|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | **8** |
|  | TEMA |

Diseño de la arquitectura

[8.1] Arquitectura del *software*

[8.2] Géneros arquitectónicos

[8.3] Estilos arquitectónicos

[8.4] Diseño arquitectónico

[8.5] Evolución de los diseños alternativos para la arquitectura

[8.6] Mapeo de la arquitectura con el uso del flujo de datos

[8.7] Referencias

Esquema



Ideas clave

8.1. Arquitectura del *software*

La **arquitectura del *software*** proporciona una visión del sistema a construir: estructura y organización de los componentes, sus propiedades y conexiones.

La arquitectura del *software* permite analizar la efectividad del diseño para cumplir los requisitos establecidos, considerar alternativas arquitectónicas, reducir riesgos asociados con la construcción del *software* y facilita la comunicación entre todos los participantes en el desarrollo del *software*.

8.2. Géneros arquitectónicos

El género arquitectónico dicta el enfoque específico para el *software* a construir. Algunos géneros arquitectónicos son: inteligencia artificial, comunicaciones, contenido de autor, entretenimiento, financiero, medicina, militar, sistemas operativos, utilidades, etc.

8.3. Estilos arquitectónicos

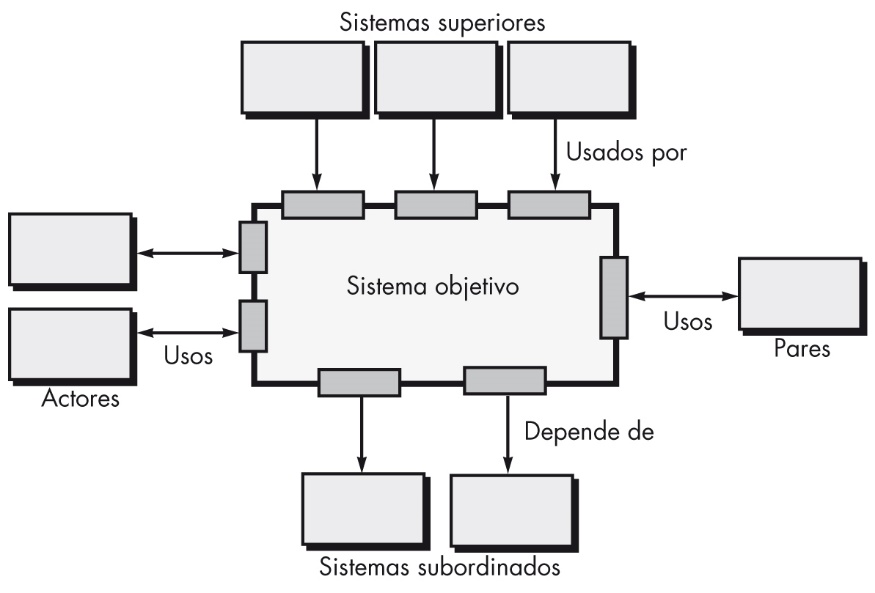
La mayoría de los sistemas se pueden clasificar en alguna de las siguientes tipologías de estilos arquitectónicos:

* Arquitecturas centradas en los datos.
* Arquitecturas de flujo de datos.
* Arquitecturas de llamada y retorno:
* Arquitectura de programa principal y subprogramas.
* Arquitectura de llamada a procedimiento remoto.
* Arquitecturas orientadas a objetos.
* Arquitecturas en capas.

8.4. Diseño arquitectónico

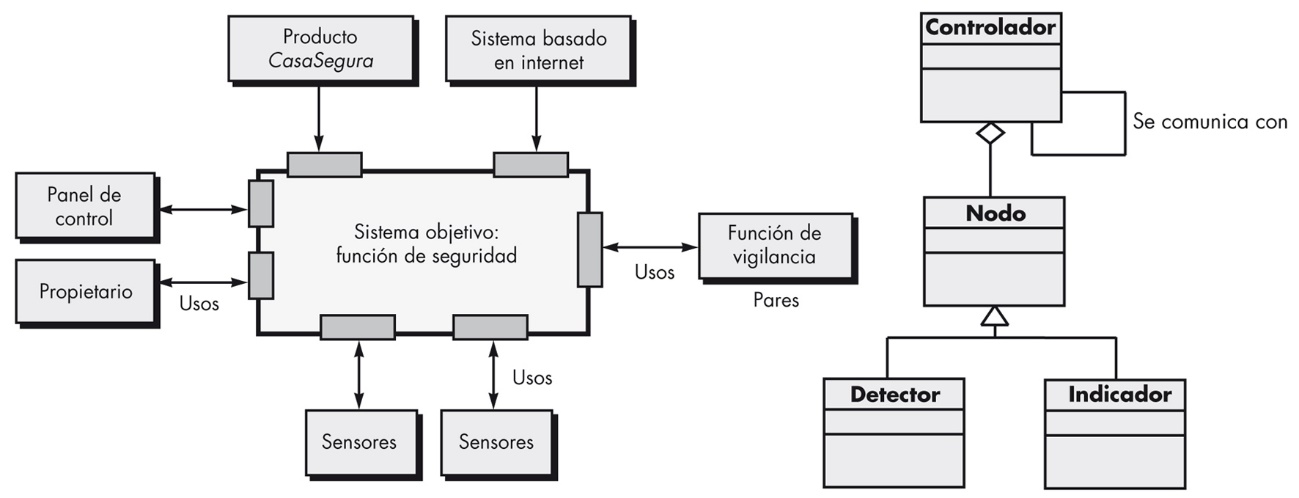
El **diseño arquitectónico del sistema** se debe realizar en su contexto en el que interactuará con los diferentes actores y con sistemas que pueden ser superiores, subordinados o iguales.

La comunicación con cada una de estas entidades externas se realiza mediante una interfaz.



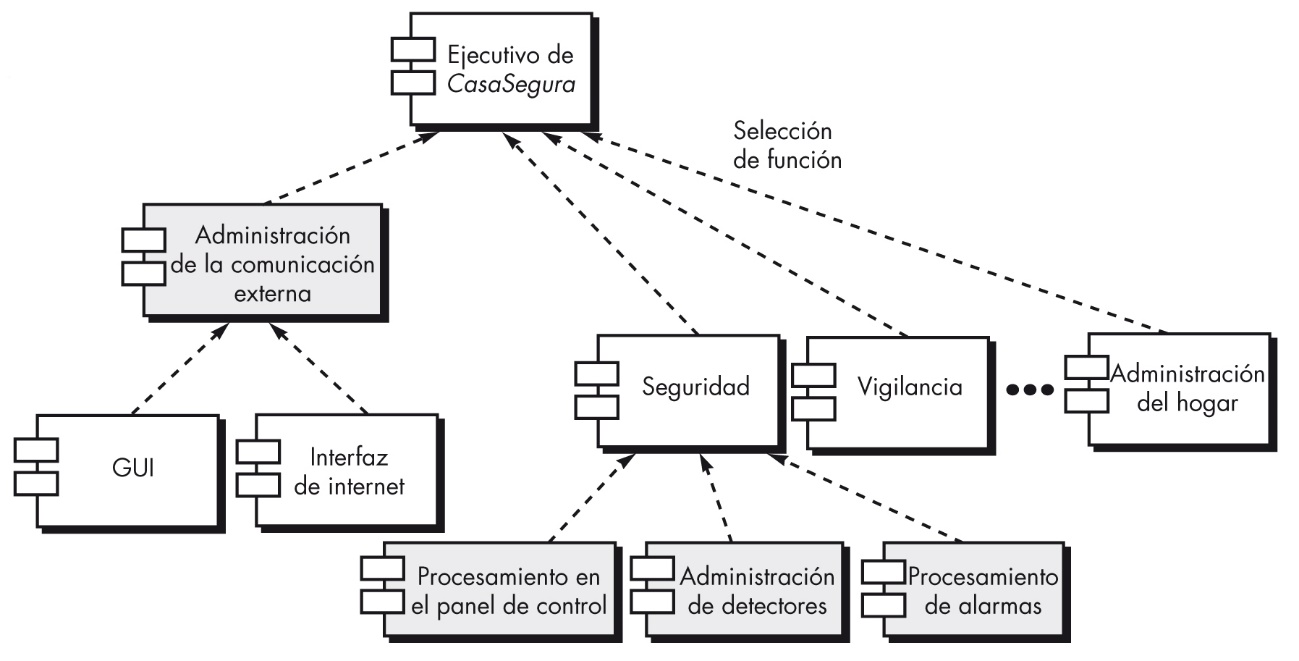
Fuente: Pressman, R. S. (2010). *Ingeniería del software.   
Un enfoque práctico* (7ª ed.) (p. 218). México: McGraw Hill.

En el diseño de la arquitectura del sistema se representan los arquetipos, que son clases o patrones que representan abstracciones de importancia crítica.



Fuente: Pressman, R. S. (2010). *Ingeniería del software.   
Un enfoque práctico* (7ª ed.) (p. 218-219). México: McGraw Hill.

El diseño de la arquitectura se debe ir refinando hacia los componentes que constituirán el sistema.



Fuente: Pressman, R. S. (2010). *Ingeniería del software.   
Un enfoque práctico* (7ª ed.) (p. 220). México: McGraw Hill.

8.5. Evaluación de los diseños alternativos para la arquitectura

El diseño da como resultado varias alternativas de arquitectura que deben ser evaluadas para determinar cuál es la más adecuada para resolver el problema. Para esta evaluación se puede utilizar:

* **Método de la negociación para analizar la arquitectura**: en el que de forma iterativa:

1) Se desarrolla el conjunto de casos de uso del sistema.

2) Se obtienen los requisitos, restricciones y descripción del entorno.

3) Se describen los estilos o patrones arquitectónicos.

4) Se evalúa independientemente cada atributo de calidad: fiabilidad, seguridad, mantenibilidad, portabilidad, etc.

5) Identificación de la criticidad de los atributos de calidad.

6) Crítica de las arquitecturas candidatas.

* **Complejidad arquitectónica**: evalúa la complejidad de la arquitectura en base a las dependencias de flujo de información o de control de los componentes.

8.6. Mapeo de la arquitectura con el uso del flujo de datos

El diseño estructurado propone la **transición del Diagrama de Flujo de Datos** (DFD) del sistema a la arquitectura *software* con los siguientes pasos:

1. Se establece el tipo de flujo de información.
2. Se determinan las fronteras del flujo.
3. Se mapea el DFD en la estructura del programa.
4. Se define la jerarquía de control.
5. Se refina la estructura resultante.
6. Se elabora la descripción arquitectónica.

8.7. Referencias

Pressman, R. (2010). *Ingeniería del software* (7ª ed.). México: McGrawHill.

Lo + recomendado

No dejes de leer…

**Visión filosófica de los componentes**

Artículo que presenta otro punto de vista de los componentes desde una visión más filosófica.

Accede al artículo desde el aula virtual o a través de la siguiente dirección web:

<http://www.existentialprogramming.com/2010/05/hole-for-every-component-and-every.html>

No dejes de ver…

**Arquitectura *software***

Este vídeo muestra la relación entre la arquitectura de *software* y el diseño de *software*. Presenta como los arquitectos construyen modelos que les permiten razonar sobres sus diseños y reducir los potenciales riesgos.



Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:

<https://www.youtube.com/watch?v=Rn1g6V-vlHw>

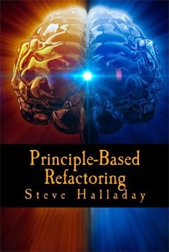
+ Información

A fondo

**Visión general de la ingeniería del *software***

Halladay, S. (2012). *Principle-Based Refactoring: Learning Software Design Principles by Applying Refactoring Rules*. Principle Publishing.

Este libro explica un conjunto de reglas de refactorización de *software* y las vincula a los principios básicos que impulsan el diseño de *software* robusto. En el libro se identifican ocho principios fundamentales de diseño y unas cincuenta reglas de refactorización que ilustran los principios. Cada regla tiene una descripción resumida, la explicación y ejemplos de aplicación de las reglas. También se analiza el proceso de refactorización incluyendo estrategias de prueba para verificar la calidad del código.



**Estándar IEEE 1016-2009 sobre el diseño *software***

Estándar que recoge la representación del diseño de *software* para la comunicación de la información de diseño a los participantes en un proyecto. En el estándar se especifica los requisitos para los lenguajes de diseño.

Accede al artículo desde el aula virtual o a través de la siguiente dirección web:

[http://zeynepaltan.info/IEEE%20SDD-1016-2009.pdf](http://zeynepaltan.info/IEEE SDD-1016-2009.pdf)

Recursos externos

Herramienta *online* para el modelado de diagramas de componentes

Gliffy es una herramienta *online* gratuita que permite el diseño de diagramas de componentes de un modo sencillo.

Accede a la página desde el aula virtual o a través de la siguiente dirección web:

<http://www.gliffy.com/>

Test

**1.** La arquitectura del *software*:

A. Determina el lenguaje de programación que se utilizará en la construcción del *software*.

B. Proporciona una visión la estructura y los componentes a construir.

C. Es una alternativa a la fase de diseño del *software*.

D. Se genera automáticamente a partir del diagrama de estado del sistema.

**2.** La arquitectura del *software*:

A. Se elabora con independencia de los requisitos del sistema.

B. Dificulta la comunicación del equipo de desarrollo del *software*.

C. Reduce los riesgos asociados con la construcción del *software*.

D. Todas son correctas.

**3.** Un género arquitectónico:

A. Determina el enfoque específico para el *software* a construir.

B. Determina el número de miembros que tendrá el equipo de desarrollo.

C. Determina el mantenimiento futuro que tendrá el sistema.

D. Determina el número de ciclos de refinamiento que habrá que llevar a cabo para el diseño de la arquitectura.

**4.** Las arquitecturas de flujo de datos se aplican cuando:

A. El centro de la arquitectura son los datos a los que acceden los componentes *software*.

B. El programa principal llama a subprogramas para obtener los datos.

C. Puede representarse con un Diagrama de Flujo de Datos (DFD).

D. Los datos de entrada se transforma en datos de salida mediante componentes *software*.

**5.** Son arquitecturas de llamada y retorno:

A. Las arquitecturas de programa principal/subprograma y las arquitecturas de llamada a procedimiento remoto.

B. Las arquitecturas orientadas a objetos y las arquitecturas orientadas a datos.

C. Las arquitecturas de flujo de datos y las arquitecturas en capas.

D. Las arquitecturas jerárquicas y las arquitecturas secuenciales.

**6.** Un arquetipo es:

A. Un componente que se marca como obsoleto durante un ciclo refinamiento en la construcción de la arquitectura.

B. Una clase o patrón que representa una abstracción de importancia crítica en la arquitectura.

C. Un subconjunto de componentes de la arquitectura que se agrupan en arquetipos para facilitar su gestión.

D. Un tipo estándar de arquitectura basada en el conocimiento que da la experiencia de diseños anteriores.

**7.** El método de la negociación para analizar la arquitectura se utiliza para:

A. Acotar los límites del sistema con el cliente.

B. Simplificar la lista de requisitos iniciales del *software*.

C. Acelerar la fase de diseño de la arquitectura en caso de conflictos.

D. Evaluar las alternativas de arquitectura y determinar la más adecuada.

**8.** La complejidad arquitectónica evalúa la complejidad de la arquitectura en base a:

A. El tamaño de un sistema.

B. El número de componentes.

C. Las dependencias entre los componentes.

D. Algoritmos matemáticos de cálculo de complejidad de ciclos.