitos pero otros...
- Comerciales.
- C++ Designer, Java Designer, Junit, etc.
- Multiplataforma.



boUML

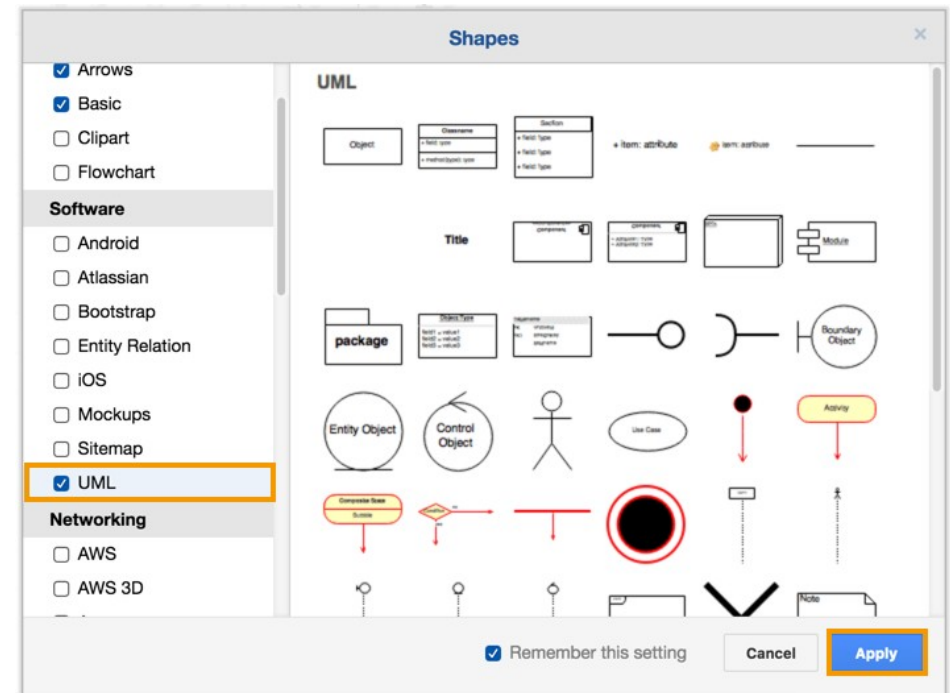


- Sencillo de utilizar.
- MUY rápida con bajo consumo de memoria.
- Puede generar documentación en formato HTML, XML (para intercambio de diagramas entre herramientas)
- Trabajo en grupo con los módulos Project Control y Project Synchro.
- Se queda corto en proyectos grandes.
- Multiplataforma.



Draw.io

- Herramienta online: sin necesidad de instalar nada.
- Sin trampas.
- Gran integración con servicios de almacenamiento remoto: Google Drive, Dropbox, OneDrive...
- Herramienta de dibujo.



Plugins para UML

No es necesario cambiar de aplicación. Todo en uno: diagramas/modelos y editor de texto para codificar.

UML 2 Tools:

- Bastante ligero.



UMLet:

- Simple y poco pretencioso.
- También existe la versión standalone.

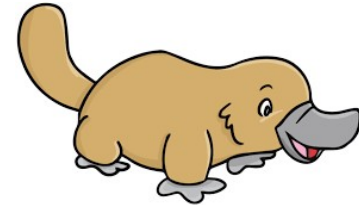
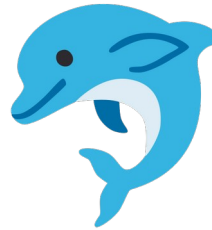





Papyrus:

- Estándar de facto en Eclipse.
- Muy completo.
- Difícil de usar y necesidad de documentación.
- Requiere crear un proyecto y la exportación de diagramas no es tan fácil.

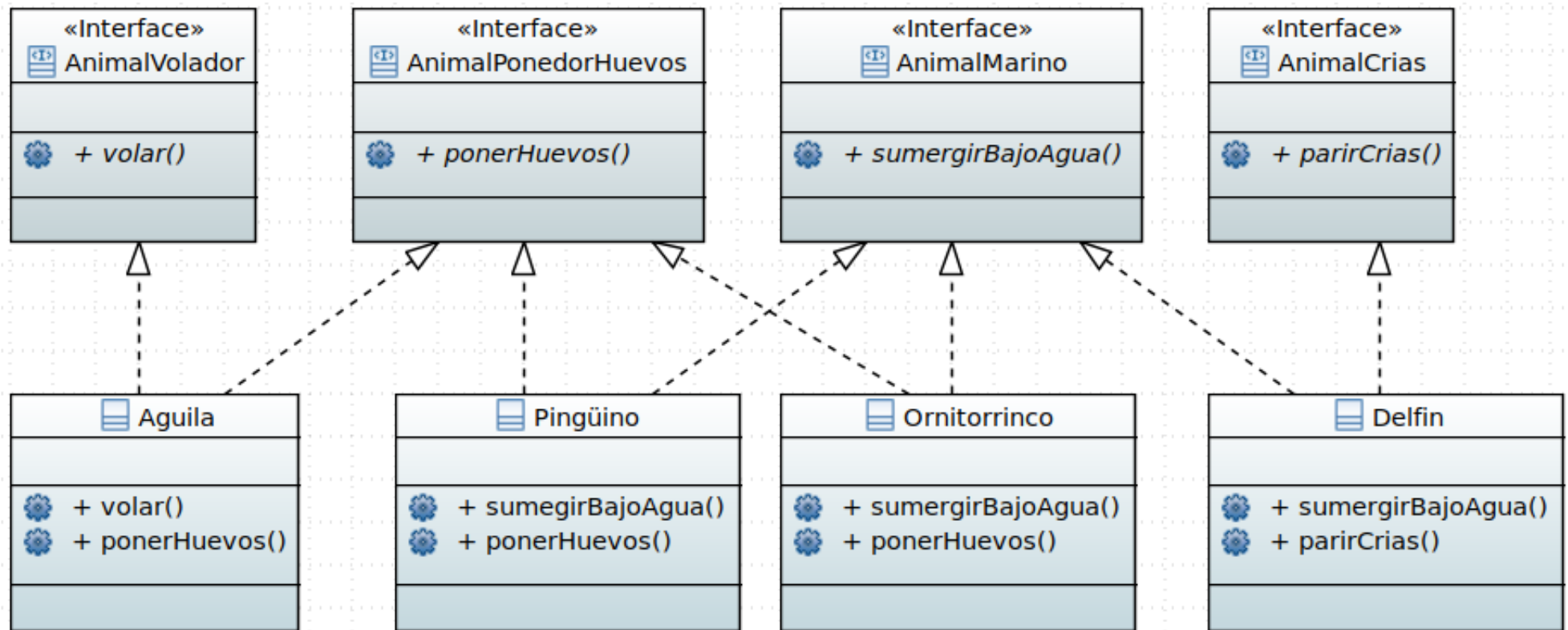
Caso práctico: Dia vs Papyrus

- Queremos modelar un zoológico-acuario con: pingüinos, águilas, delfines y ornitorrincos.

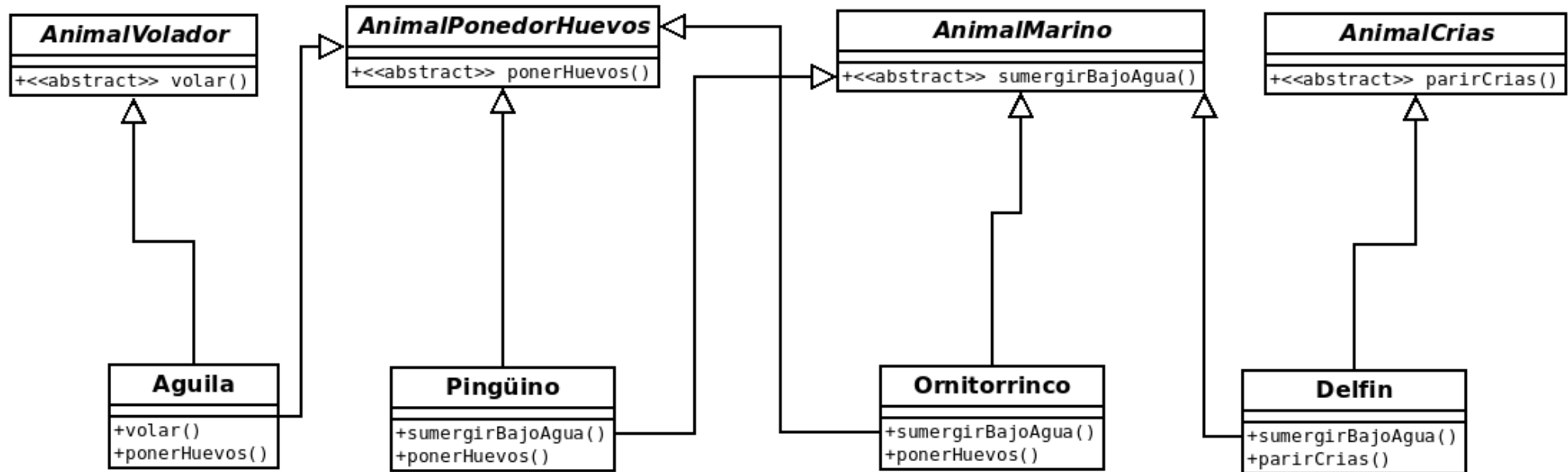


- Utilizaremos un diagrama de clases.
- ¿Clase Mamífero con operación parirCrias()? 
- ¿Clase Ave con operación volar()? 
- ¿Utilizar multiherencia? 

Solución en Papyrus



Solución en Dia



¿Diferencias? MUCHAS.

Diferencias entre ambos modelos

- Papyrus utiliza Interfaces, Dia clases abstractas. ¿Detectas algún error conceptual?
- Papyrus puede indicar fácilmente que una operación es abstracta, Dia no. Solución: usamos estereotipos.

The screenshot shows the Papyrus UML editor's configuration window for an operation named 'volar ()'. The window has a sidebar on the left with tabs for 'UML', 'Comments', 'Profile', 'Style', and 'Appearance'. The 'UML' tab is selected. The main area contains the following fields:

Property	Value
Name	volar
Label	
Is abstract	<input checked="" type="radio"/> true <input type="radio"/> false
Is static	<input type="radio"/> true <input checked="" type="radio"/> false

- Lío en las flechas de Dia. Cada elemento tiene un número fijo de “puntos” en los que conectar.



Conclusiones

- Depende...



- Herramientas de dibujo



- Herramientas de modelado

