DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

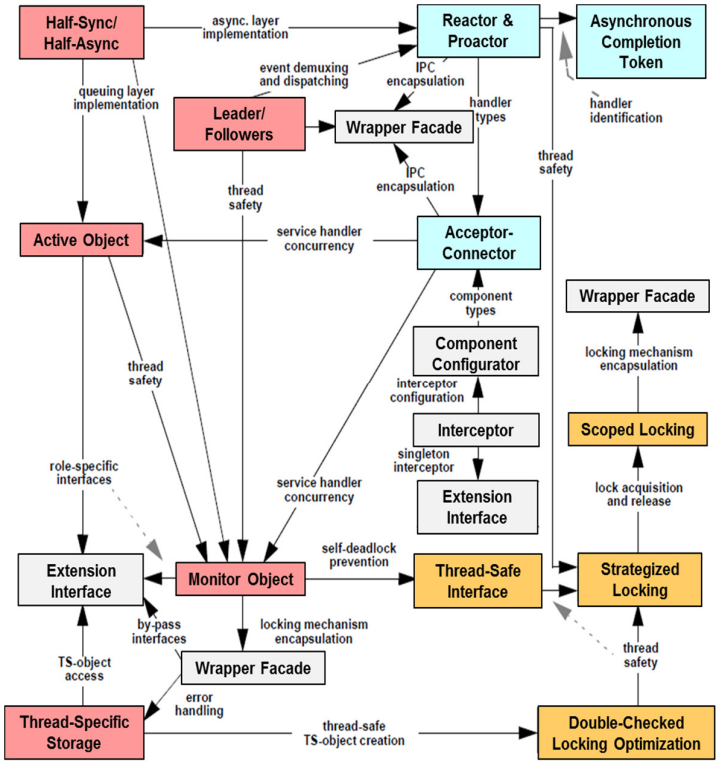
Middleware – Patrones de diseño

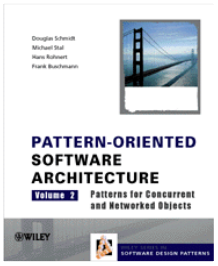
© Fernando Berzal, berzal@acm.org

Middleware – Patrones de diseño

- Patrones de diseño
- Frameworks
- Diseño de brokers
  - Reactor
  - Half-Sync/Half-Async
  - Leader/Followers
  - Acceptor-Connector

Middleware – Patrones de diseño



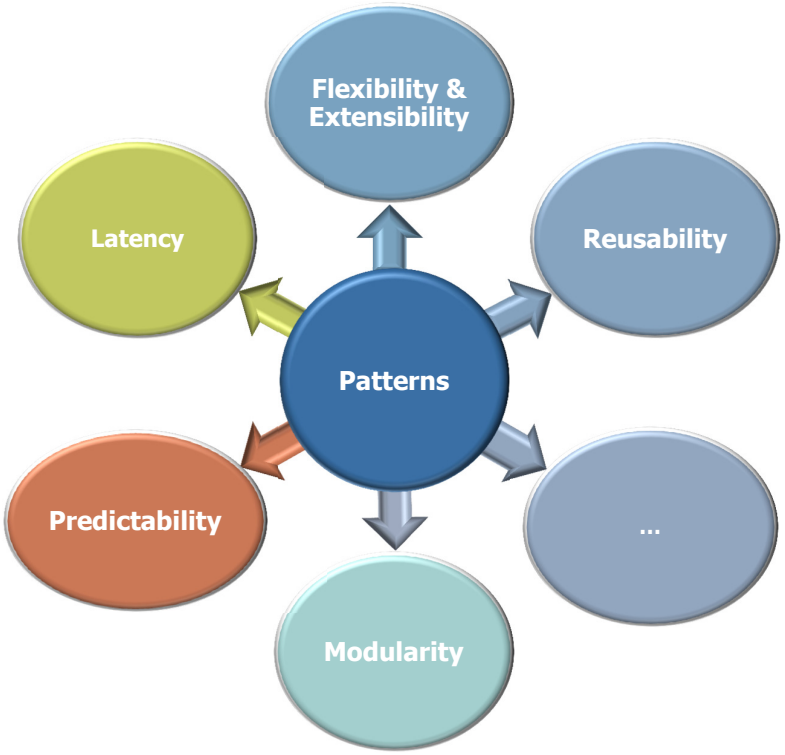


Middleware – Patrones de diseño

To slay the dragons of accidental & inherent complexity associated with concurrent & networked software we need to codify *proven software experience*

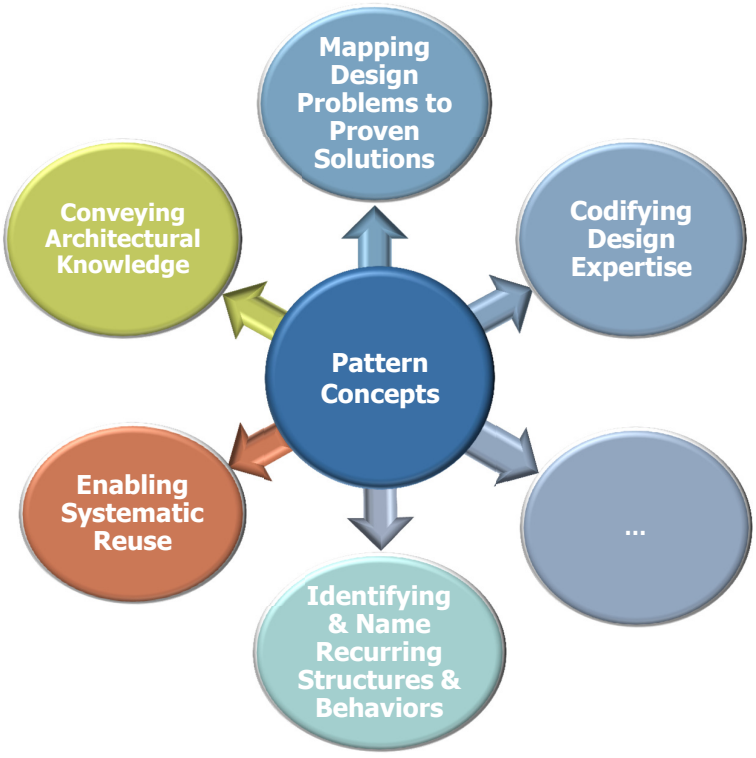
Middleware – Patrones de diseño

Middleware – Patrones de diseño



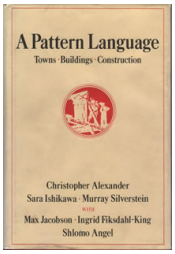
Middleware – Patrones de diseño

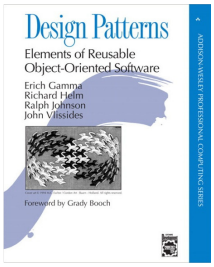
Middleware – Patrones de diseño

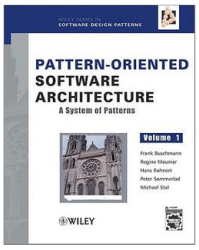


Middleware – Patrones de diseño

Descripción de patrones de diseño

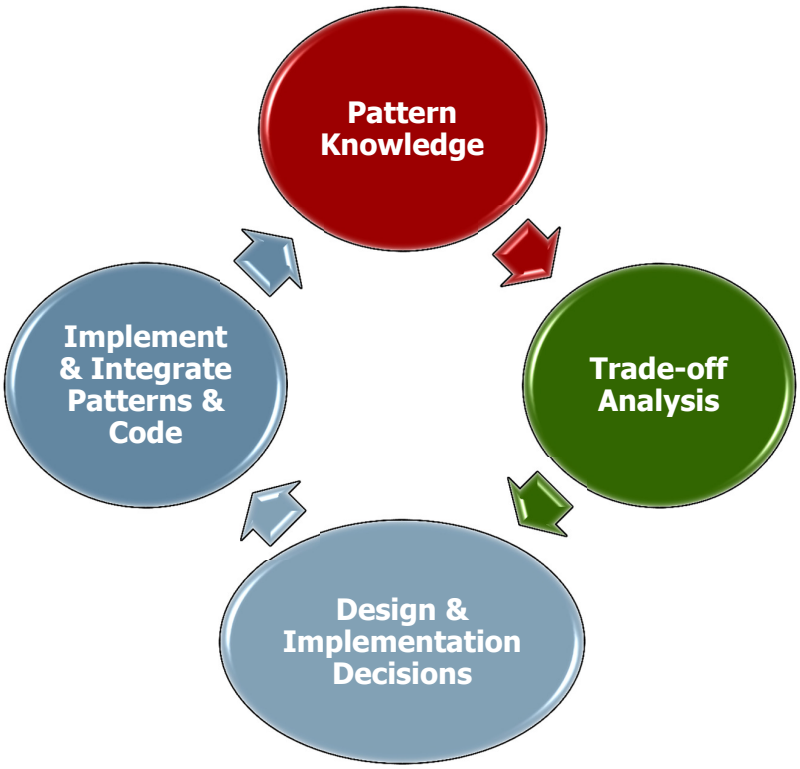




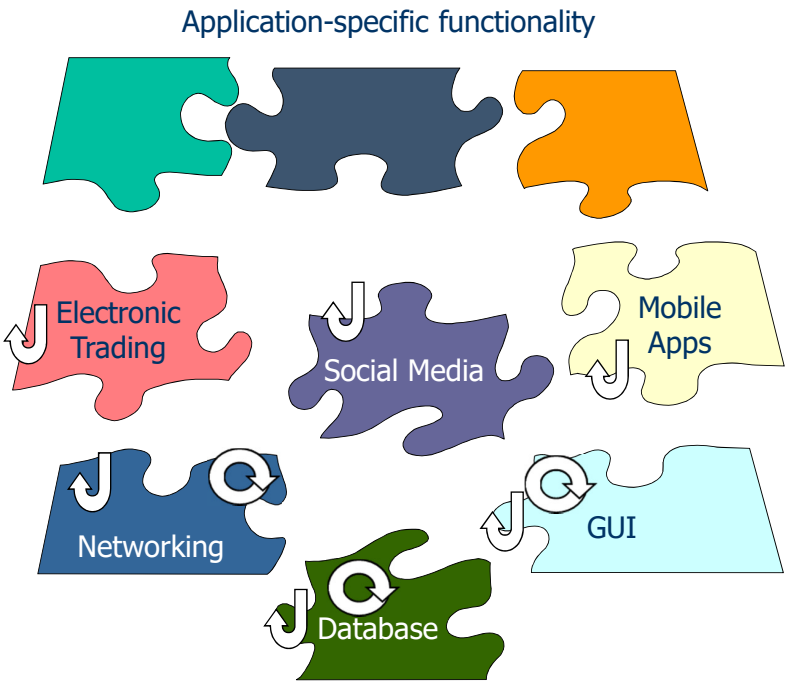


- **Name** & statement of pattern **intent**
- **Problem** addressed by pattern
  - Including "forces" & "applicability"
- **Solution**
  - Visual & textual descriptions of pattern structure & dynamics
- **Examples & Implementation** guidance
  - Source code snippets
- **Consequences**
  - Pros & cons of applying the pattern
- **Known uses** ("Rule of three")
- **Related patterns** (tradeoffs between alternative patterns)

Middleware – Patrones de diseño



Middleware – Frameworks



Frameworks are integrated sets of software components that collaborate to provide reusable architectures for families of related applications

Application domains

Infrastructure domains

Middleware – Frameworks

Beneficio clave

Reutilización sistemática

- Diseño
- Código

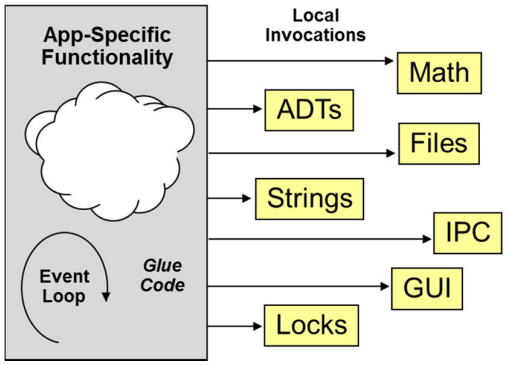
Patterns & frameworks help avoid "reinventing the wheel" for many accidental & inherent complexities of concurrent & networked software

Problem	Pattern
Encapsulating low-level system functions to enhance portability	Wrapper Façade
Demuxing broker core events efficiently	Reactor
Managing broker connections efficiently	Acceptor-Connector
Enhancing broker scalability by processing requests concurrently	Half-Sync/ Half-Async
Efficiently synchronize the Half-Sync/Async request queue	Monitor Object
…	…
Defining the broker's base-line architecture	Broker

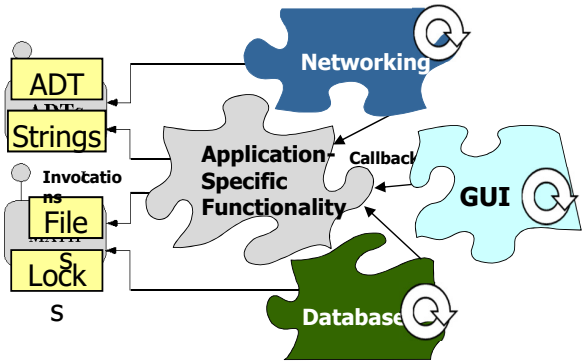
Middleware – Frameworks

Técnicas de reutilización

- Bibliotecas de clases



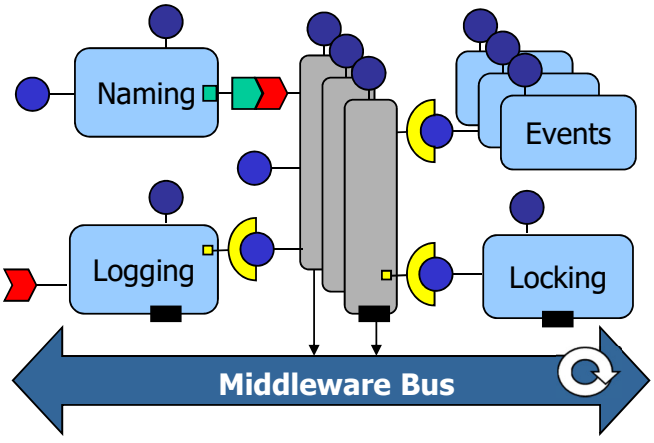
- Frameworks



Middleware – Frameworks

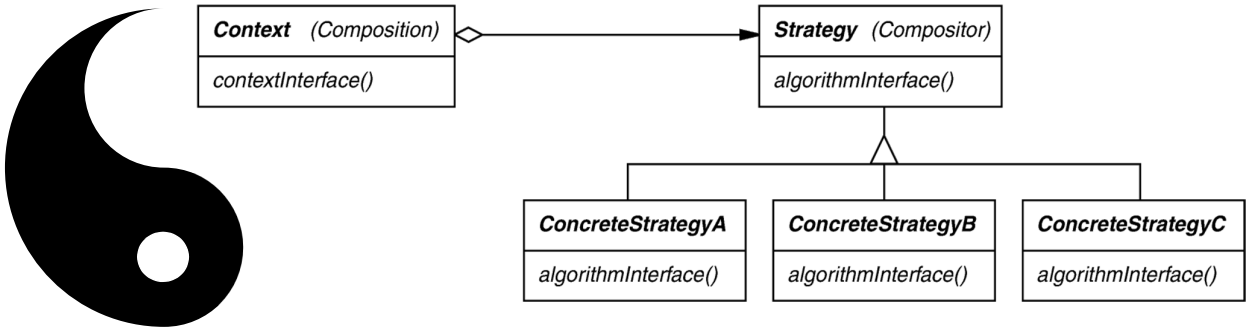
Técnicas de reutilización

Componentes & SOA [Service-Oriented Architecture]



Middleware – Frameworks

Black-box frameworks

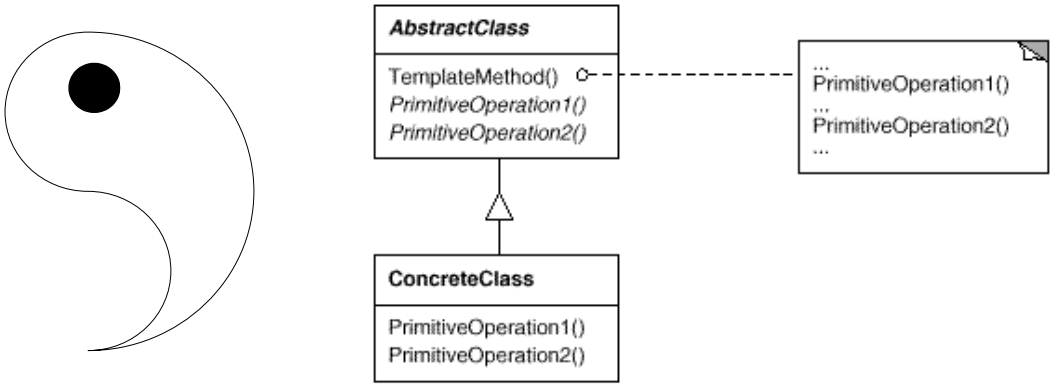


Parametrización y ensamblado de objetos:

- Extensibilidad por composición.
- Patrones de diseño: Strategy & Decorator.

Middleware – Frameworks

White-box frameworks

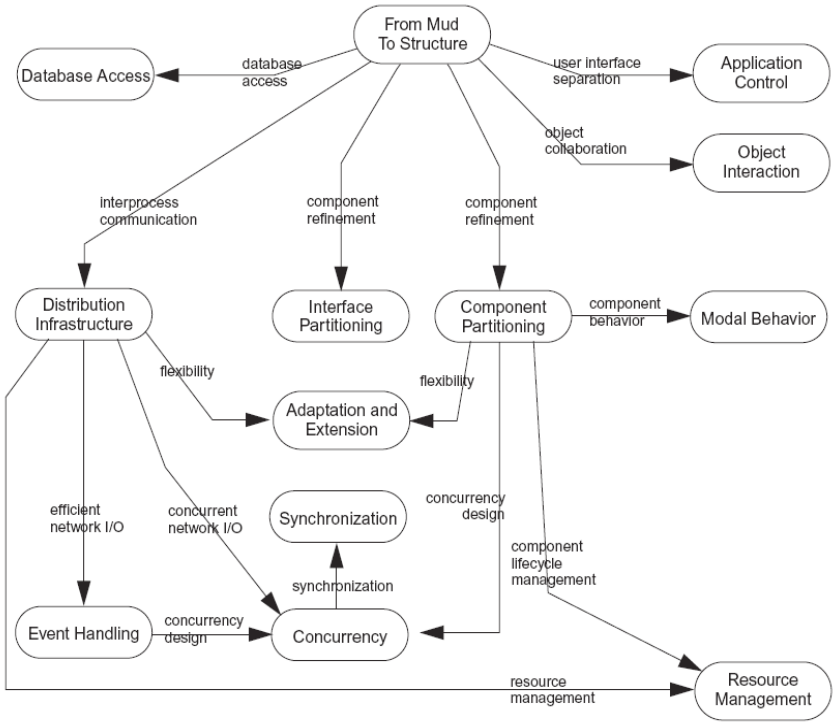


Creación de subclases y redefinición de métodos:

- Extensibilidad por herencia.
- Patrones de diseño: Template Method & State.

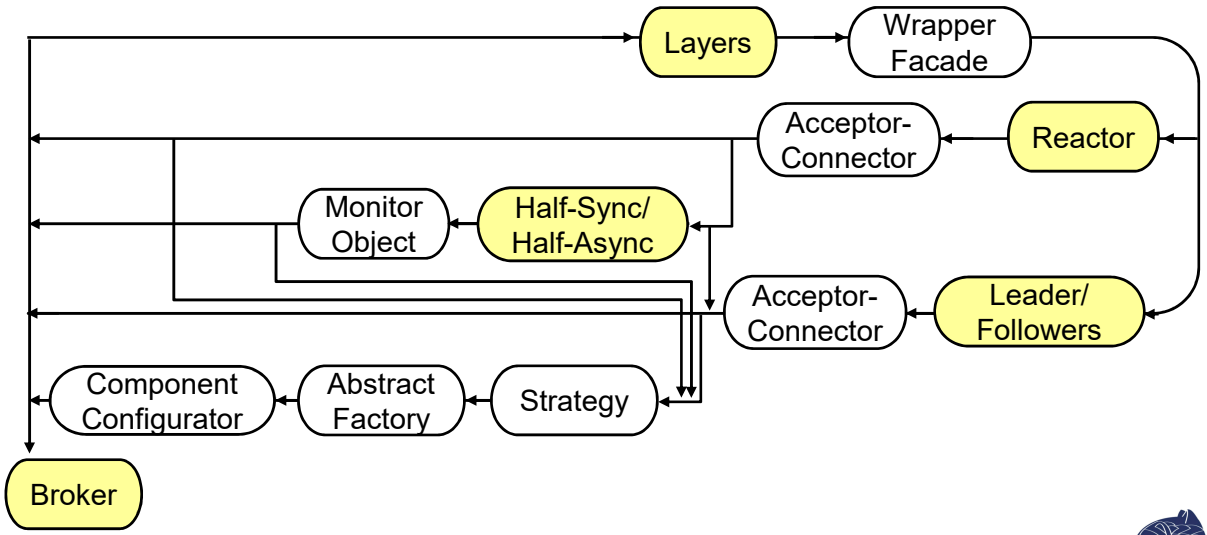
Middleware – Pattern Languages

POSA



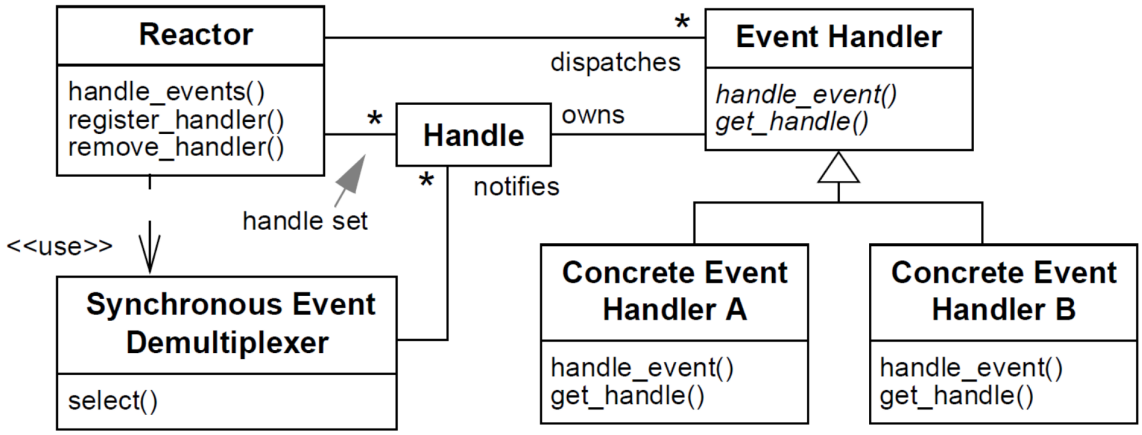
Middleware – Pattern Languages

Diseño de un broker



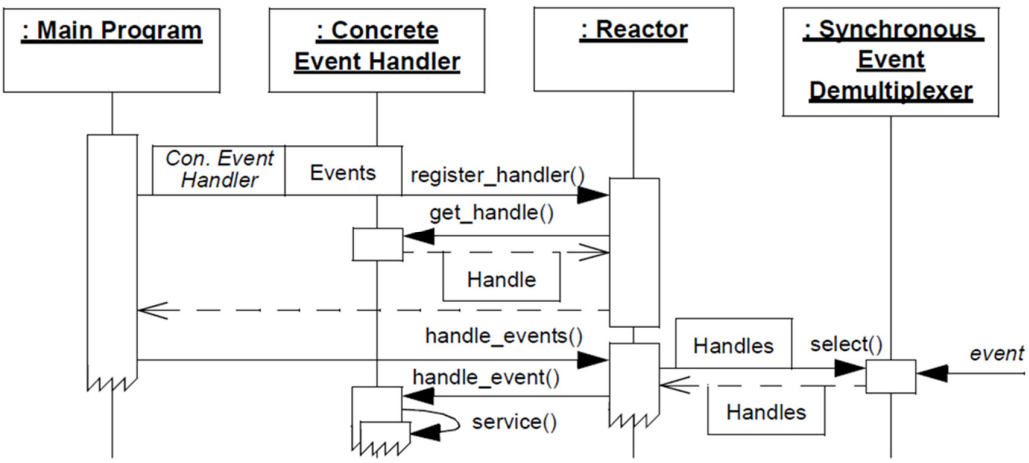
Reactor

Simplificación del manejo de eventos



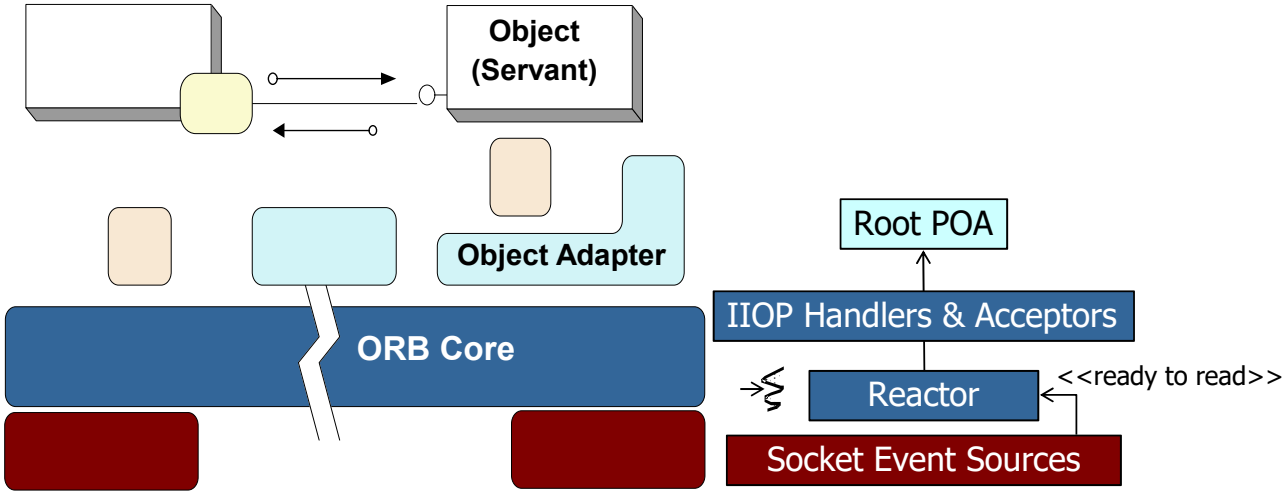
Reactor

Simplificación del manejo de eventos

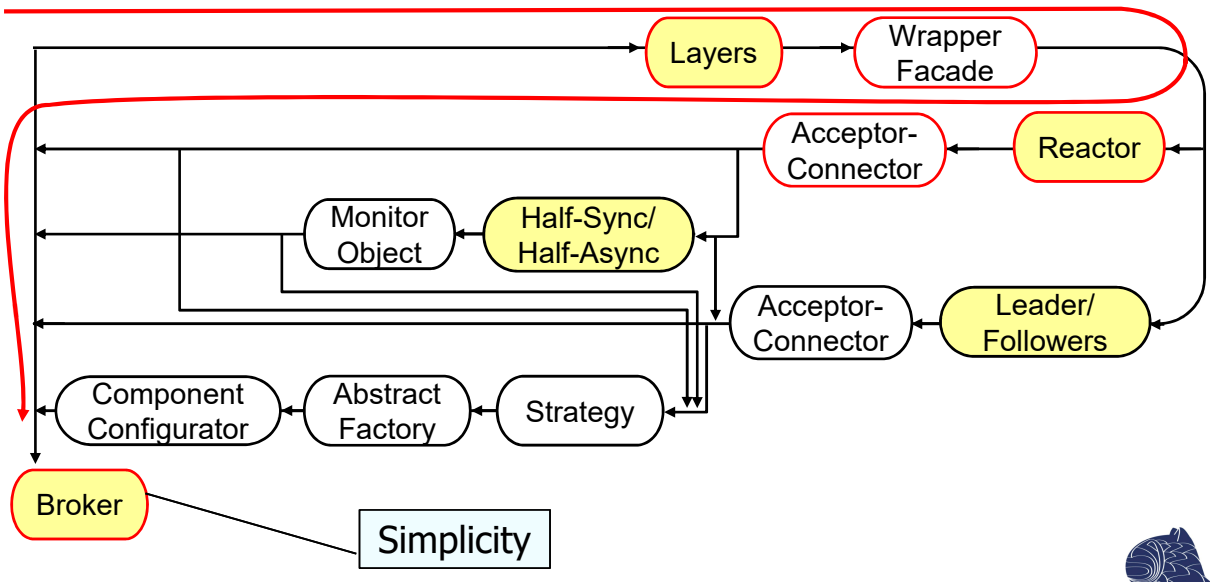


Reactor

Simplificación del manejo de eventos

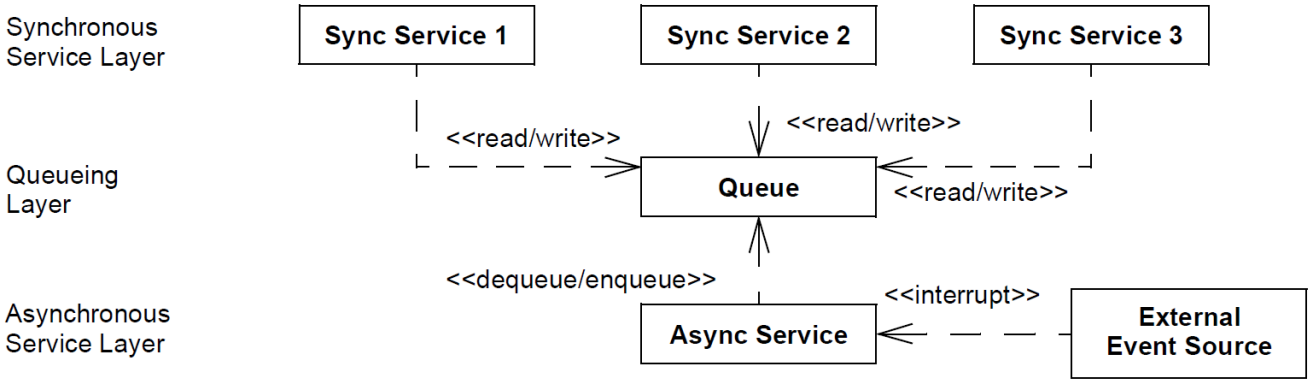


Reactor



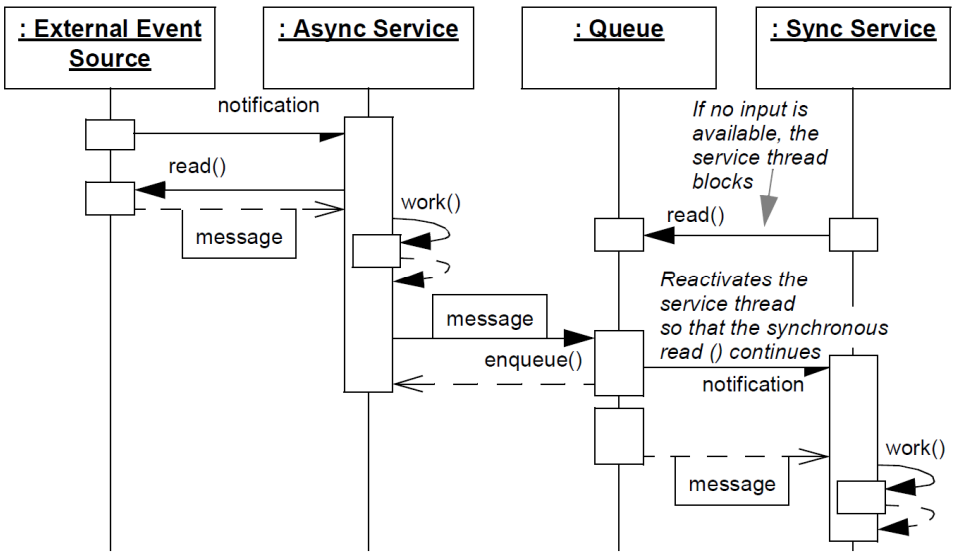
Half-Sync/Half-Async

Mejora de la escalabilidad



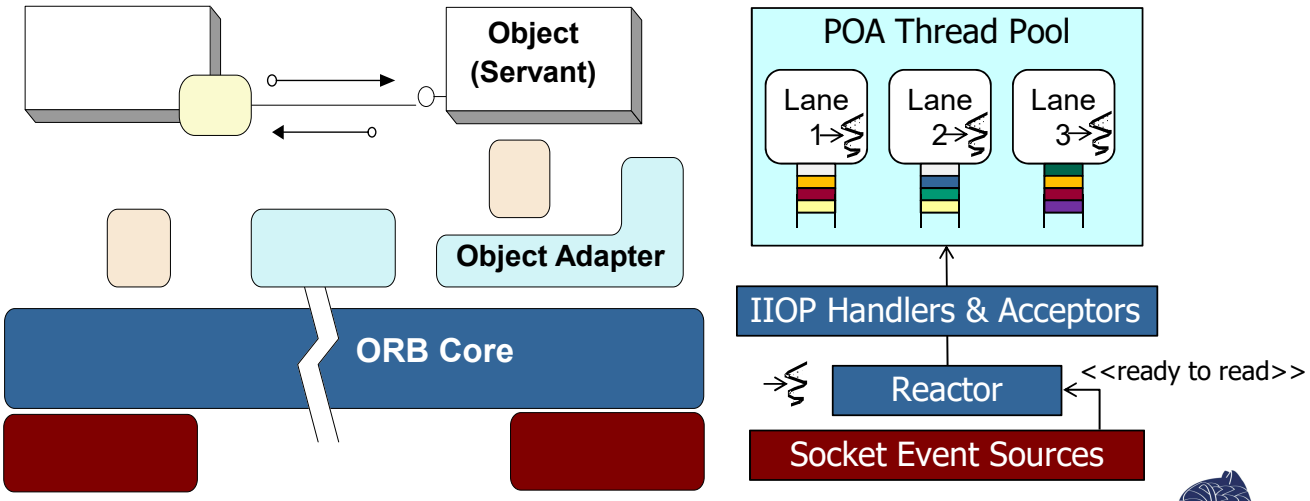
Half-Sync/Half-Async

Mejora de la escalabilidad

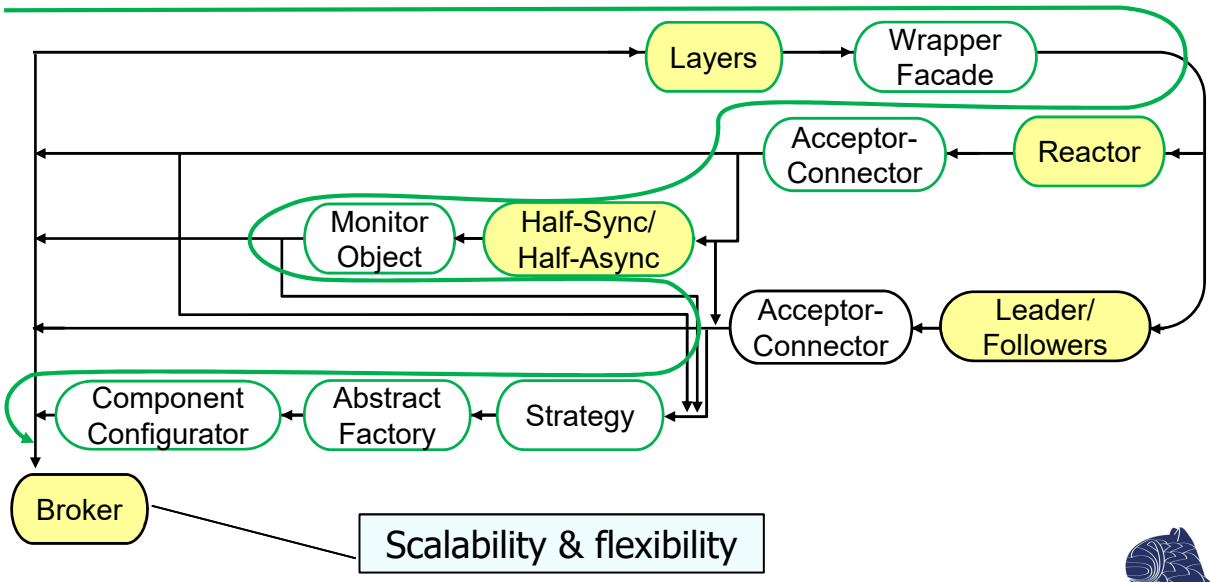


Half-Sync/Half-Async

Mejora de la escalabilidad:
Múltiples solicitudes en paralelo

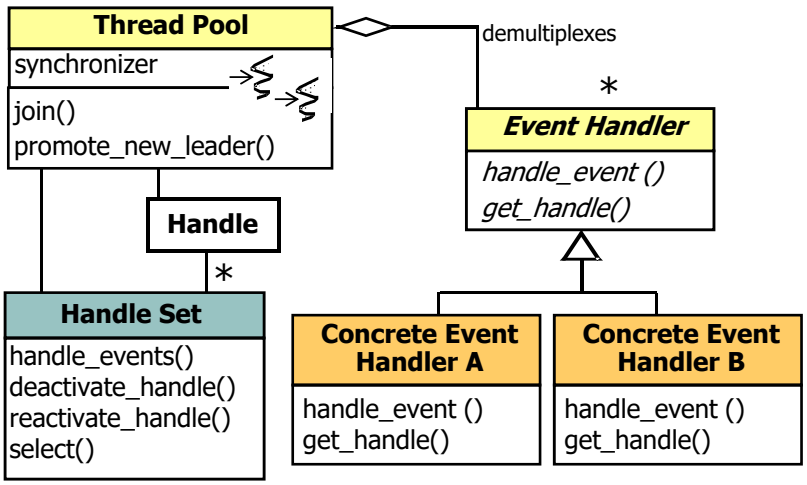


Half-Sync/Half-Async



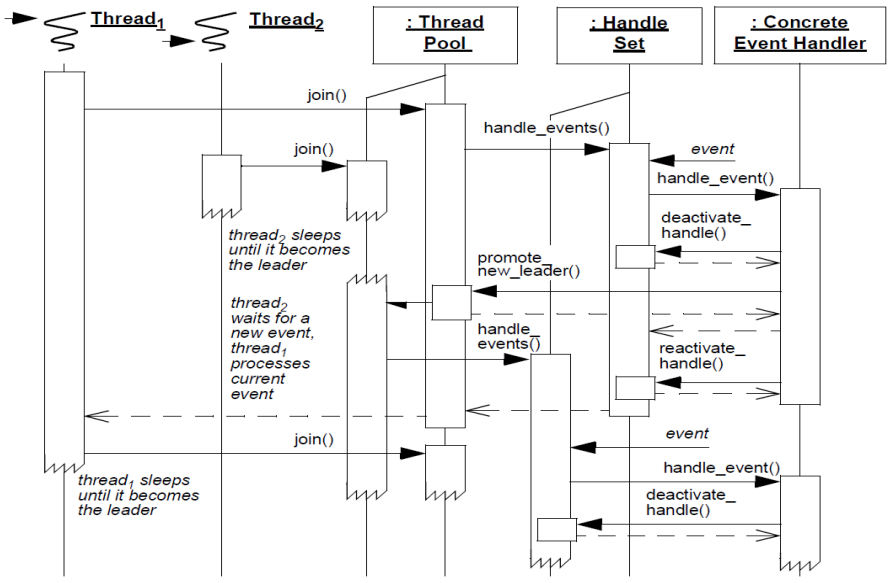
Leader/Followers

Reducción de jitter & overhead



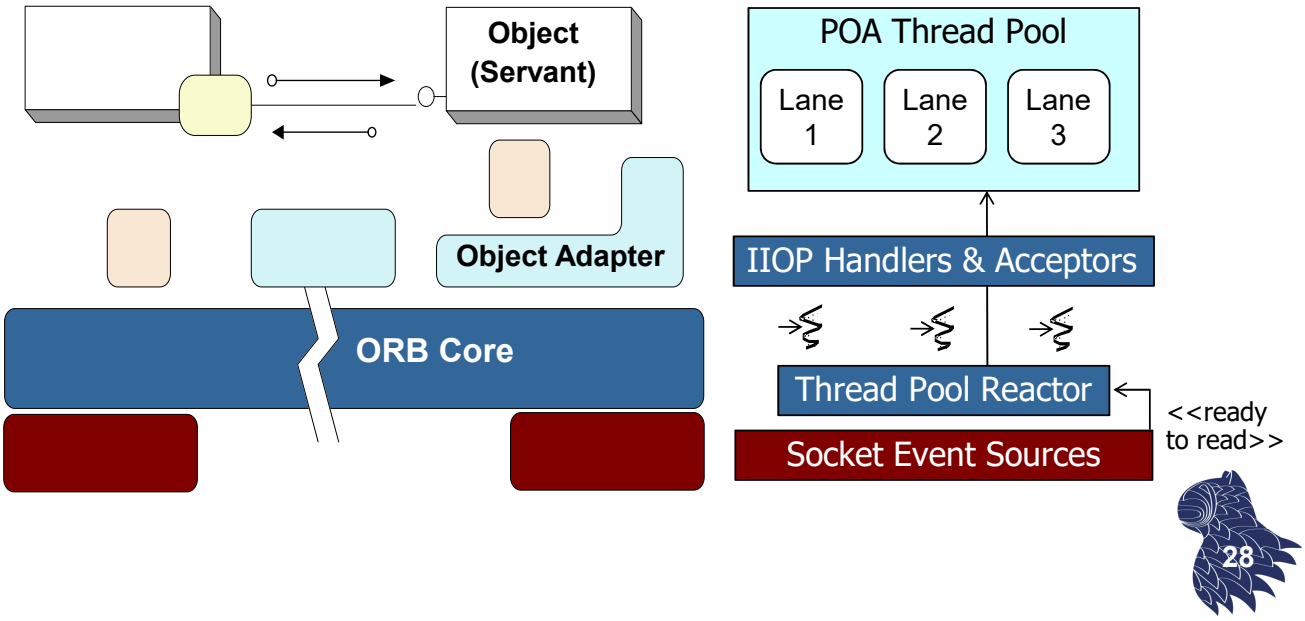
Leader/Followers

Reducción de jitter & overhead

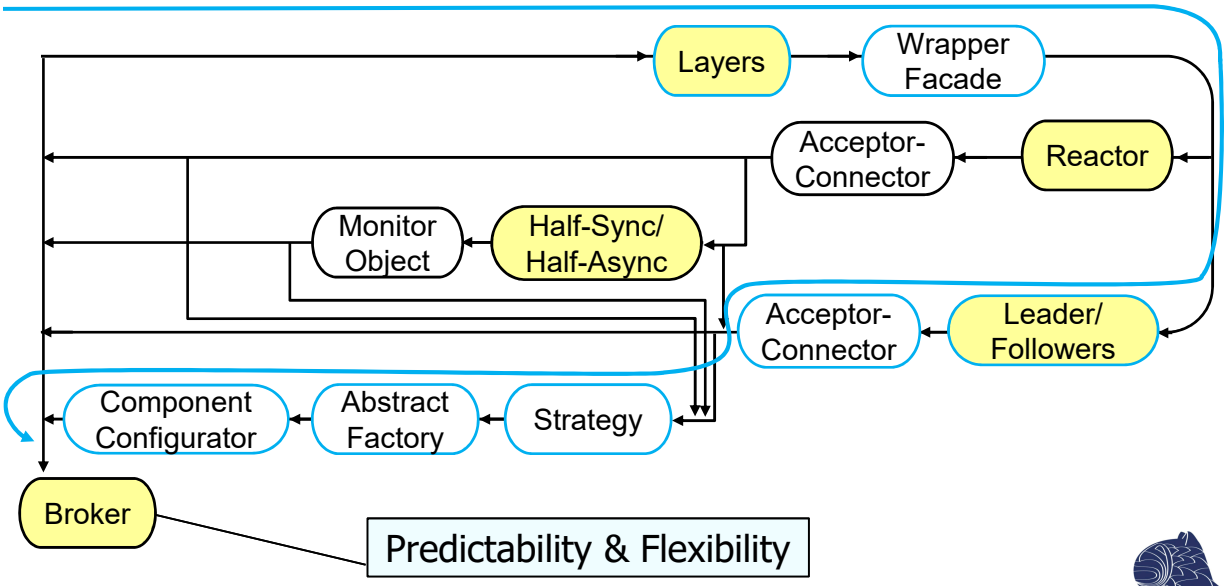


Leader/Followers

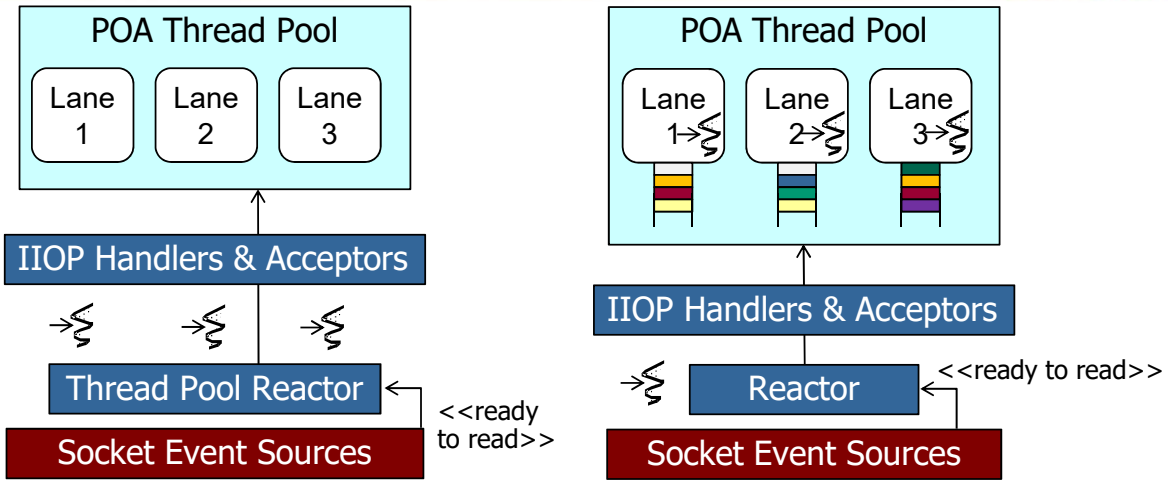
Reducción de jitter & overhead



Leader/Followers



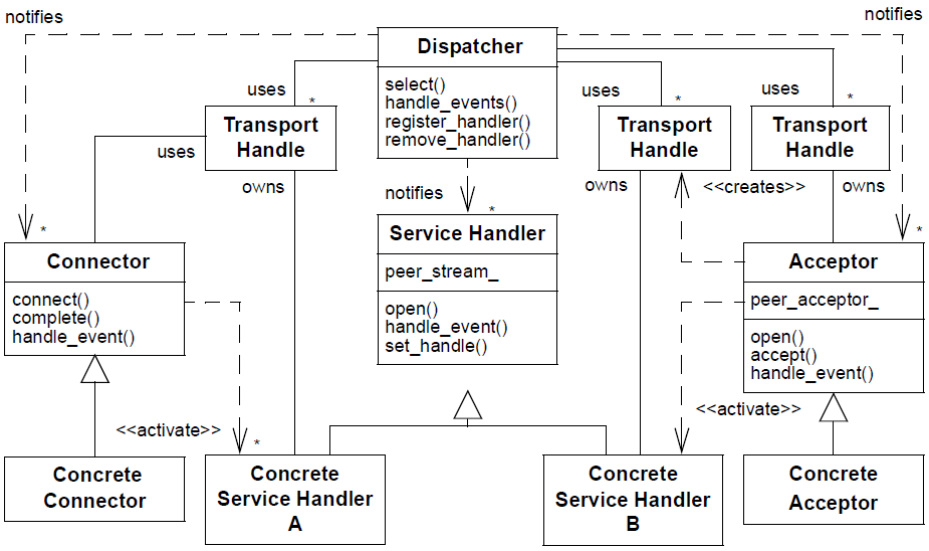
Decisiones de diseño



	Leader/Followers	Half-Sync/Half-Async
Features	Poor	Good
Scalability	Poor	Good
Efficiency	Good	Poor
Optimizations	Good	Poor
Priority inversion	Good	Poor

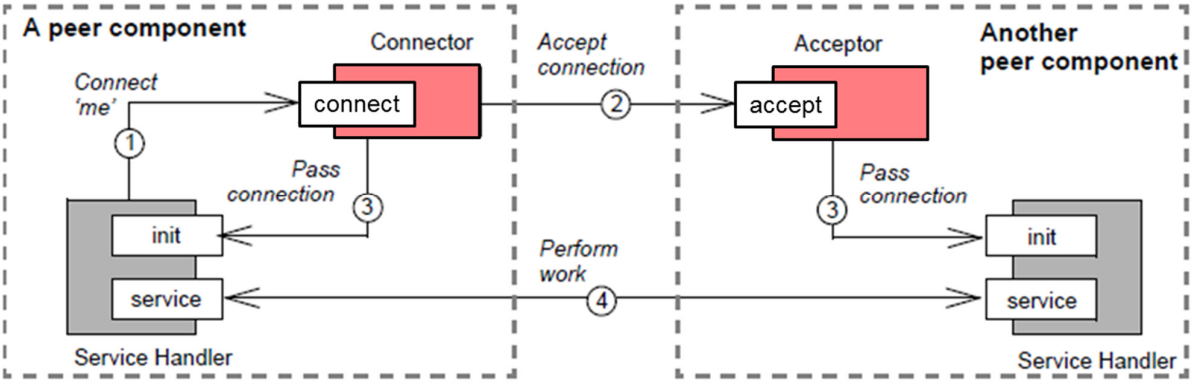
Acceptor-Connector

Separación conexión/inicialización vs. procesamiento



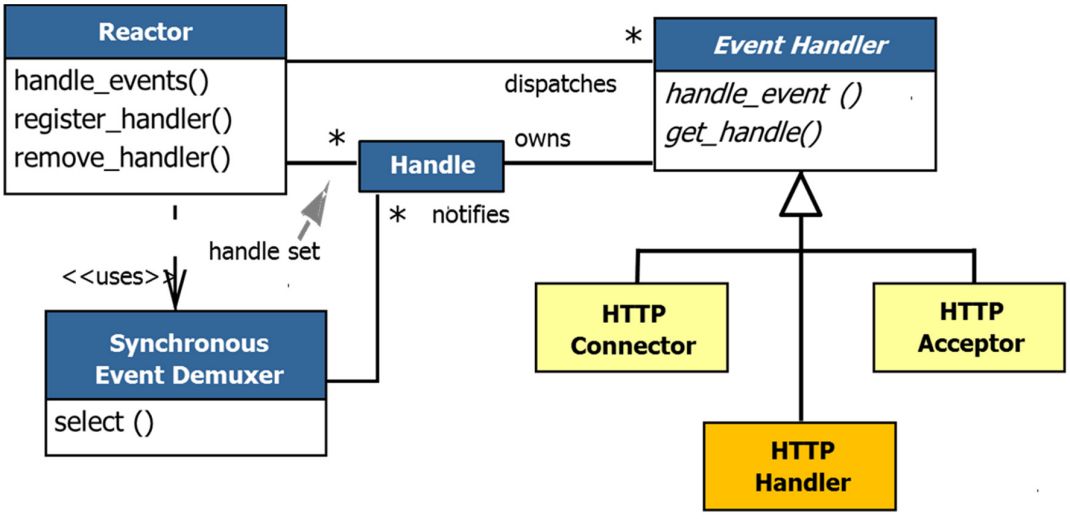
Acceptor-Connector

Separación conexión/inicialización vs. procesamiento



Acceptor-Connector

Separación conexión/inicialización vs. Procesamiento



e.g. JAWS web server

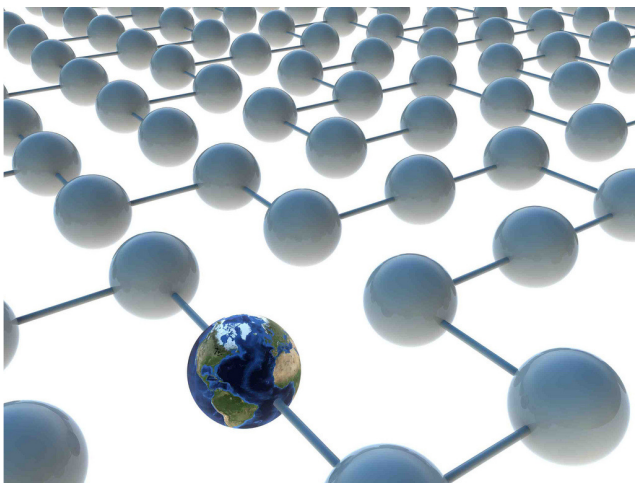
Bibliografía

Pattern-Oriented Software Architecture a.k.a. POSA

- Volume 1: A System of Patterns
- Volume 2: Patterns for Concurrent and Networked Objects
- Volume 3: Patterns for Resource Management
- Volume 4: A Pattern Language for Distributed Object Computing
- Volume 5: On Patterns and Pattern Languages

Curso recomendado

Pattern-Oriented Software Architectures for Concurrent and Networked Software
Douglas C. Schmidt (Vanderbilt University)
https://www.coursera.org/course/posasoftware



coursera