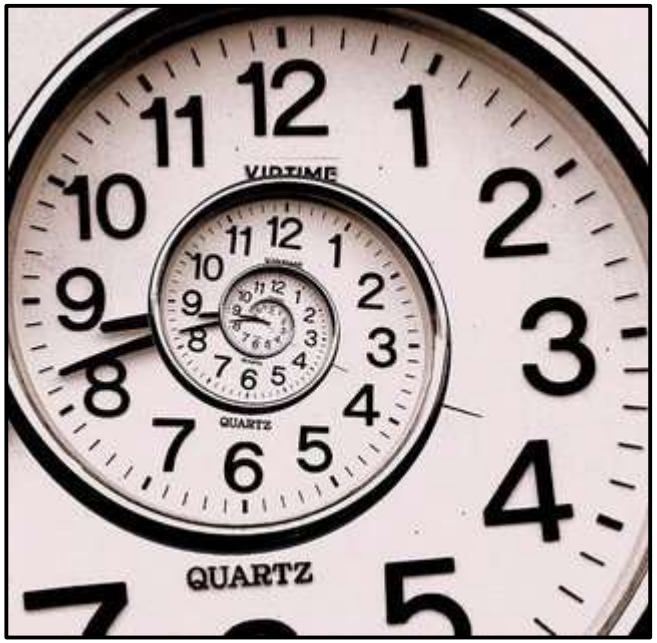
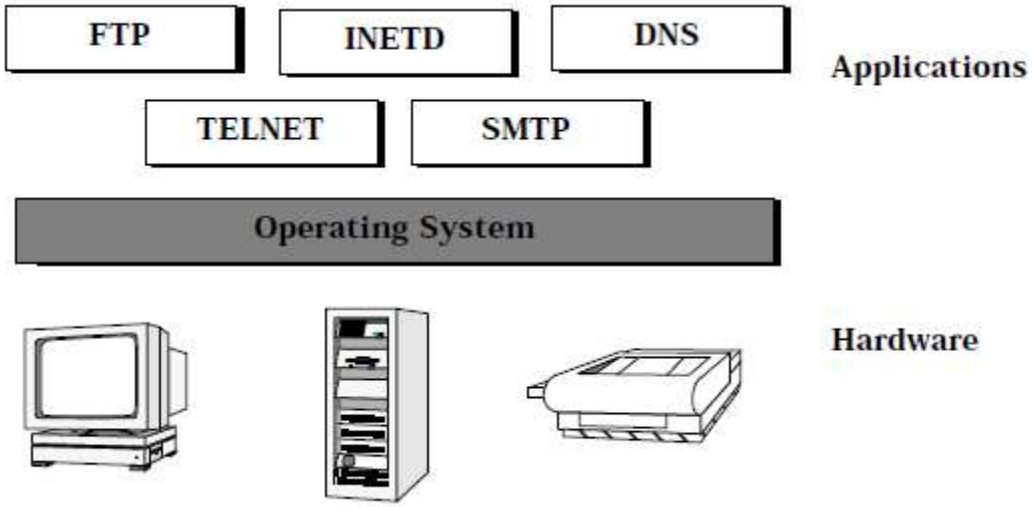
Half-Sync/Half-Async



http://homes.bio.psu.edu/people/faculty/bshapiro/spiral-clock.jpg

Half-Sync/Half-Async – architektonický vzor

- **Kontext**
  - vícevláknový systém s komunikací synchronních a asynchronních služeb
- **Účel**
  - zjednodušit použití takových služeb bez ztráty výkonnosti
- **Příklad**
  - síťování v BSD UNIXu

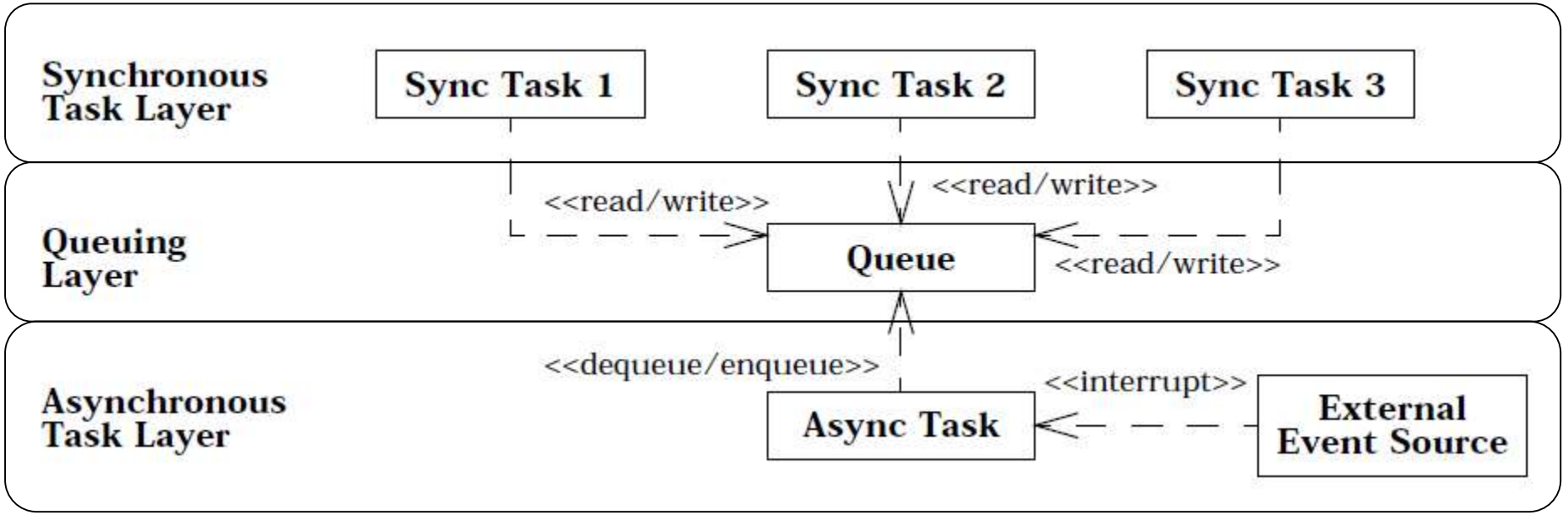


Half-Sync/Half-Async – problémy

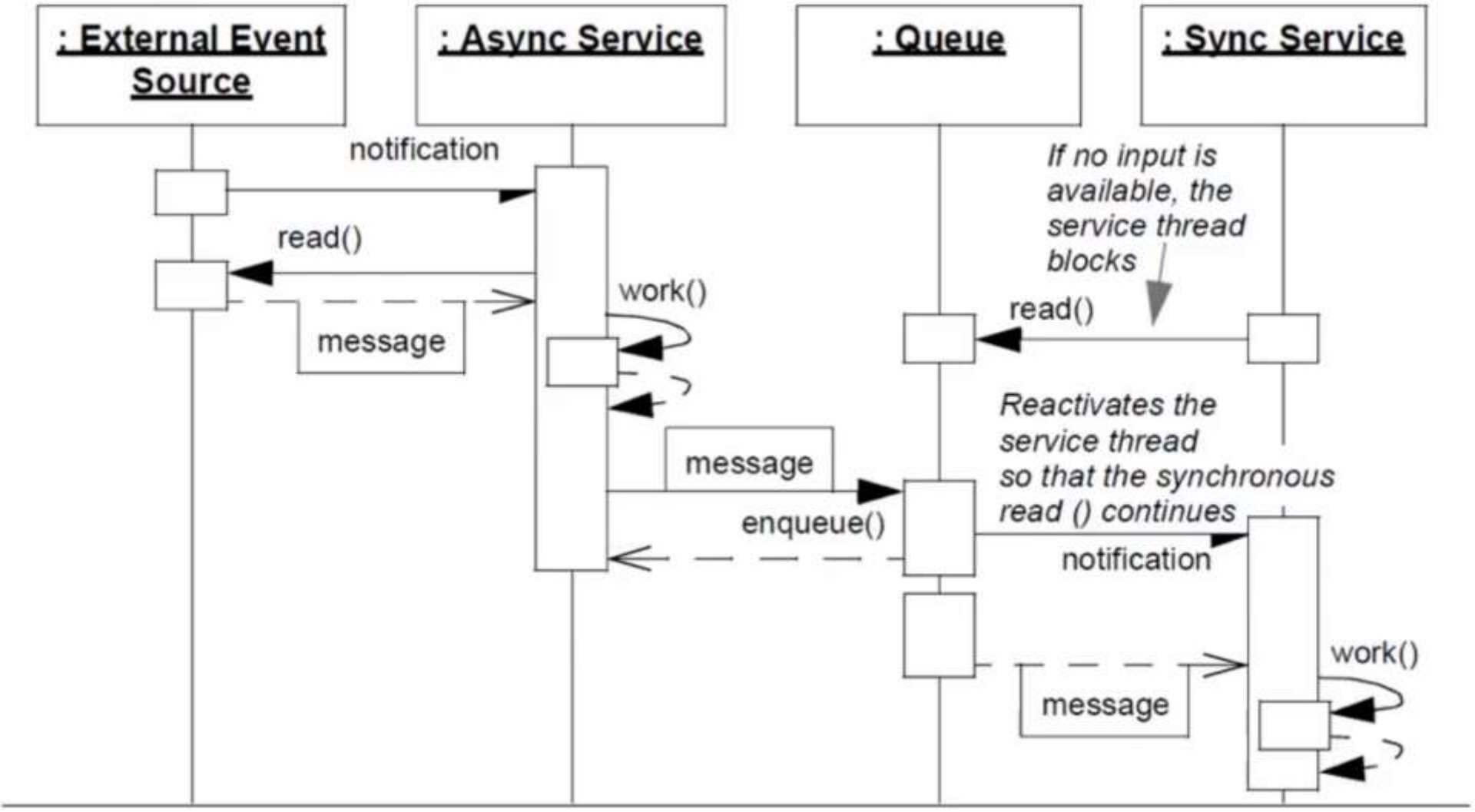
- **Problémy**
  - synchronní zpracování služeb
    - jednodušší programování, ale může dlouho blokovat
    - typicky high-level služby
  - asynchronní zpracování služeb
    - složitější programování, možnost vyššího výkonu
    - někdy je vynuceno přímo hardwarem
    - typicky low-level služby
  - tyto služby spolu potřebují komunikovat
  - jak to vše skloubit?

Half-Sync/Half-Async – struktura

- **Řešení**
  - rozdělit systém na synchronní a asynchronní vrstvu
  - mezi ně vložit komunikační mezivrstvu s frontou

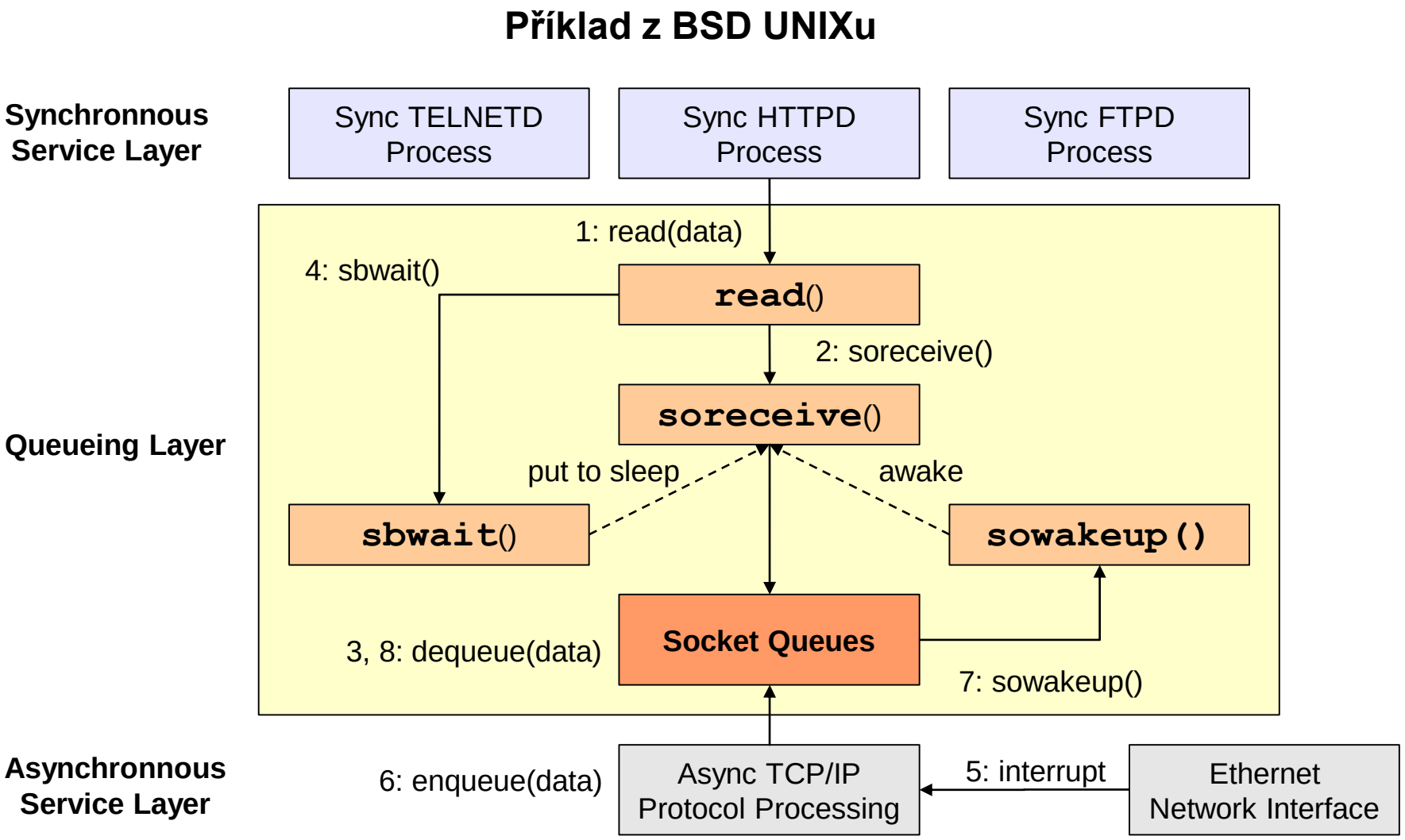


Half-Sync/Half-Async – dynamické chování



Overview of the Half-Sync/Half-Async Pattern, Douglas C. Schmidt, 2014, https://www.youtube.com/watch?v=dde0GAWBRnQ

Half-Sync/Half-Async – příklad

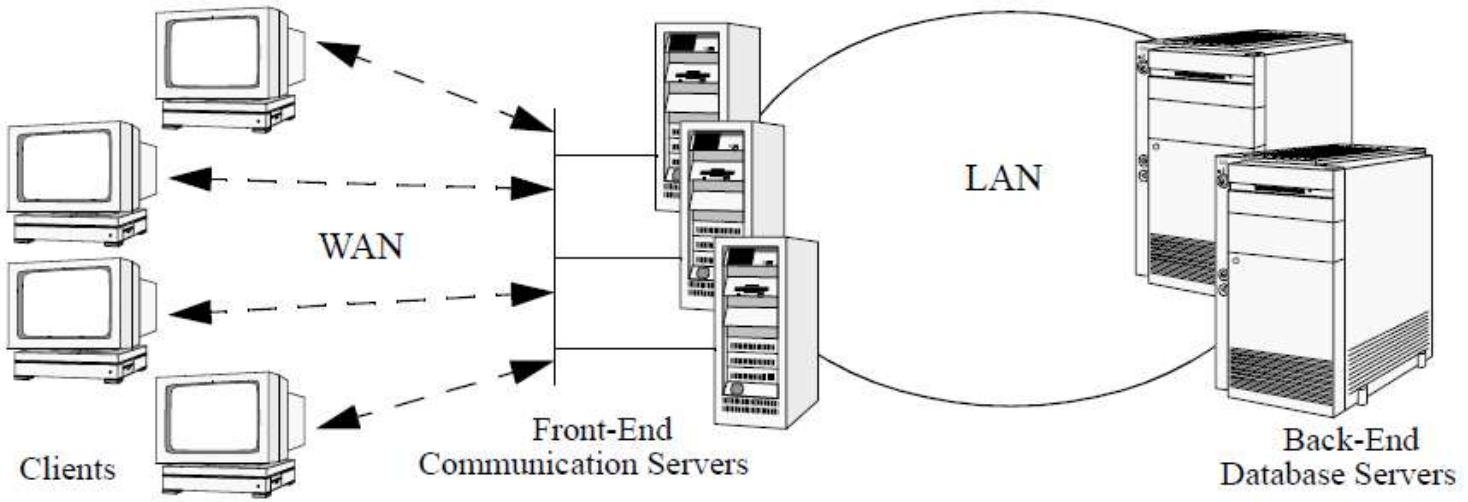


Half-Sync/Half-Async – varianty

- **Varianty**
  - **Asynchronní řízení, synchronní přístup k datům**
    - služby synchronní vrstvy mohou být notifikované asynchronně o vložení zprávy do fronty
    - [+] vyšší responzivita synchronních služeb
    - [-] zavádění problémů implementace async do sync vrstvy
    - příklad: UNIX I/O mechanismus - SIGIO signal (IPC)
  - **Half-Async/Half-Async**
    - asynchronní zpracování je přístupné i pro high-level služby
    - [-] zavádění problémů implementace async do sync vrstvy
    - když jde většina služeb obsluhovat async - lepší použít *Proactor*
    - příklad: POSIX realtime signal (asynchronní I/O)
  - **Half-Sync/Half-Sync**
    - synchronní zpracování i pro low-level služby
    - více vláken v jádře OS, služby mohou běžet autonomně, bez blokování jiných služeb
    - [+] zjednodušení implementace low-level služeb
    - [-] vyšší overhead synchronního zpracování
    - příklady: Mach (microkernel), Solaris (macrokernel)

Half-Sync/Half-Async – varianty

- **Varianty**
  - **Half-Sync/Half-Reactive**
    - v objektově orientovaných systémech
    - složení vzorů *Reactor* a *Active Object* s thread poolem
    - asynchronní vrstva – *Reactor*
    - mezivrstva – *Activation Queue*
    - synchronní vrstva – *Servant*
    - příklad: On-line Transaction Processing (OLTP)



Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Half-Sync/Half-Async – souvislosti, příklady

- **Souvislosti**
  - Half-Sync/Half-Reactive – *Reactor, Active Object, Servant*
  - vrstvení v Half-Sync/Half-Async je příkladem vzoru *Layers*
  - mezivrstva – *Mediator*
  - komunikace na mezivrstvě – *Pipes and Filters* (producer/consumer)
  - Serializace přístupu k objektu fronty z více vláken - *Monitor*

- **Praktické použití**
  - široké použití v operačních systémech (přerušení, správa aplikací)
  - moderní GUI frameworky
  - všude tam, kde je omezení určitých typů operací v určitých kontextech
    - blokující – neblokující
    - krátko trvající – dlouho trvající

- **Příklad ze života**
  - Hosteska v restauraci – vítání a usazování hostů

Half-Sync/Half-Async – shrnutí

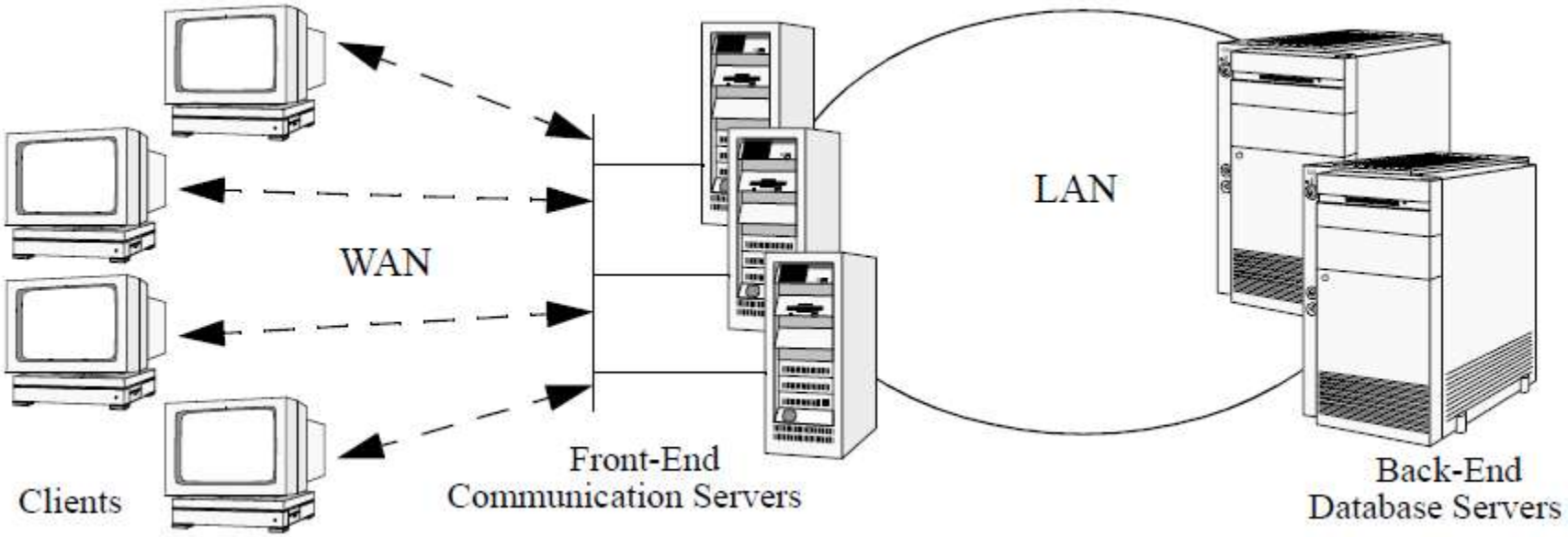
- **Princip**
  - oddělení synchronních a asynchronních služeb do dvou vrstev
- **Výhody**
  - jednodušší programování synchronních služeb při zachování výkonnosti
  - uzavření složitosti asynchronních služeb do malé části systému
  - oddělení synchronizačních politik
  - centralizovaná komunikace mezi vrstvami
- **Nevýhody**
  - režie za komunikaci mezi vrstvami
    - dynamická (re)alokace paměti (synchronizace, fragmentace)
    - synchronizace procesů (zámky, semafory)
    - změna kontextu CPU
    - update cache u vícejádrových procesorů
  - složitější debugování a testování

Leader/Followers

Tony Newlin, http://www.tonynewlin.com/Follow_the_Leader.html

Leader/Followers – architektonický vzor

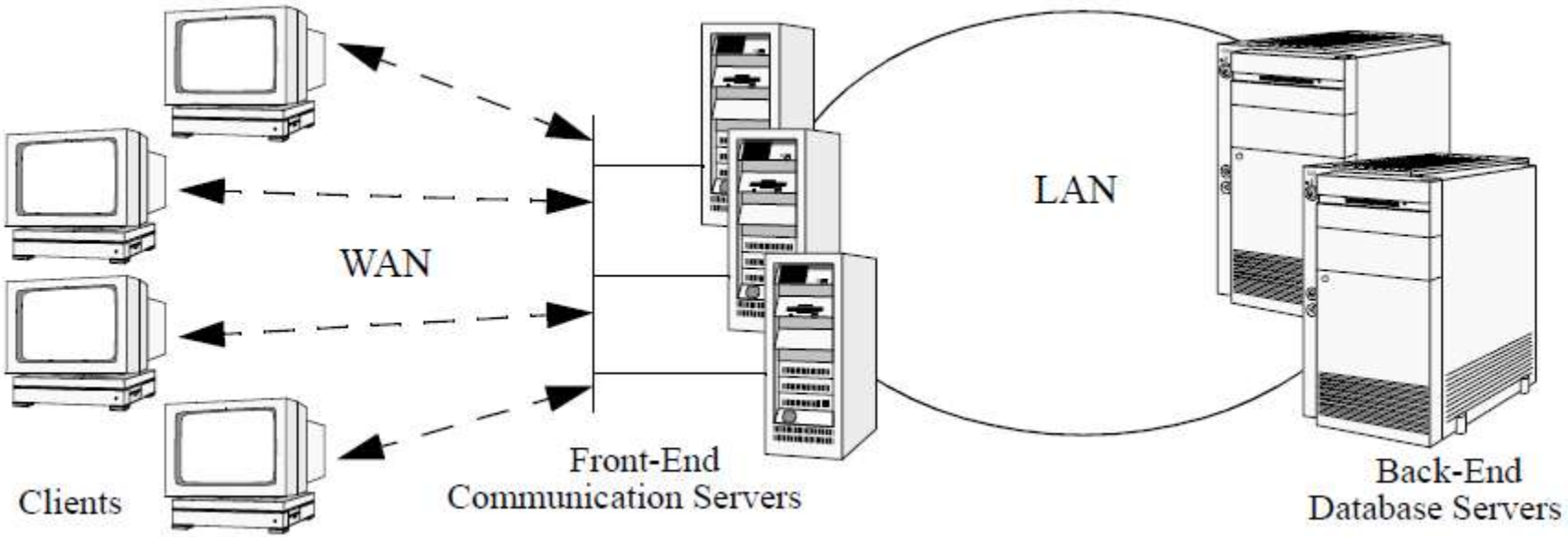
- **Kontext**
  - vlákna musí efektivně zpracovávat události ze sdíleného zdroje
- **Účel**
  - více vláken se střídá v přijímání, demultiplexování a zpracování požadavků, které přicházejí z více zdrojů
- **Příklad**
  - On-line Transaction Processing (OLTP)



Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Leader/Followers – problémy a řešení

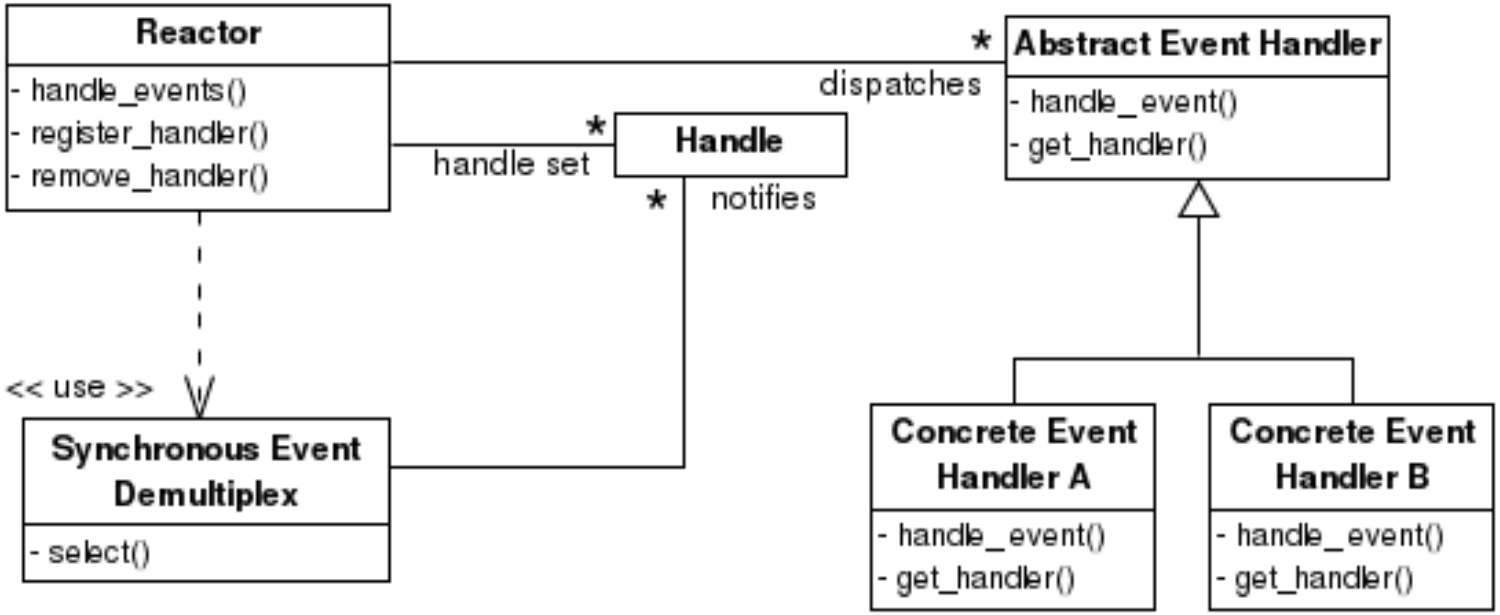
- **Problémy**
  - chceme efektivní demultiplexování událostí
  - nutné omezit režii – přepínání kontextů, synchronizace, cache, alokace
  - koordinace vláken při demultiplexování – ochrana před race conditions



Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Leader/Followers – problémy a řešení

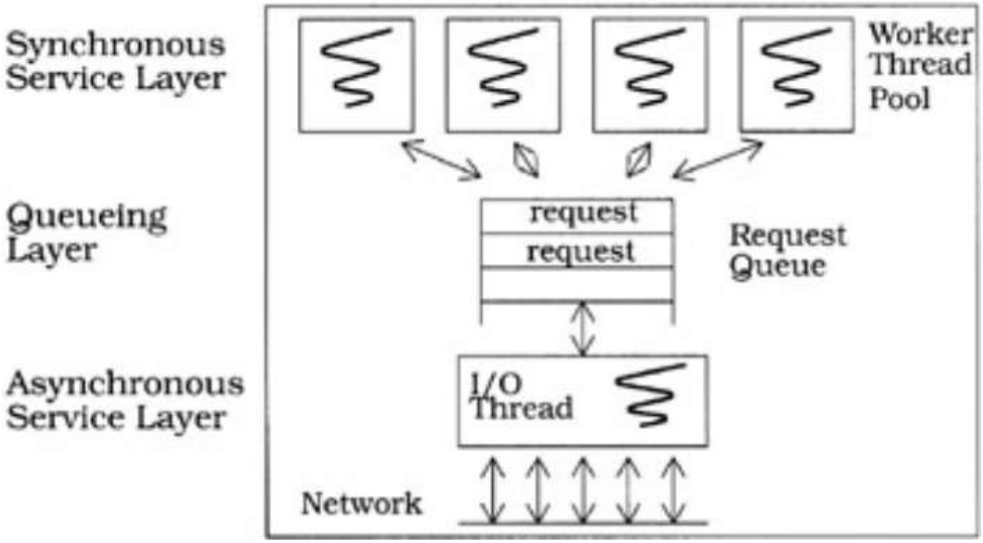
- **Pokus 1: Reactor**
  - jednovláknové zpracování událostí
- **Nevýhody**
  - serializuje zpracování událostí
  - výkon degradují dlouho běžící a blokující dotazy
  - nebenefituje transparentně z vícevláknových platforem



Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Leader/Followers – problémy a řešení

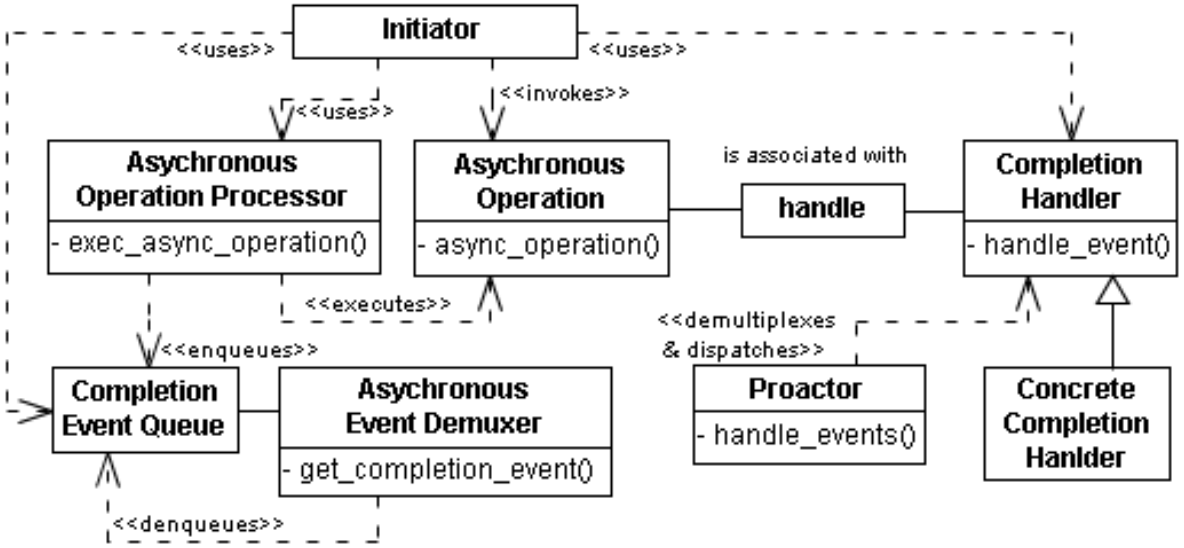
- **Pokus 2: Half-Sync/Half-Reactive**
  - vícevláknové zpracování událostí
  - vyhrazené síťové I/O vlákno
    - čekající na události od klientů (socketů)
  - synchronní Thread Pool
    - pracovní vlakna, obsluhující požadavky na frontě
  - synchronizovaná fronta zpráv
    - používá vzor *Monitor*
- **Nevýhody**
  - vyšší režie
    - alokace paměti
    - synchronizace
    - změna kontextu CPU
    - cache coherence
  - latence
    - zbytečně vysoká
“v nejlepším případě”



Leader/Followers, Douglas C. Schmidt, Carlos O’Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

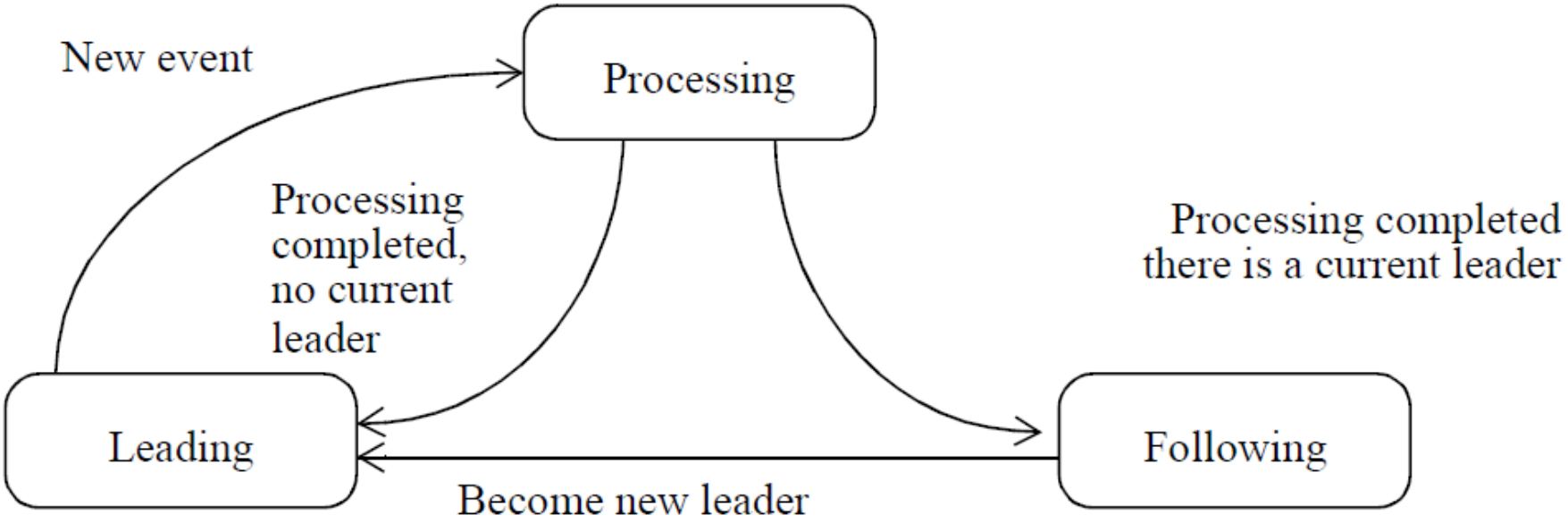
Leader/Followers – problémy a řešení

- **Pokus 3: plně asynchronní Thread Pool**
  - návrhový vzor *Proactor*
- **Výhody**
  - eliminuje potřebu vyhrazeného síťového vlákna
    - redukuje režii (alokace, synchronizace, změna kontextu)
- **Nevýhody**
  - vyžaduje efektivní implementaci v operačním systému (OS)
    - mnoho OS nepodporuje, resp. podporuje neefektivně
    - zvyšuje složitost implementace



Leader/Followers – problémy a řešení

- **Řešení Leader/Followers**
  - události demultiplexuje více vláken
  - tato vlákna se střídají v demultiplexování událostí
  - přijatá událost je synchronně předána příslušné službě ke zpracování

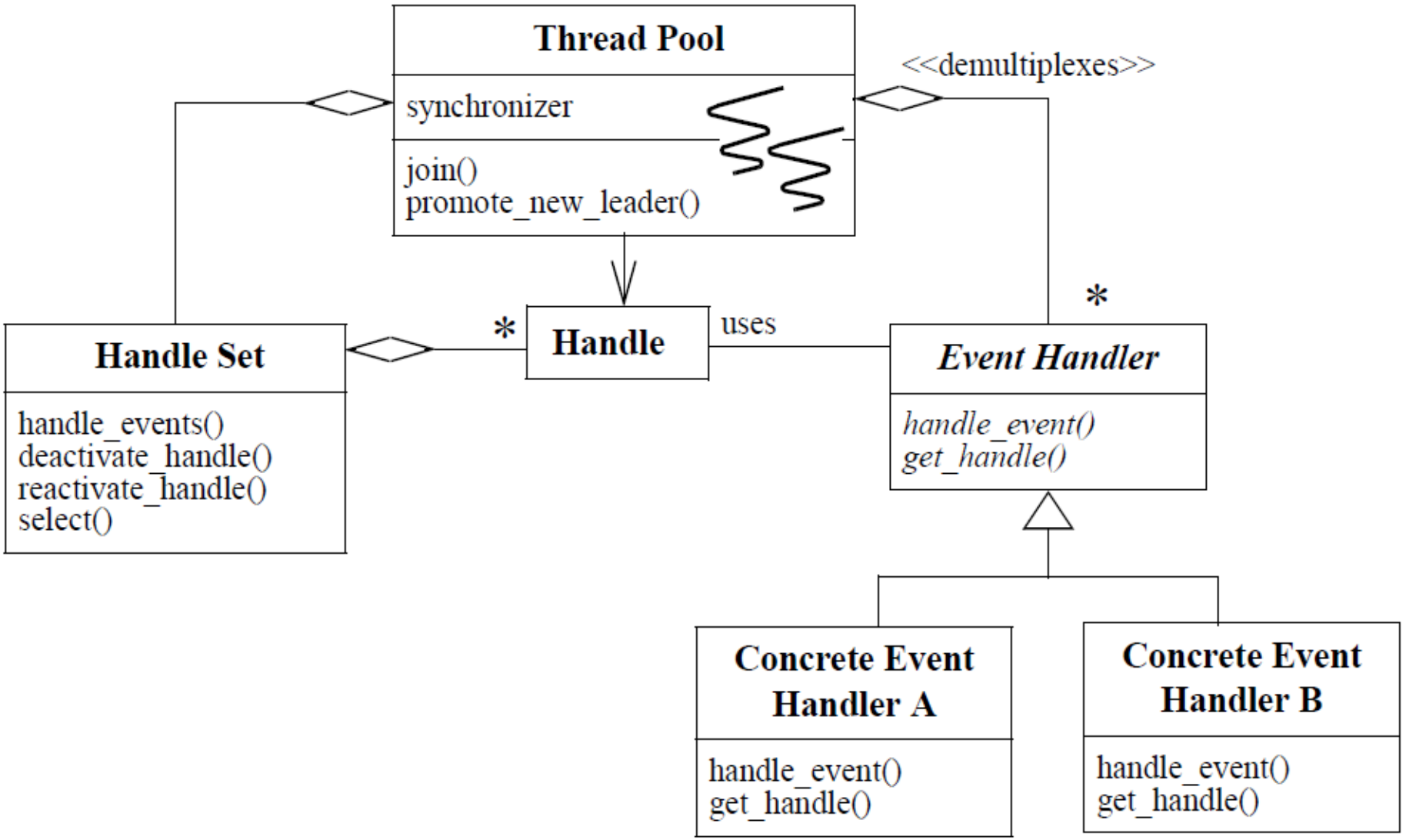


Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Leader/Followers – životní cyklus vlákna

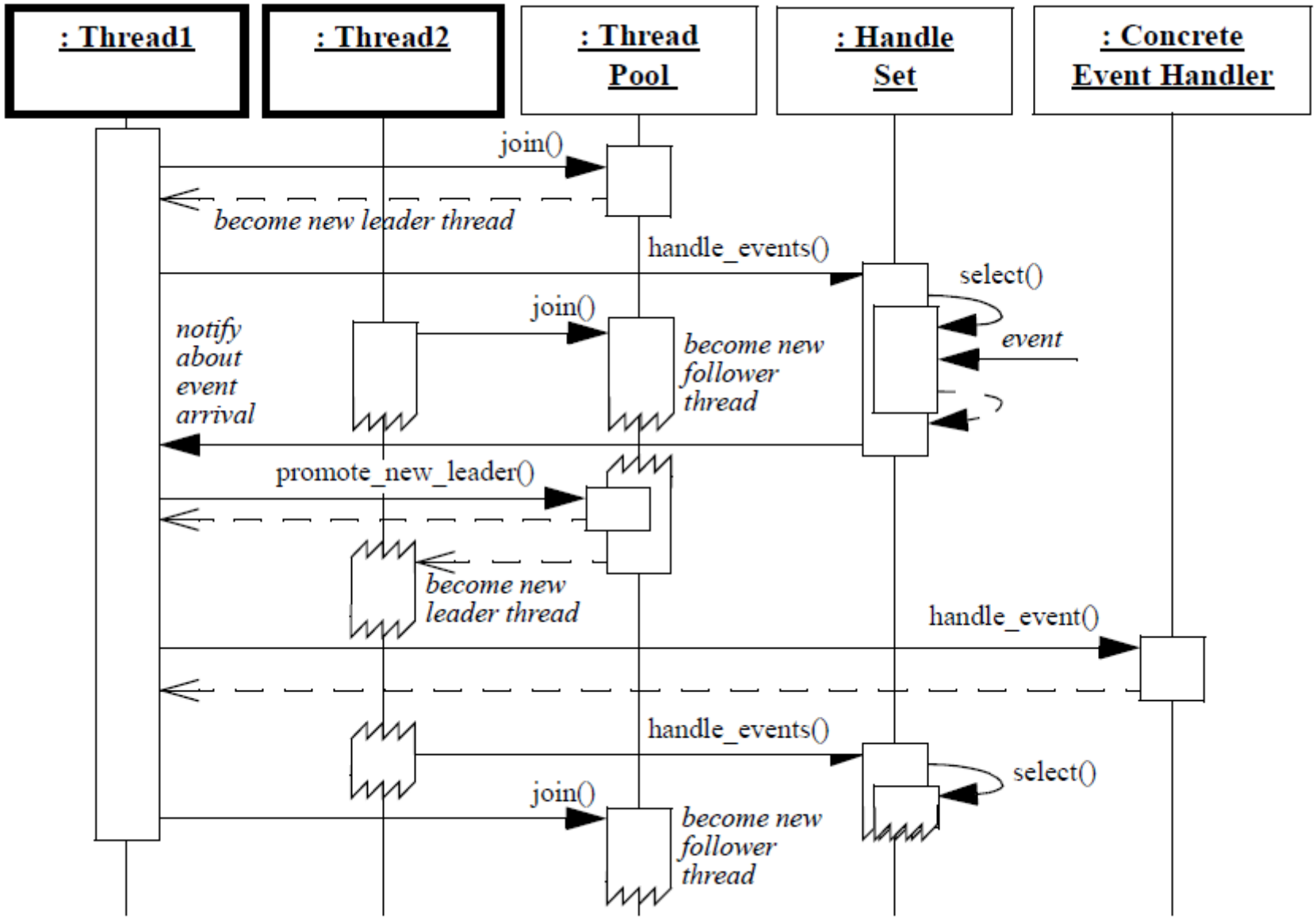
- **Leading**
  - **leader** čeká na událost ze zdroje událostí
  - když zaznamená novou událost:
    - povýší nějakého **followera** na **leadera**
    - změní se na **processing** a zpracuje událost
- **Following**
  - **followers** čekají na to, aby se stali **leadrem**
  - různé způsoby organizace čekajících **followers**
    - zasobnik LIFO pomáhá při “ohřívání” CPU cache
- **Processing**
  - demultiplexuje a dispatchuje událost do event handleru
  - více **processing** vláken zpracovává události současně zatím co **leader** čeká
  - po zpracování události se vrátí do roly **followera**
  - jestli je thread pool prázdný, tak se rovnou postaví do roly **leadera**
  - Alternativa:
    - Vlákno, které právě dopracovalo, se taky může hned nastavit jako novy **leader** - pomáhá to při “ohřívání” CPU cache

Leader/Followers – struktura



Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Leader/Followers – dynamické chování



Leader/Followers, Douglas C. Schmidt, Carlos O'Ryan, Michael Kircher, Irfan Pyarali, and Frank Buschmann, 2000

Leader/Followers – varianty

- **Varianty**
  - **Bound Handle/Thread Associations**
    - k vláknům jsou přiřazeny (bound) handles
    - obyčejný Leader/Followers přiřazení nemá (unbound)
    - jestliže leaderovi událost nepatří, předá ji dál správnému followerovi
  - **Multiple Handle Sets**
  - **Multiple Leaders and Multiple Followers**
  - **Hybrid Thread Associations** – vlákna bound i unbound
  - **Hybrid Client/Servers** – přiřazení vláken se muže měnit
  - **Alternative Event Sources and Sinks**
    - vice typů/zdrojů událostí
    - jeden leader přiřazen ke každému zdroji
    - nevýhoda – je potřeba vždy více vláken nez zdrojů (škálovatelnost)
- **Praktické použití**
  - webové servery, transakční monitory
  - CORBA Object request brokery
- **Příklad ze života**
  - stanoviště taxíků

Leader/Followers – shrnutí

- **Výhody**
  - efektivita, vyšší výkon
  - jednoduchost programování
- **Nevýhody**
  - složitější implementace
  - menší flexibilita
  - potenciální bottleneck – pouze jedno vlákno pro I/O

Srovnání s Half-Sync/Half-Reactive

- **Vhodné použití**
  - když potřebujeme nízkou (realtime) odezvu
  - máme mnoho událostí s krátkou dobou běhu
  - “predictability over scalability”

- **Alternativy**
  - **Half-Sync/Half-Async** – **“**scalability over predictability”
  - **Active Object**
  - **Reactor** – když je zpracování událostí krátké
  - **Proactor** – pokud nám nevadí asynchronní zpracování a OS to umí