3.2 TÉCNICA DE MODELADO DE OBJETOS (OMT) (JAMES RUMBAUGH).

3.2.1 Introducción.

En este documento se trata tanto el OMT-1 como el OMT-2, el primero contenido en el Libro Modelado y Diseño Orientado a Objetos (Metodología OMT) de James Rumbaugh 1991, el segundo publicado en la página www.rational.com en la sección de white papers.

Un **modelo** es una abstracción de algo, con la finalidad de comprenderlo, antes de construirlo, ya que un modelo omite los detalles no esenciales, es más sencillo manejarlos, que manejar la entidad original.

Esta técnica es trilateral, ya que toma en cuenta tres puntos de vista: modelo de objetos, modelo dinámico y modelo funcional.

a) **El modelo de objetos**. El modelo de objetos es el modelo más importante, ya que en él se identifican las clases dentro del sistema junto con sus relaciones, así como sus atributos y operaciones, lo que representa la estructura estática del sistema. El modelo de objetos se representa mediante un diagrama de clases.
b) **El modelo dinámico**. Representa los aspectos temporales de comportamiento "de control" del sistema, mediante la secuencia de operaciones en el tiempo.
c) **El modelo funcional**. Representa los aspectos transformacionales "de función" del sistema, mediante la transformación de valores de los datos. Se representa mediante un diagrama de flujo.

Cada modelo describe un aspecto del sistema pero contiene referencias a los demás modelos. Lo cual indica que los tres no son totalmente independientes.

3.2.2 Pasos del proceso de desarrollo orientado a objetos

- **Conceptualización:** Se describen los requerimientos para la solución del sistema. Comienza identificando las necesidades desde el punto de vista de los usuarios. Dicha información puede ser extraída de los casos de uso y del dominio del problema.

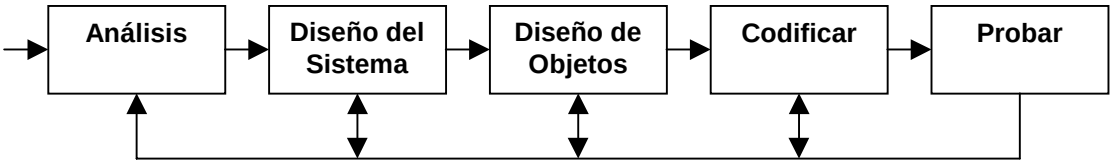
- **Análisis**: Entender y modelar el problema en el dominio de la aplicación.

- **Diseño del sistema**: Determinar la arquitectura del sistema en términos de subsistemas.

- **Diseño de objetos**: Refinar y optimizar el modelo de análisis, agregando conceptos de programación.

- **Código**: Implementar las clases de objetos en un lenguaje de programación.

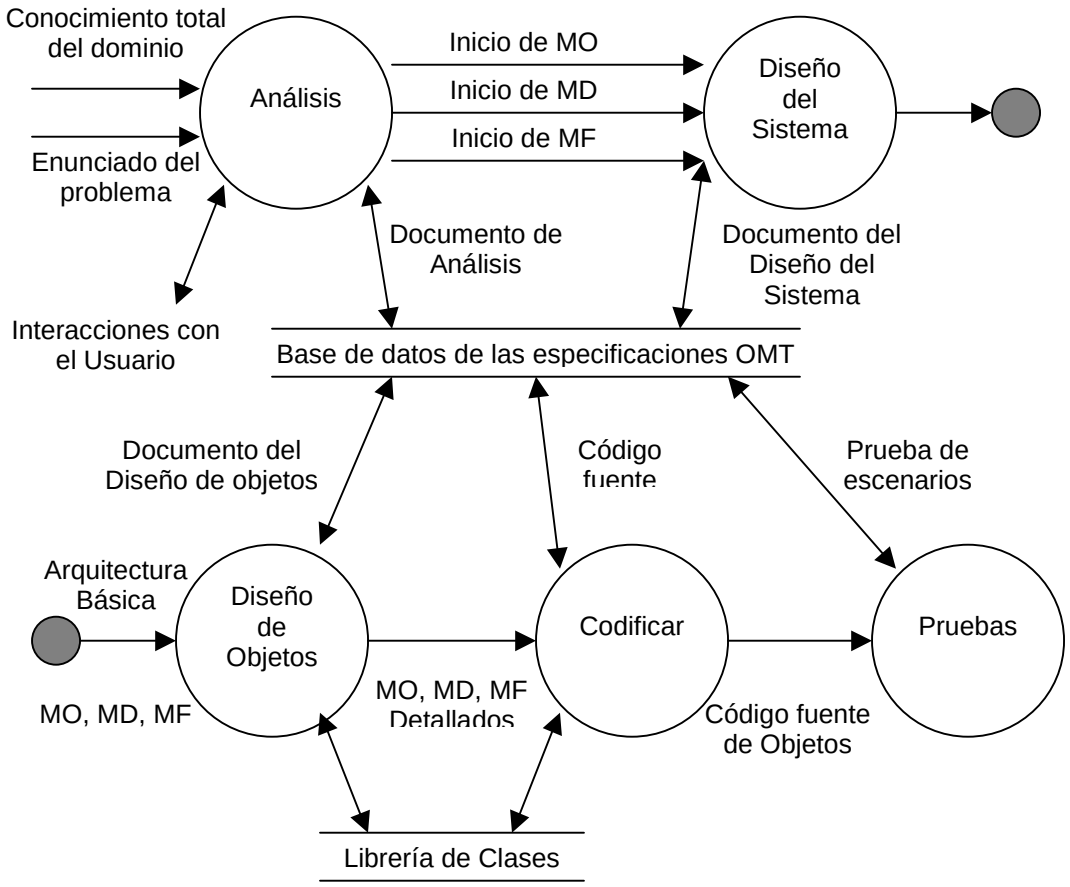
- **Pruebas**: se realizan para verificar el comportamiento de las clases y objetos que se encuentran descritos en los escenarios.



Proceso OMT

Figura # 19

Cada paso del proceso transforma algunas entradas para generar una salida diferente, comenzando en un alto nivel de abstracción hasta llevarlo a un nivel de detalle que finalmente representa la solución del problema.



Entradas y salidas del proceso de desarrollo Orientado a Objetos

Figura # 20

La figura # 20 se realizo utilizando el modelo funcional que será visto a mayor detalle mas adelante.

- **Conceptualización**

Casos de uso:

La utilidad de construir un caso de uso es para ver las diferentes vistas que tiene el usuario del sistema. Un caso de uso es una interacción entre el sistema y un actor, para describir el propósito del uso del sistema.

Actor:

Es algo externo al sistema que interactúa con él.

Los casos de uso pueden especificarse mediante escenarios que contendrán el propósito del caso de uso, los actores que interactúan con el, el evento inicial y la condición final. Los escenarios pueden ser representados mediante diagramas de traza de eventos o bien por el diagrama de interacción.

Pasos a seguir para producir un caso de uso:

1. Delimitar los limites del sistema: Cuales objetos pertenecen a el y cuales no. Desarrollar el caso de uso.
2. Determinar los actores que interactúan con el sistema según los roles dentro del mismo.
3. Para cada actor determinar la forma en como interactúa con el sistema produciéndose para cada uno e ellos un o mas casos de uso.
4. Identificar el evento inicial de cada caso de uso.
5. Determinar la condición de terminación para cada caso de uso.
6. Listar los eventos.
7. Si hay variaciones listar escenarios adicionales que las describan.
8. Identificar y describir todas las excepciones que están asociadas lógicamente con un dado caso de uso.
9. Verificar que los casos de uso engloben la funcionalidad del sistema.

- **Análisis del Sistema.**

Durante el **análisis** se construye un modelo en el dominio de la aplicación sin tener en cuenta la implementación que se deberá efectuar posteriormente. Deberá incluir aquella información que sea significativa desde el punto de vista del mundo real, presentando el especto externo del sistema, resultando comprensible para el cliente del sistema, proporcionando una buena base para extraer los verdaderos requisitos (congruentes y realizables) del sistema

El análisis comprende los pasos siguientes:

1. Se establece la definición del problema.
2. Se construye un modelo de objetos.
3. Se desarrolla un modelo dinámico.
4. Se construye un modelo funcional.
5. Se verifican, iteran y refinan los tres modelos.

Los pasos para construir el modelo de objetos son los siguientes:

1. Identificación de objetos y/o clases.
2. Crear un diccionario de datos.
3. Identificación de las asociaciones y agregaciones entre los objetos.
4. Identificación de atributos y enlaces.
5. Organización y simplificación de las clases empleando herencia.
6. Verificación de las vías de acceso necesarias para llevar a cabo las probables consultas.
7. Realizar las iteraciones necesarias para el refinamiento del modelo.
8. Agrupar las clases en módulos.

Modelo de objetos = Diagrama de modelo de objetos + diccionario de datos.

Diagrama de clases:

En el se describen las clases que se descubrieron para el sistema analizado en términos del dominio del problema. Además se especifican los atributos y operaciones que distinguen a cada una de las clases y las relaciones con las que podemos conocer su responsabilidad en el sistema.

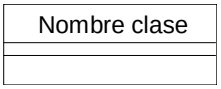


Figura # 21 Clase

Notación: dentro del diagrama de clases, una clase se representa mediante un rectángulo, donde pueden existir tres separaciones, en la primer parte se coloca el nombre de la clase, en la segunda y tercera parte se pueden agregar los atributos y las operaciones, pero sino se desea agregar ninguno de ellos, es porque no son tan importantes para la comprensión del sistema, entonces el rectángulo solo se queda con el nombre de la clase.

Atributos

Es un dato que distingue a una clase y que puede almacenar valores para el mismo en cada instancia que genere la clase. Debe tener un nombre y el tipo de dato que va a recibir. Los atributos se colocan en la segunda parte del rectángulo de la clase, primero se coloca el nombre del atributo, después precedido de dos puntos (:) el tipo de dato que recibirá y en algunos casos se podrá especificar el valor inicial que recibe precedido por un signo de igual (=).

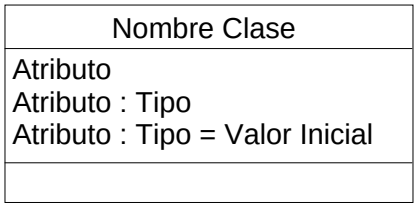


Figura # 22 Clase con Atributos

Operaciones

Las operaciones son funciones que pueden realizar las instancias de una clase. Mediante ellas se pueden visualizar cuales son las responsabilidades de cada clase dentro del sistema. Se colocan en la tercera parte del rectángulo de la clase y debe de contener el nombre de la operación que puede ir seguida de una lista de argumentos entre paréntesis y de un tipo de dato que regresará precedido de dos puntos (:).

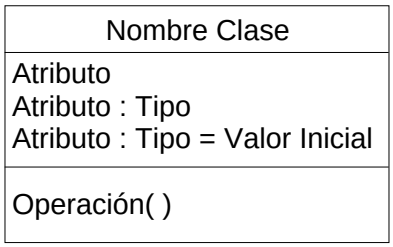


Figura # 23 Clase con atributos y operaciones

Existe un tipo de clase especial llamada clase abstracta. Este tipo de clase no genera instancias directas, lo único que hace es heredar a otras clases que pueden generar instancias directas. Como en el siguiente ejemplo la clase figura se denomina clase abstracta debido a que no se generan instancias para esa clase y lo único que hace es heredar los atributos; tamaño y color a las subclases círculo y cuadrado.

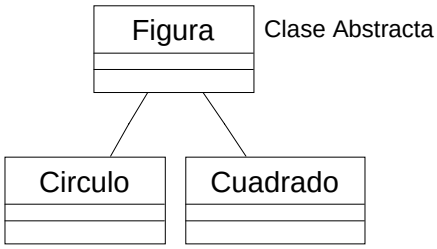


Figura # 24 Clase Abstracta

Diagrama de objetos:

En este diagrama se representan las instancias de las clases relacionadas entre sí.

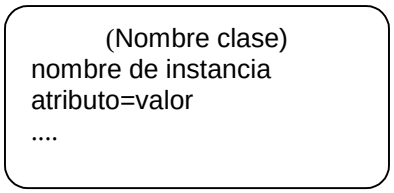


Figura # 25 Objeto

Objeto:

Es una instancia de una clase, se representa mediante un cuadro con esquinas redondeadas, donde se especifica un nombre para el objeto, se podrá definir entre paréntesis y en la parte superior el nombre de la clase que lo genera, así como los valores para sus atributos.

Relaciones (asociaciones):

Representan los enlaces entre las instancias dentro del diagrama. Se representan mediante una línea que conecta a las instancias junto con el nombre de la asociación que por lo general es un verbo.

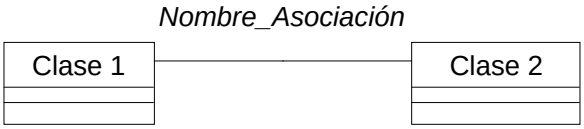


Figura # 26 Asociación

Multiplicidad en la asociación:

La multiplicidad especifica cuantas instancias de una clase estarán relacionadas con cada instancia de la otra clase.

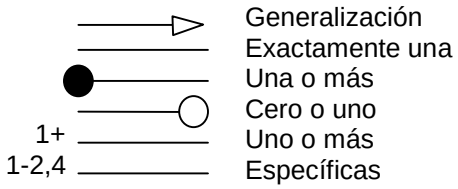


Figura # 27 Representación de Multiplicidad

Tipos de asociaciones

Existen dos tipos particulares de asociaciones: agregación y generalización.

Agregación (parte de):

Compuesta de dos tipos de instancias: la agregada y sus componentes. Dicha relación representa la asociación que hay entre los componentes y la clase que se compone de ellos. Se representa mediante un rombo en el extremo de la asociación que va hacia el agregado. Como se muestra en la Figura # 10, la clase árbol esta compuesta de hojas, por lo cual el agregado será árbol y sus componentes las instancias de hojas.

Figura # 28 Relación de agregación

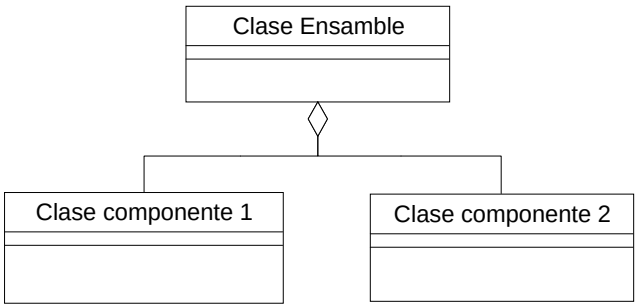


Figura # 29 Relación de Agregación

Generalización:

Es una relación entre una superclase que hereda sus características (atributos y operaciones) y subclases que harán suyas dichas características. La asociación se representa mediante un triangulo que conecta a la superclase con sus subclases.

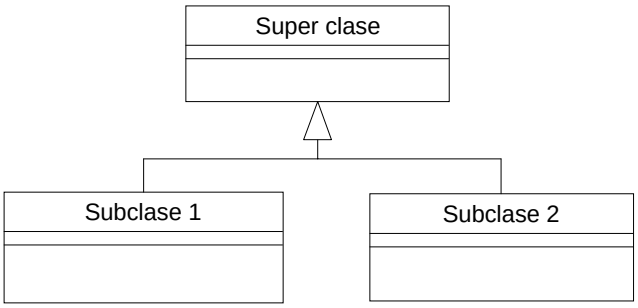


Figura # 30 Relación de Generalización

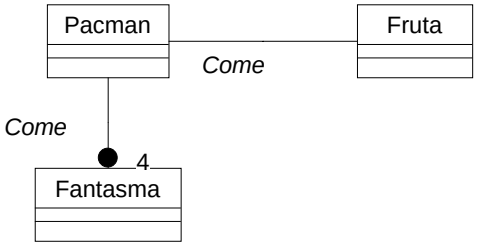


Figura # 31 Diagrama de Clases

Los pasos para construir el modelo dinámico son los siguientes:

1. Preparación de escenarios de secuencias típicas de iteración.
2. Identificación de sucesos que actúan entre objetos.
3. Preparar un seguimiento de sucesos para cada escenario.
4. Construcción de un diagrama de estado para cada objeto.
5. Comparación de los sucesos intercambiados entre objetos para verificar la congruencia.

Modelo dinámico = Diagrama de estados + diagrama global de flujo de sucesos.

Escenario:

Es la representación escrita de los casos de uso y de la interacción de los actores con ellos para especificar el propósito del sistema.

Escenario
Es una descripción escrita en lenguaje natural que relata los pasos que se siguen para la ejecución de un proceso

Figura # 32 Escenario.

Notación de traza de sucesos:

Suceso:

Es un evento que ocurre en un determinado momento del sistema y por medio del cual se pueden transmitir valores entre los objetos.

Desarrollar el diagrama de traza de sucesos es el siguiente paso para representar los escenarios después de haberlos realizado.

Cada objeto se muestra como una línea vertical. Los sucesos son representados mediante una flecha que va desde el objeto emisor al objeto receptor, y en el cual se puede incrementar el tiempo de arriba hacia abajo, según avanza este. Mediante el diagrama de traza de sucesos se muestra la forma en como los objetos se comunican entre si enviándose mensajes, visto de otra forma, son peticiones de operaciones a realizar que un objeto le pide a otro.

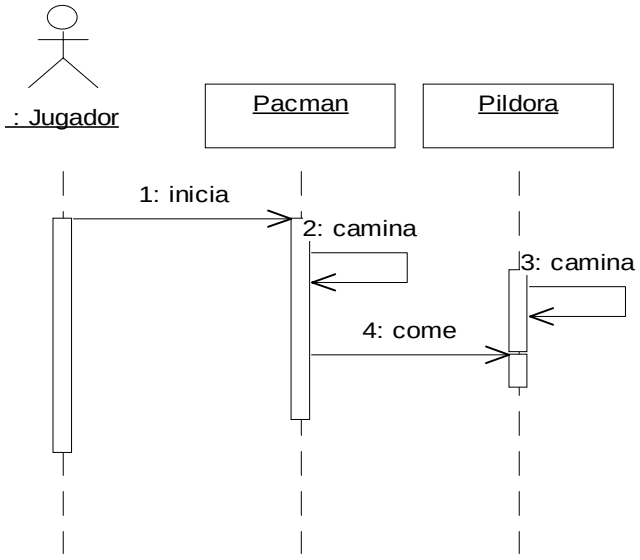


Figura # 33 Diagrama de seguimiento de traza de sucesos.

Diagramas de estados:

Relaciona sucesos y estados. Un diagrama de estados se representa mediante estados, transiciones, condiciones y acciones.

Estados:

Los estados representan las respuestas de los objetos a varios sucesos en determinado tiempo dentro del sistema. Dicha respuesta puede cambiar el estado del objeto. Se representan mediante cuadros redondeados que contienen un nombre.

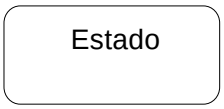


Figura # 34 Estado

Transiciones:

Se representan mediante flechas que salen del estado receptor hasta el estado destino y el nombre que se coloca en la flecha es el nombre del suceso que dio lugar a dicha transición, cada transición que sale de un estado corresponde a un suceso distinto, lo cual indica que no deben de existir sucesos duplicados dentro de un estado.

Evento/[condición]/acción

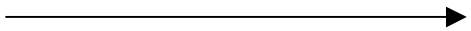


Figura # 35 Transición

Condiciones:

Una condición se puede pensar como una protección en las transiciones, debido a que si se cumple dicha condición la transición se dará y podrá pasar el objeto de un estado a otro, si dicha condición no se cumple inclusive podría pasar a otro estado mediante otra transición o quedarse en el estado receptor hasta que la condición se cumpla.

Acción:

Es una operación que va asociada a un suceso y se representa mediante una barra "/" y el nombre de la acción, después del nombre de la transición.

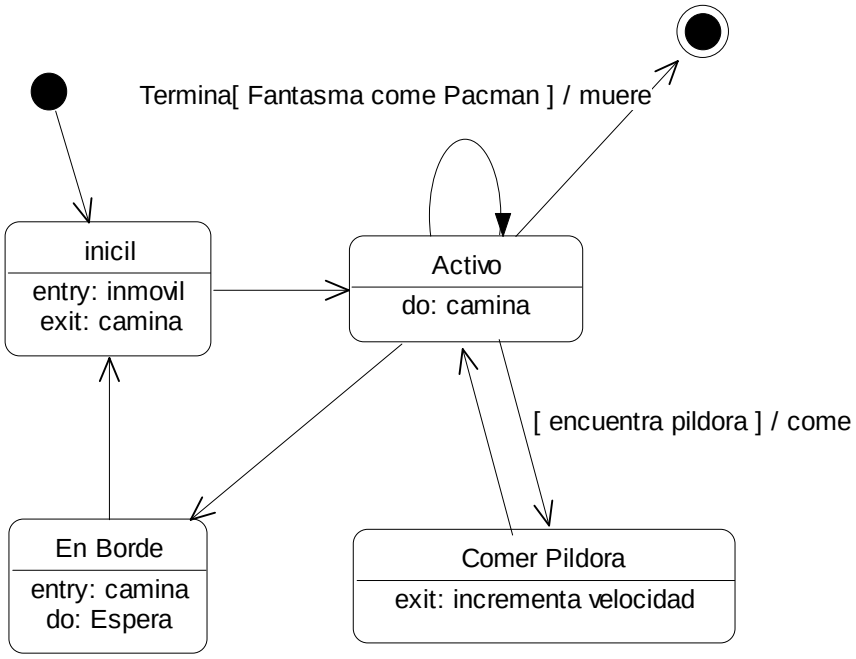


Figura # 36 Diagrama de Estados
del objeto Pacman.

Los pasos para construir el modelo funcional son los siguientes:

a) Identificación de los valores de entrada y de salida.
b) Construcción de diagramas de flujo de datos que muestren las dependencias funcionales.
c) Descripción de las funciones.
d) Identificación de restricciones.
e) Especificación de los criterios de optimización.

Modelo Funcional = Diagrama de flujo de datos + restricciones.

Mediante el modelo funcional se puede observar los resultados que se tienen de un calculo de valores, especificando solamente entradas y salidas de los valores, mas no como son calculados estos.
El modelo funcional consta básicamente de diagramas de flujo de datos. Los diagramas de flujo de datos son grafos que muestran el flujo de valores de datos a través de procesos los cuales modifican dichos valores para transformarlos en otros.

Los diagramas de flujo están compuestos de:

- Procesos
- Flujos de datos
- Actores
- Almacenes

Procesos: se representan mediante una elipse, como se muestra en la figura # 14, los procesos tienen como entrada datos, los cuales serán transformados, por lo cual un proceso es visto como un método de una operación aplicada a una clase de objetos.

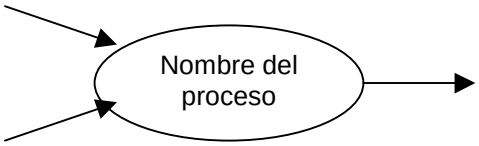


Figura # 37 Proceso

Flujo de datos: un flujo de datos conecta la salida de un proceso a la entrada de otro. Se representa en el diagrama por medio de una flecha, la cual puede llevar el nombre o el tipo de dato. Además de trasladar los datos a otros procesos, los flujos de datos pueden usarse para copiar un valor (figura # 15), realizar la composición de un agregado (figura #16) y así como su inverso (figura # 17).

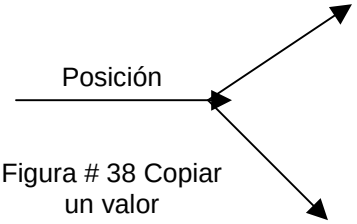


Figura # 38 Copiar
un valor

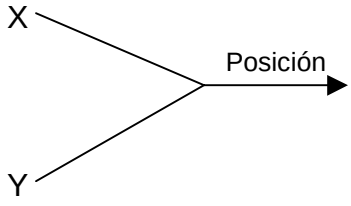


Figura # 39 Composición
de un valor

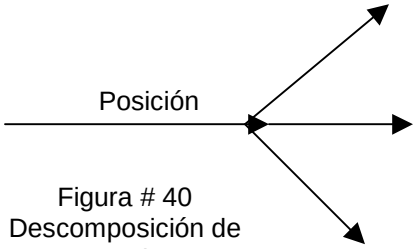


Figura # 40
Descomposición de
un valor

Actores: los actores son objetos que consumen y producen datos generando operaciones por si mismos, estos se encuentran siempre en las fronteras del diagrama indicando entradas y salidas de datos. Los actores también son llamados terminadores debido a que su función principal es hacer concluir el flujo de datos. En el diagrama son representados mediante rectángulos.

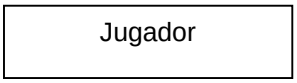


Figura # 41 Actor

Almacenes de datos: son objetos cuya tarea es permitir el almacenamiento y acceso de datos. Se representan en el diagrama mediante unas líneas paralelas que tienen el nombre del almacén.

B. DATOS

Figura # 42

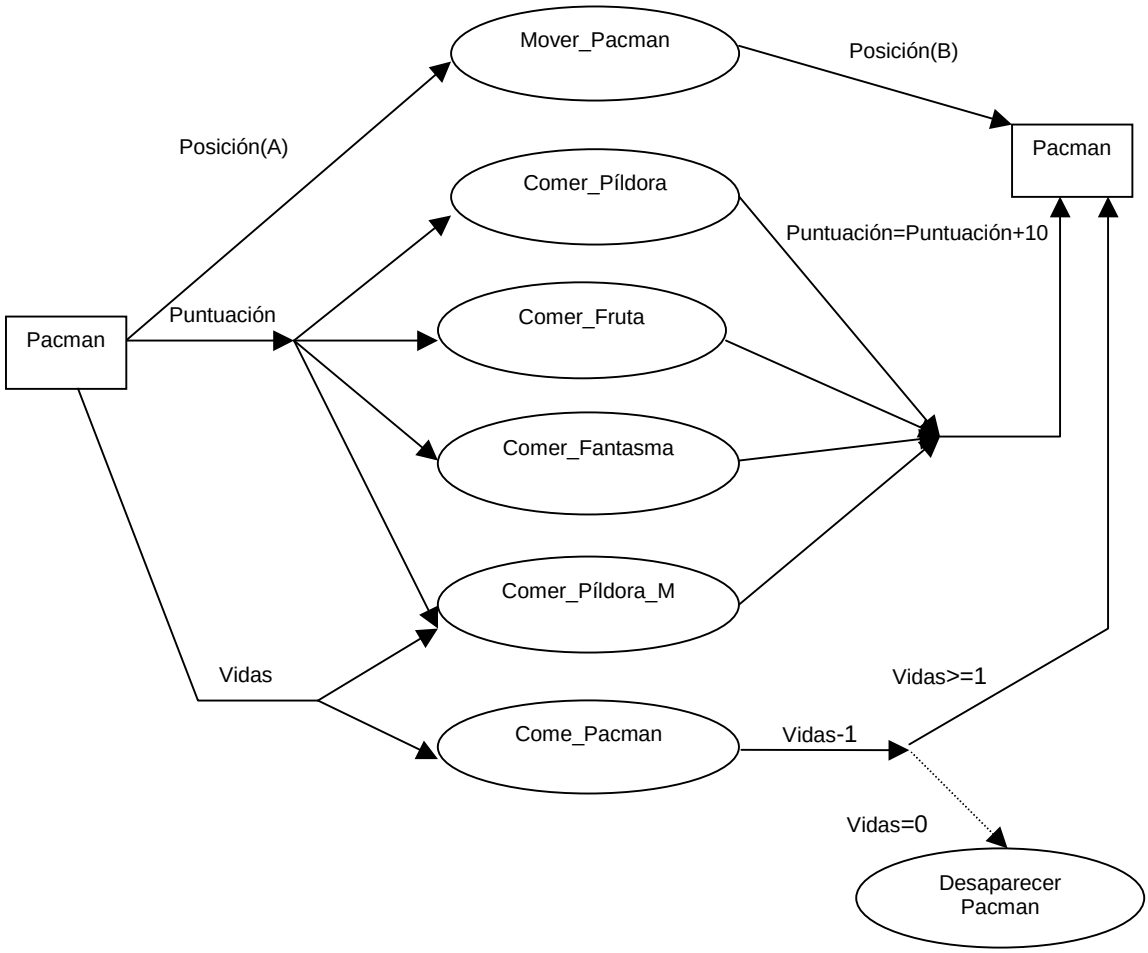


Figura # 43 Diagrama de Flujo del Pacman

Teniendo ya la representación del sistema en los tres modelos, se lleva a cabo una iteración general de cada uno de ellos.

a) Se deben comparar los tres modelos con la definición del problema y con el conocimiento en el dominio de la aplicación
b) Se añaden las operaciones claves descubiertas durante la preparación del modelo funcional.
c) Se hace una verificación entre clases, atributos, asociaciones y operaciones de tal manera que resulten congruentes.

d) Comprobar los modelos utilizando escenarios. Se desarrollan escenarios mas detallados, incluyendo condiciones de error.
e) Se realiza una iteración de los pasos anteriores, hasta considerar satisfactorio el análisis.

Documento de Análisis = Definición del problema + modelo de objetos + modelo dinámico + modelo funcional.

El documento abarca el análisis, el diseño y la implementación. Contiene una notación gráfica para expresar modelos orientados a objetos, es posible modelar, diseñar e implementar tanto a objetos en el dominio de la aplicación como a objetos en el dominio de la computadora.

- **Diseño del Sistema.**

El diseño del sistema es la estrategia de alto nivel para resolver el problema y construir una solución, incluye decisiones acerca de la organización del sistema (arquitectura del sistema) en subsistemas, la asignación de subsistemas a componentes de hardware y software y decisiones fundamentales conceptuales y de política que son las que constituyen el marco de trabajo para el diseño detallado.

Durante el **diseño del sistema** se añaden estructuras en el dominio de la solución. El modelo de diseño debe ser razonablemente eficiente y práctico a la hora de codificar, tratando detalles de bajo nivel que se omiten en el modelo de análisis.

La metodología de Rumbaugh, no abarca este aspecto, sin embargo sugiere algunas ideas generales

1. Organizar el sistema en subsistemas.
2. Identificar la concurrencia inherente en el problema.
3. Asignar los subsistemas a procesadores y a tareas.
4. Seleccionar la estrategia básica de implementación de los almacenes de datos, en términos de estructuras de datos, archivos y bases de datos.
5. Identificar los recursos globales y determinar los mecanismos para controlar el acceso a tales recursos.
6. Seleccionar una aproximación para implementar el control del software.
7. Consideraciones de condiciones de contorno.
8. Establecimiento de prioridades de compensación.

- **Diseño de Objetos.**

En el **Diseño de Objetos** se toman las decisiones necesarias para construir un sistema sin descender a los detalles particulares de un lenguaje o sistema de base de datos. El diseño de objetos es el comienzo de un desplazamiento con respecto al mundo real, en el modelo de análisis, aproximándose a la orientación a la computadora, necesaria para una implementación práctica.

Rumbaugh sugiere las siguientes etapas:

1. Obtención de las operaciones para el modelo de objetos a partir de los demás modelos.
2. Diseño de algoritmos para la implementación de las operaciones.

3. Optimización de las vías de acceso a los datos.
4. Implementar el control del software completando la aproximación seleccionada durante el diseño del sistema.
5. Ajuste de la estructura de clases para incrementar la herencia.
6. Diseño de la implementación de las asociaciones.
7. Se determina la representación exacta de los atributos que son objetos.
8. Empaquetamiento de las clases y asociaciones en módulos.

- **Implementación del Sistema.**

Durante la **implementación** se codifican, tanto las estructuras en el dominio de la aplicación como las estructuras en el dominio de la solución. La base que la sustenta es el **diseño de objetos**.

El código puede ser una simple transición de las decisiones de diseño a las características propias de un lenguaje.

- **Pruebas**

Es una actividad para determinar si el sistema esta siendo construido correctamente. Tanto la implementación como las pruebas son dos etapas que están involucradas durante el análisis y diseño. Lo que significa que el análisis, diseño, la implementación y las pruebas están relacionadas durante el ciclo de vida de un sistema, lo cual da paso a tener una forma incremental de desarrollo, debido a que en cada etapa se pueden agregar características que tal vez en un primer nivel de abstracción se dejaron fuera.

Un proceso continuo.

Este proceso define en su fase inicial una serie de objetos, continuando extendiendo y refinando estos objetos en fases posteriores, la separación de fases en el ciclo de vida resulta difusa, por lo que el modelo desarrollado durante el análisis se utiliza para el diseño y para la implementación, logrando el refinamiento del modelo con niveles progresivamente mas detallados, en vez de hacer una transformación de una representación a otra. Por lo tanto el proceso carece de *costuras* ya que no existen discontinuidades en las cuales una notación de una fase tenga que ser substituida por otra notación diferente en la fase siguiente.

Iterativo mas que secuencial.

Aun cuando la descripción de esta técnica es lineal, el proceso de desarrollo real es iterativo. Su carencia de *costuras* hace más fácil el repetir los pasos de desarrollo con grados de detalle cada vez más finos, donde, cada iteración añade o clarifica características en vez de modificar un trabajo ya realizado, por lo tanto, existe menos posibilidades de introducir incongruencias y errores.