






ASIO C++

Grundlagen der Netzwerkprogrammierung

Maximilian Florkowski
Geschäftsbereich Logistik
maximilian.florkowski@inform-software.de

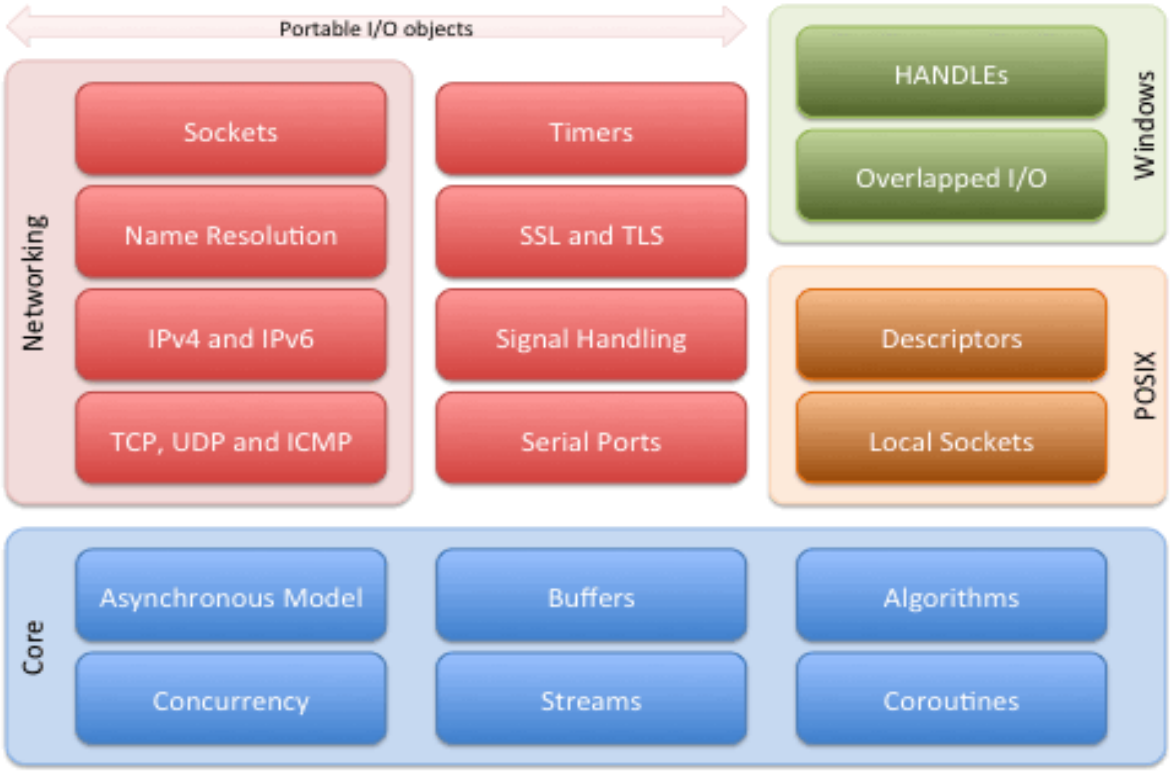
INFORM

Inhaltsverzeichnis

SyncroTESS
Efficient Logistics

INFORM

Asynchronous Input/Output



http://think-async.com/

Vorteile der Library

SyncroTESS
Efficient Logistics

- Portierbarkeit
- Skalierbarkeit
- Effizienz
- Moderne C++ Konzepte
- Einfache Nutzung
- Stark Modifizierbar

INFORM

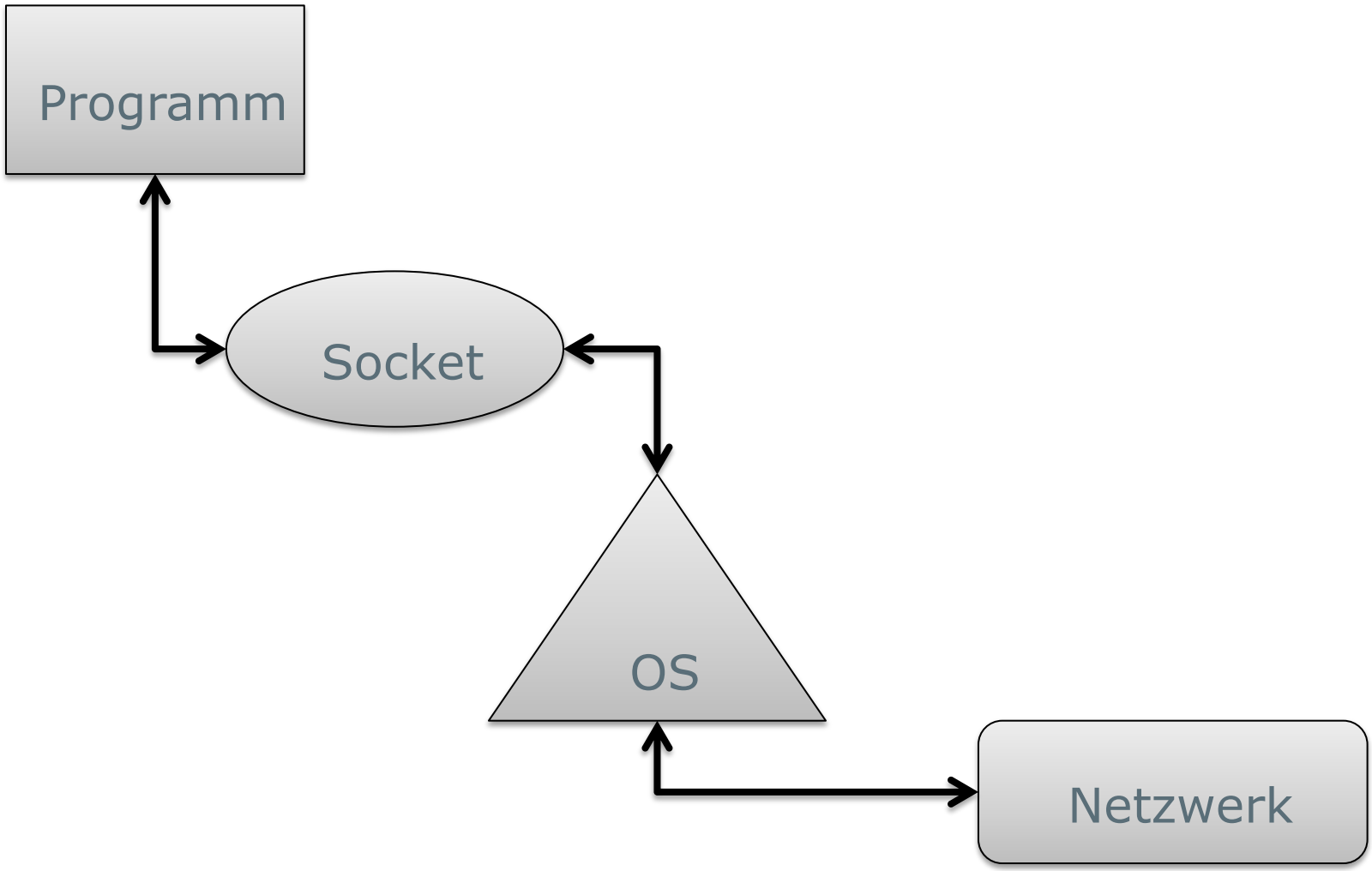
ISO/OSI-Referenzmodell

OSI-Schicht	TCP/IP-Schicht	Beispiel
Anwendungen (7)	Anwendungen	HTTP, UDS, FTP, SMTP, POP, Telnet, OPC UA
Darstellung (6)		
Sitzung (5)		
		SOCKS
Transport (4)	Transport	TCP, UDP, SCTP
Vermittlung (3)	Internet	IP (IPv4, IPv6), ICMP (über IP)
Sicherung (2)	Netzzugang	Ethernet, Token Bus, Token Ring, FDDI, IPoAC
Bitübertragung (1)		

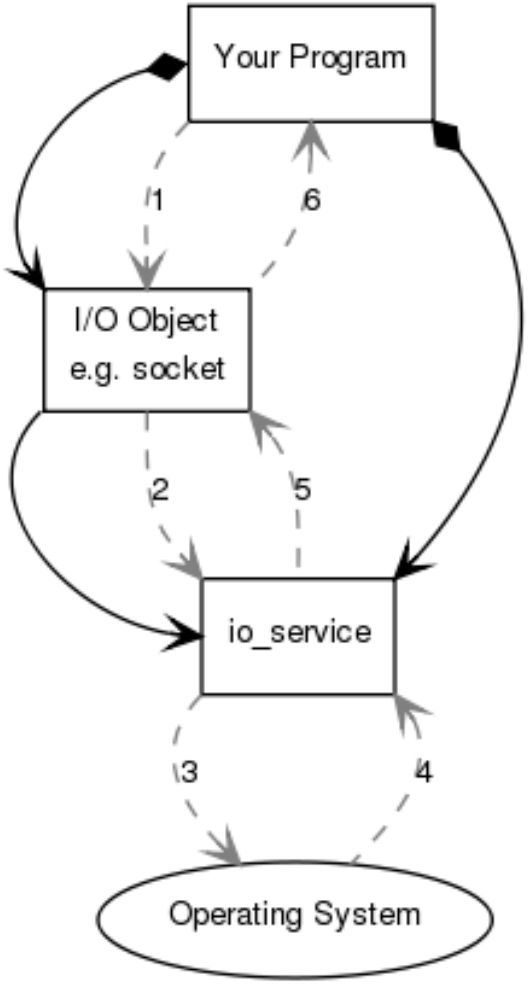
http://de.wikipedia.org/wiki/Internetprotokollfamilie

Transmission Control Protocol	User Datagram Protocol
Verbindungsorientiert	Verbindungslos
Zuverlässig	Nicht zuverlässig
Geordnet	Ungeordnet
Aufwendig	Simpel
Langsam	Schnell
Stream Sockets	Datagram Sockets

Sockets

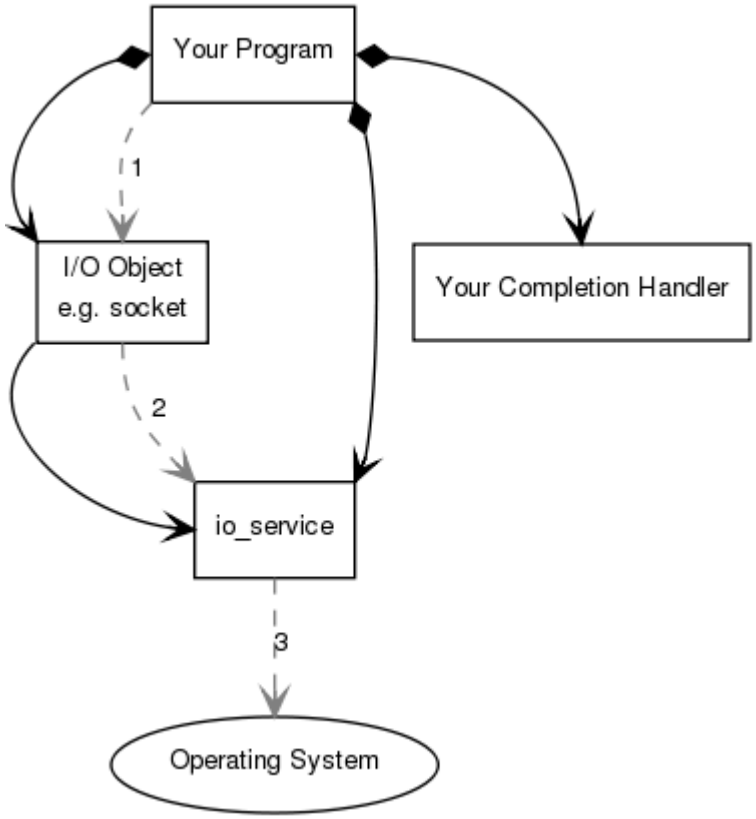


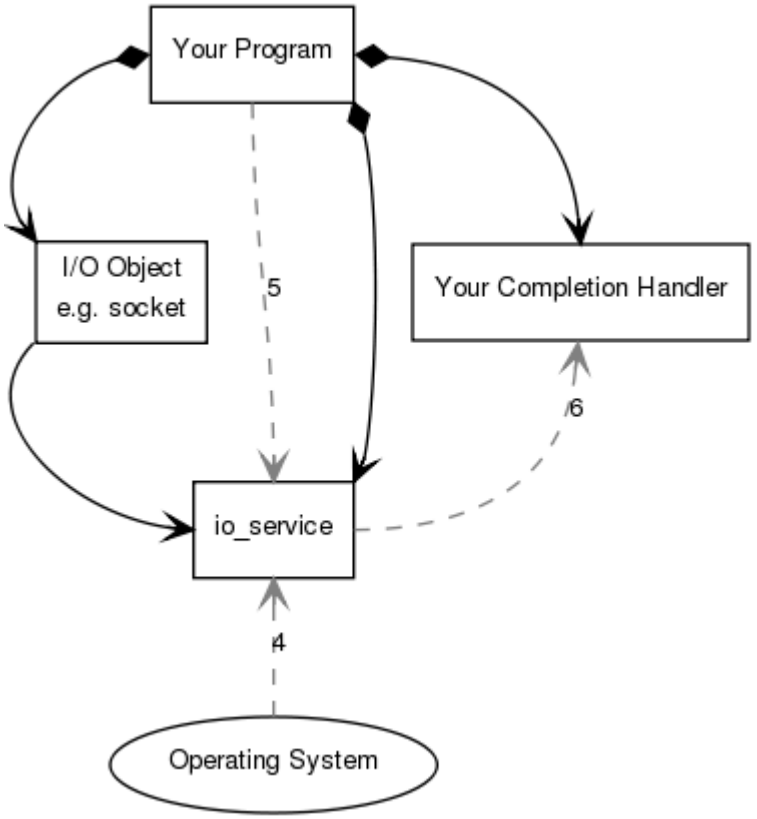
Synchrone Netzwerkkommunikation



http://think-async.com/Asio/asio-1.10.2/doc/asio/overview/core/basics.html

Think Async - Asynchrone Netzwerkkommunikation



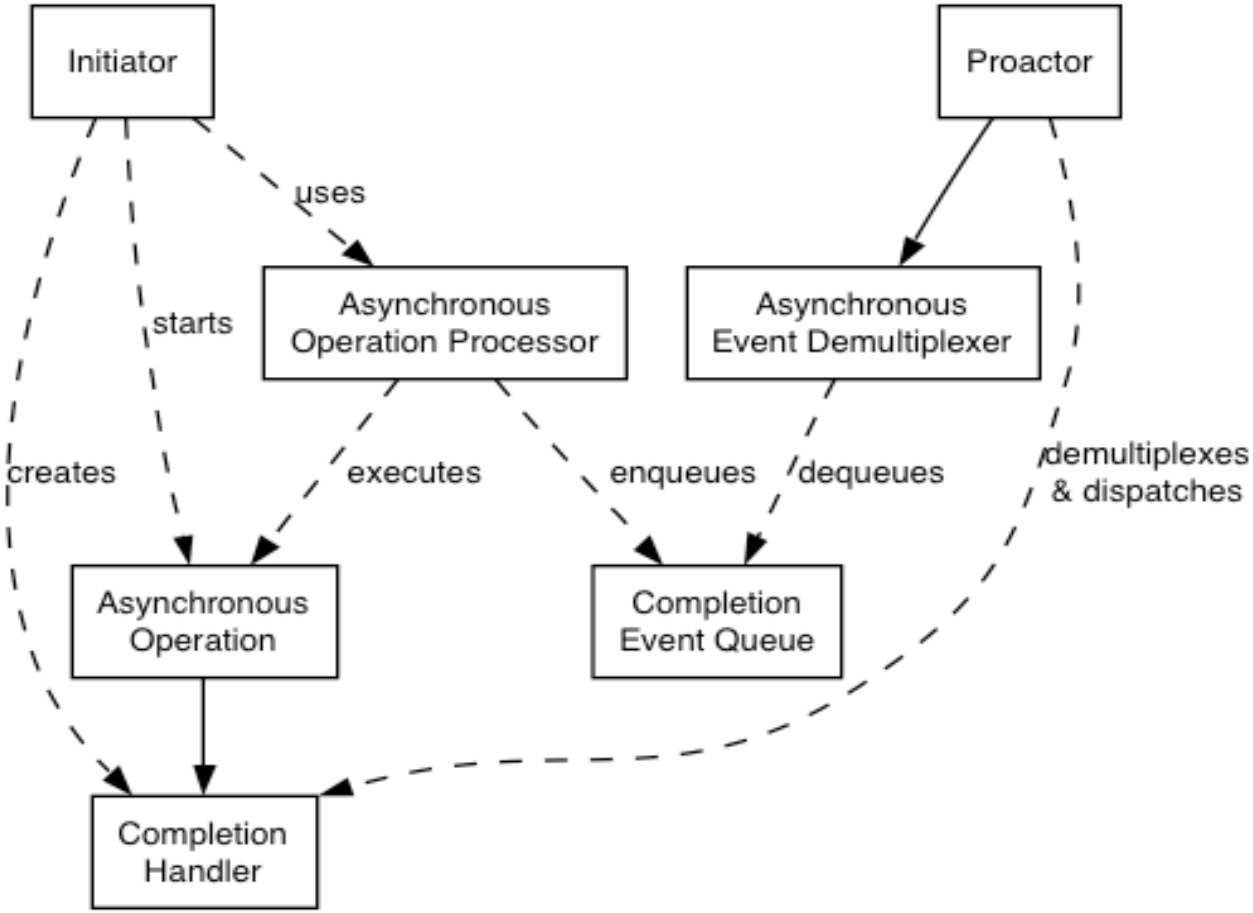


http://think-async.com/Asio/asio-1.10.2/doc/asio/overview/core/basics.html

SyncroTESS
Efficient Logistics

- Vorteile:
  - Performance
  - Weniger Threads
  - Skalierbarkeit
  - Aufteilbarkeit
- Nachteile:
  - Komplex
  - „Unnatürlicher Programmfluss“

The Proactor Design Pattern



http://www.cs.wustl.edu/~schmidt/PDF/proactor.pdf

http://think-async.com/Asio/asio-1.10.2/doc/asio/overview/core/async.html

Codebeispiele

https://github.com/mflorkow/cpp_adv_asio.git

SyncroTESS
Efficient Logistics

- Downloaden Sie das Projekt von Github.
- Führen sie den make Befehl für alle drei Programmteile aus.
- Starten Sie den Server auf beliebigen Ports.
- Versuchen Sie auf einen Server Ihrer Kommilitonen mit einem der Clients zu connecten.

Aufgaben – 2

SyncroTESS
Efficient Logistics

- Verbessern Sie das Programm, sodass im Chat Namen und nicht IP/Port-Kombinationen angezeigt werden.
- Implementieren Sie eine Verschlüsselung mithilfe von Handlern.
- Implementieren Sie die Möglichkeit, einen privaten Chat zu führen.
- Implementieren Sie einen Timeout für UDP Verbindungen.
- Führen Sie eine Rechtesystem ein, damit Admins normale User kicken/bannen/whatever können.
- Teilen sie den Server in zwei Programme auf, die für jeweils ihr Protokoll zuständig sind, aber ALLE Clients unabhängig von ihrem Typ jede Nachricht erhalten.
- Ihre Ideen.

SyncroTESS
Efficient Logistics

- Was muss man ändern, wenn man anstatt eines Strings eine Datei senden wollte?
- Welches Protokoll wird für die meisten Anwendungen genutzt?
- Braucht TCP mehr Ressourcen als UDP?
- Welcher ist der größte Vorteil zwischen Asynchroner und Synchrone Programmierung?
- Warum muss der IO-Service in mindestens einen Thread gestartet werden?
- Was ist eine häufige Fehlermeldung die beim Abbruch einer TCP-Basierten Verbindung auftritt?

- Wie kann ich Verhindern, dass ich ewig auf eine Antwort warte?
- Sollte man bei asynchronen Operationen Locks nutzen?
- Kann eine Anwendung gleichzeitig auf einem Socket lesen und schreiben?
- Welche Vorteile bietet die Nutzung von Handlern?
- Kann ein Client auf mehreren „Kanälen“ mit einem Server kommunizieren?
- Welchen der Clients würden Sie einem neuen Nutzer empfehlen und warum?

Links

SyncroTESS
Efficient Logistics

- http://think-async.com/
  http://www.boost.org/doc/libs/1_55_0/doc/html/boost_asio.html
- http://www.highscore.de/cpp/boost/asio.html
- http://www.kegel.com/c10k.html
- http://stackoverflow.com/questions/tagged/boost-asio?sort=votes
- http://www.cs.wustl.edu/~schmidt/PDF/proactor.pdf
- https://github.com/mflorkow/cpp_adv_asio.git