Evaluierung der QoS-Unterstützung in TAO/ACE

Verteidigung Großer Beleg

Ansgar W. Konermann

14. Januar 2003

Gliederung

- Aufgabenstellung
- Idealisierte QoS-Architektur
- QoS in ACE und TAO
- Unterstützungsgrad für QoS in TAO
- Beispielprogramm zur QoS-Durchsetzung
- Zusammenfassung, Ausblick

Aufgabenstellung

- Betrachtung von ACE und TAO
- Schwerpunkt: *Vertikale* QoS-Durchsetzung
- Vorarbeit für COMQUAD-Architektur
- Welche QoS-Merkmale gibt es?
- Wie werden diese Merkmale spezifiziert?
  - Welche Abstraktionsebenen existieren?
- Abbildungsmechanismen
- Durchsetzungsmechanismen

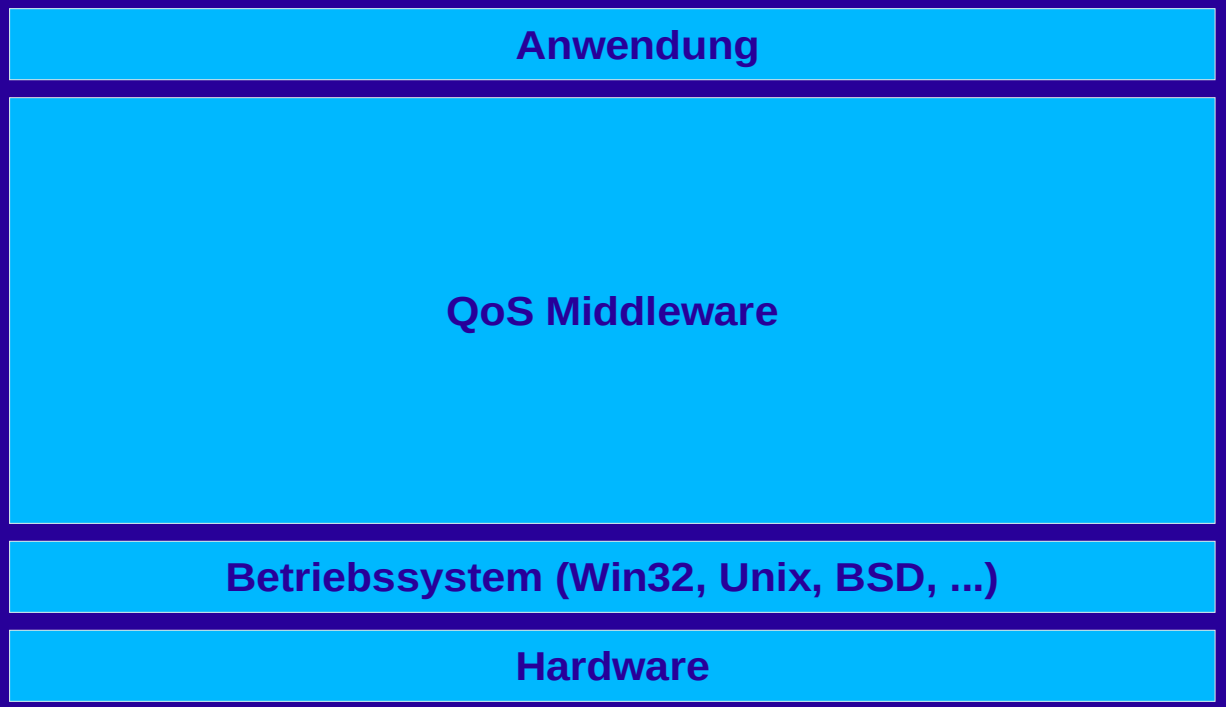
Was ist ACE, was ist TAO?

- ACE besteht aus:
  - Betriebssystem-Anpassungsschicht
  - Klassenbibliothek
  - Framework-Sammlung
  - Dienste für verteilte Anwendungen
  - Abstrahierte QoS API für Netzwerk-QoS (AQoSA)
- TAO
  - baut auf ACE auf (Implementierungsgrundlage)
  - CORBA-konforme Verteilungsmiddleware
  - QoS- und RT-Unterstützung

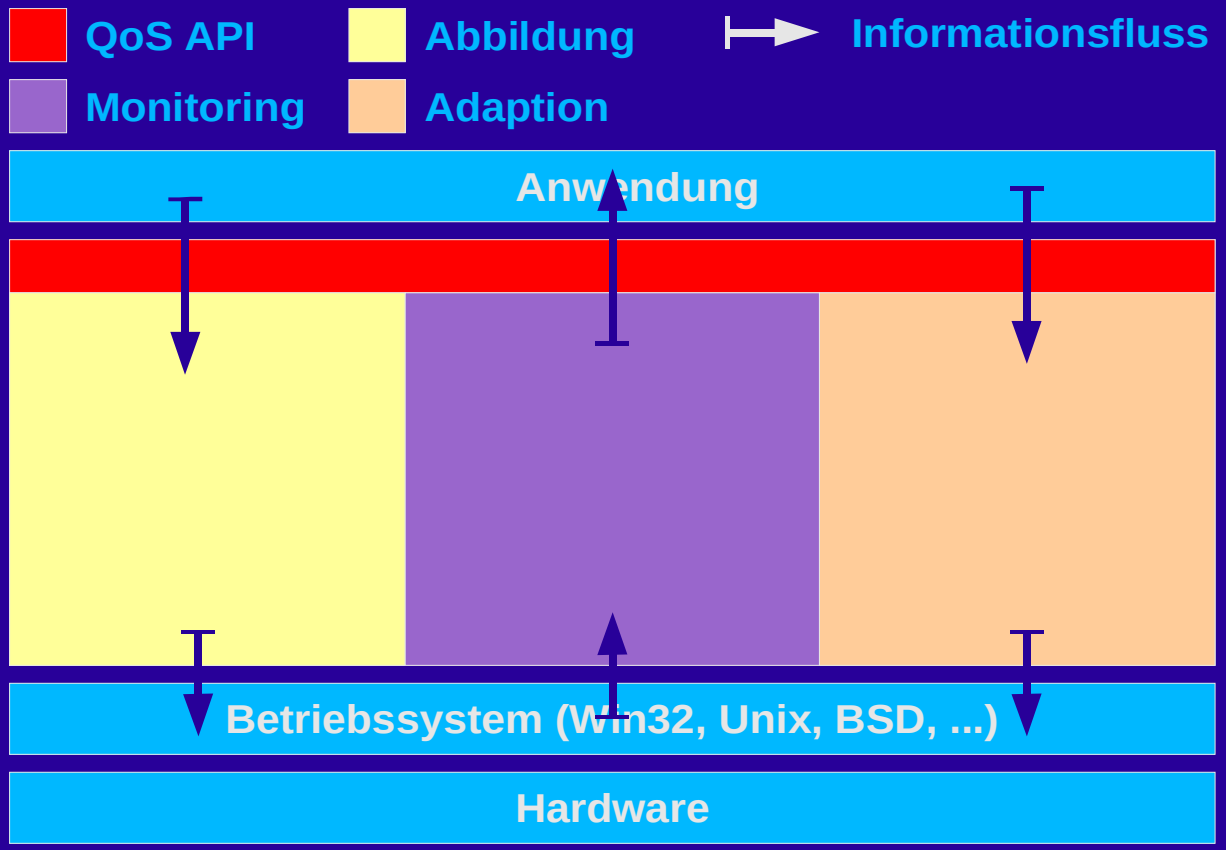
QoS in ACE und TAO

- Vorstellung einer idealisierten QoS-Architektur
- Systemarchitektur ACE, TAO und Dienste
  - Schrittweise Vorstellung
  - Abhängigkeiten
- Abbildung: idealisierte auf reale Architektur
- Vergleich und Bewertung

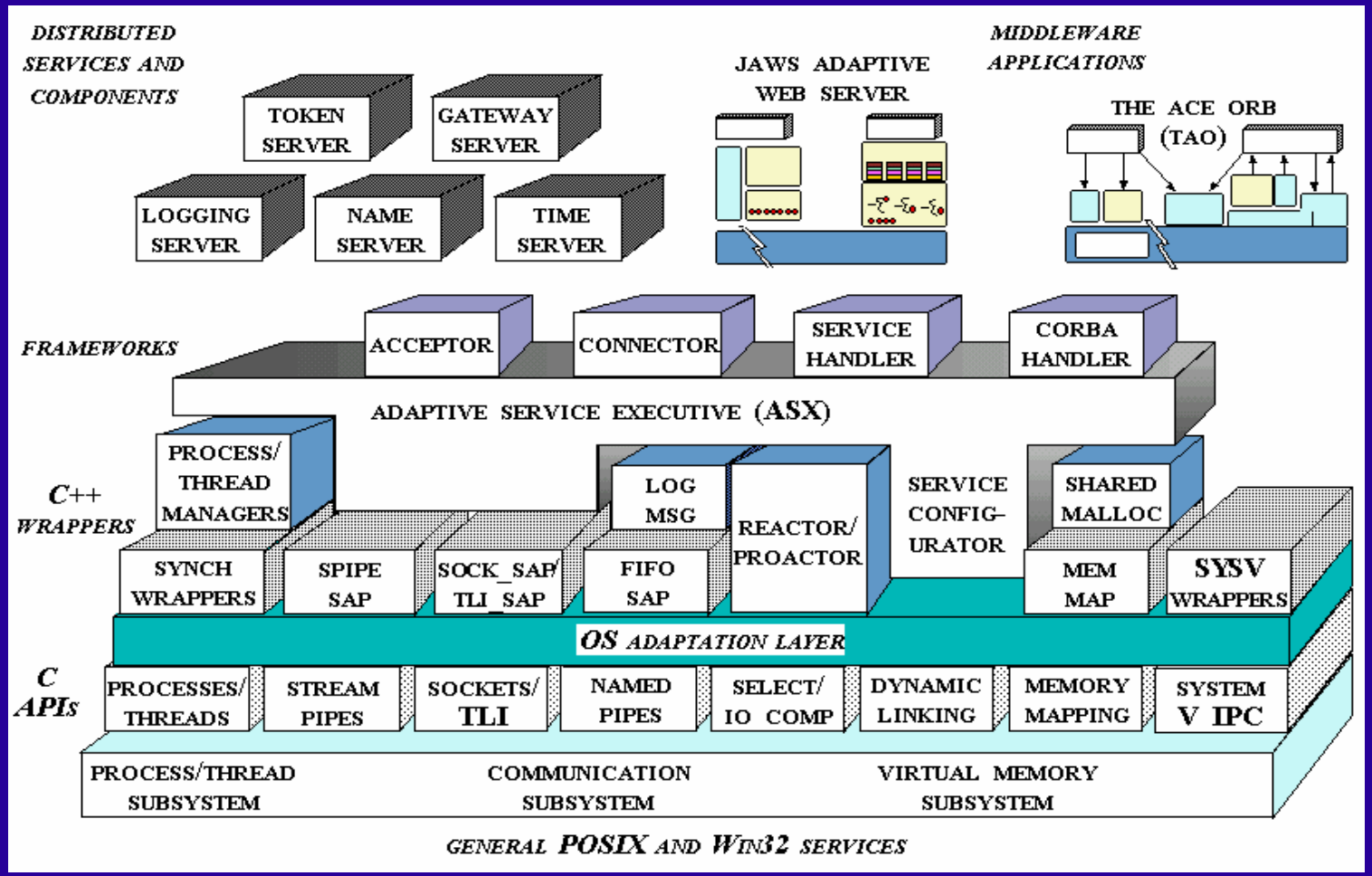
Idealisierte QoS-Architektur



Idealisierte QoS-Architektur



Grobarchitektur ACE



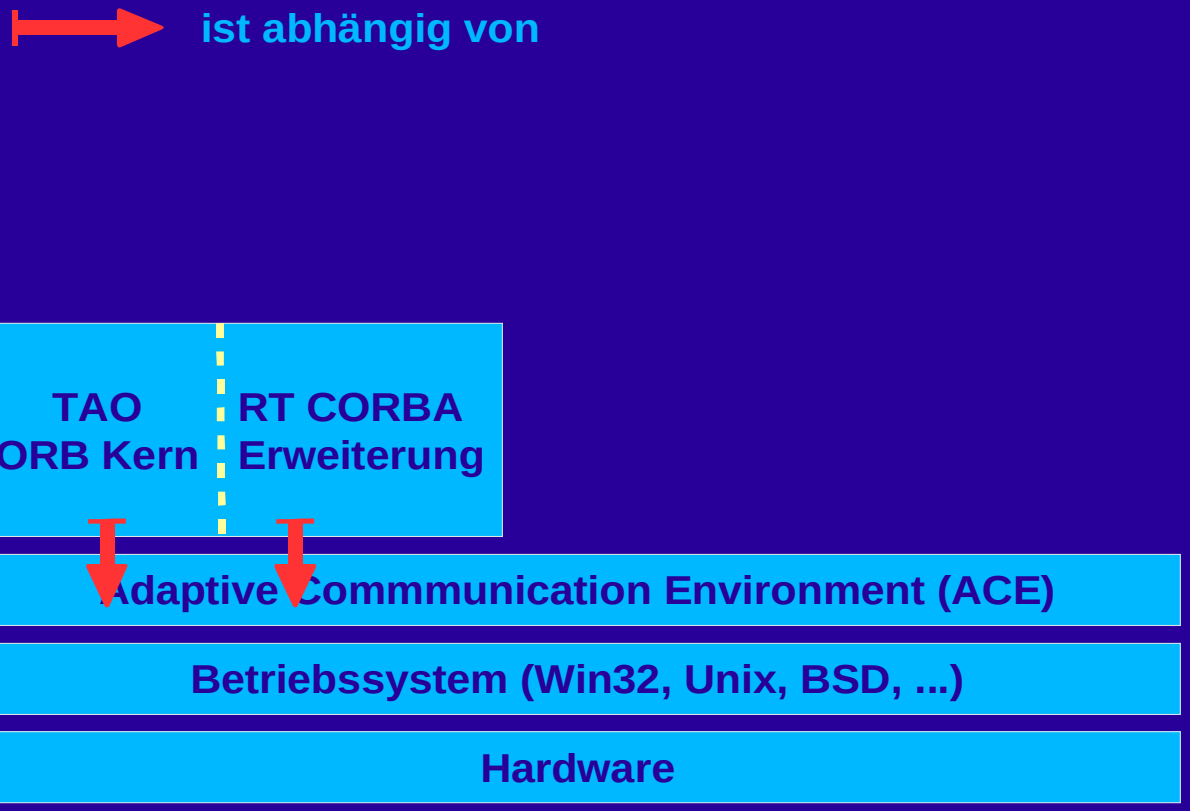
Eingrenzung: ACE

- Hier: Gleichsetzung ACE :=
  - OS Adaption Layer *(C APIs)*
  - C++ Wrapper Facades
  - Frameworks
- ACE als *Schicht* innerhalb der Systemarchitektur

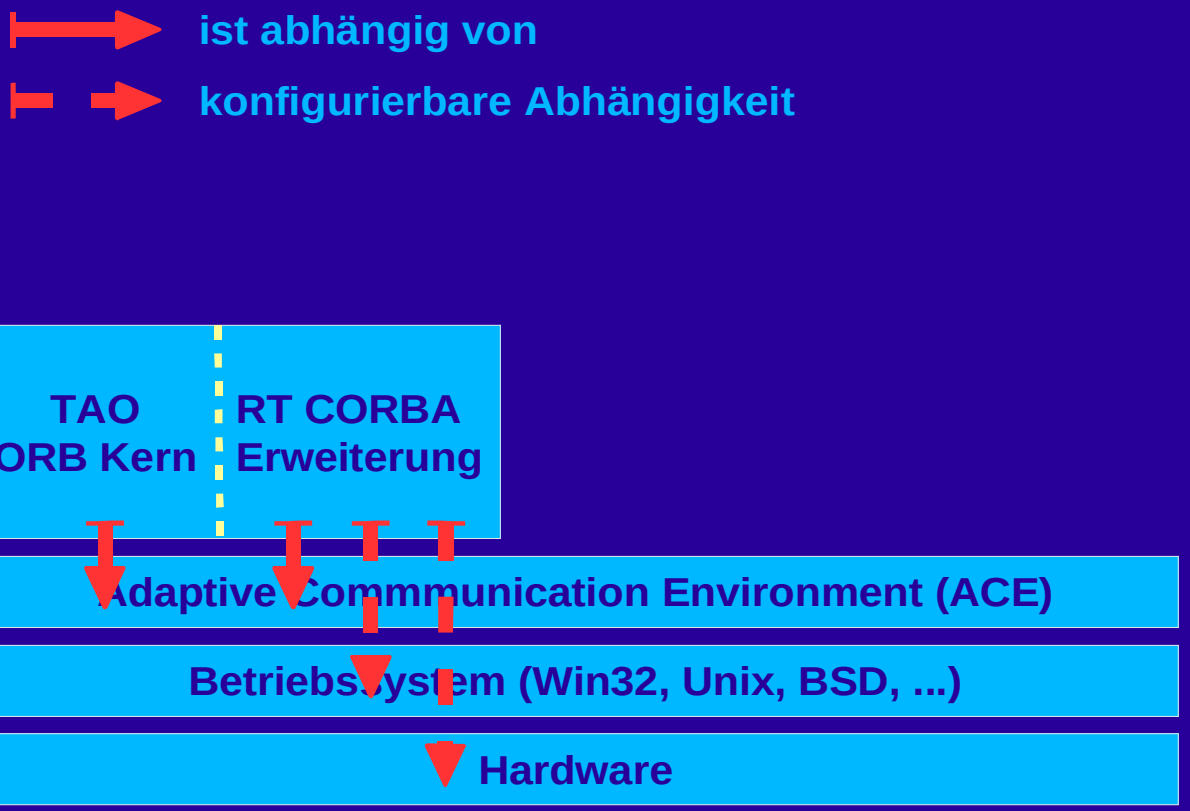
Systemarchitektur: TAO und Dienste

Systemarchitektur: TAO ORB-Kern

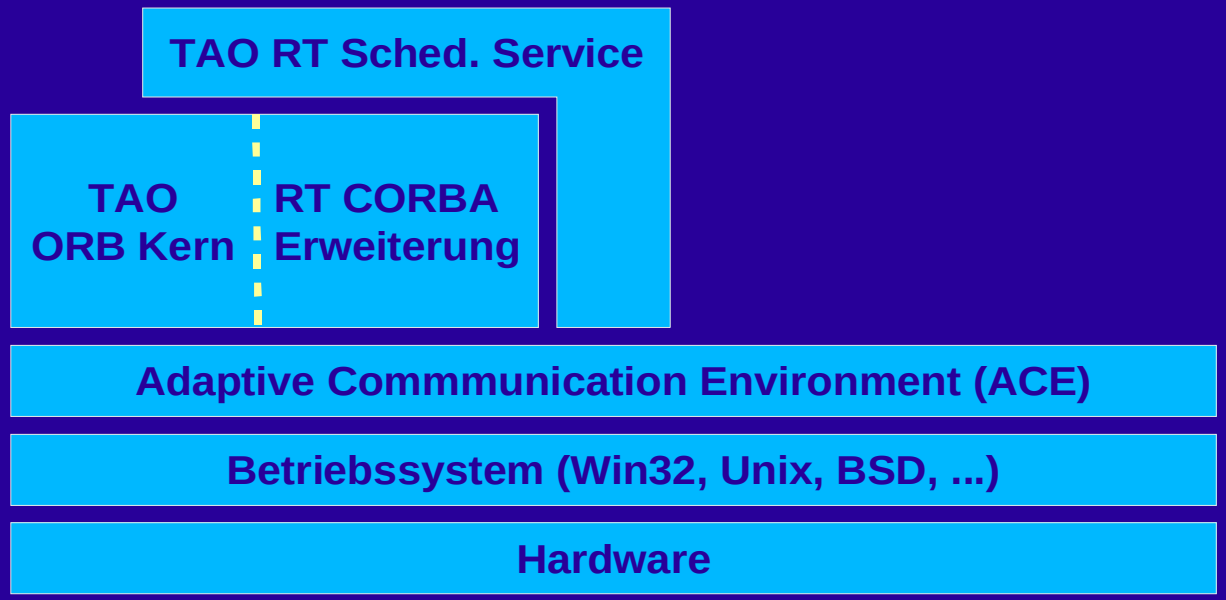
Systemarchitektur: TAO ORB-Kern



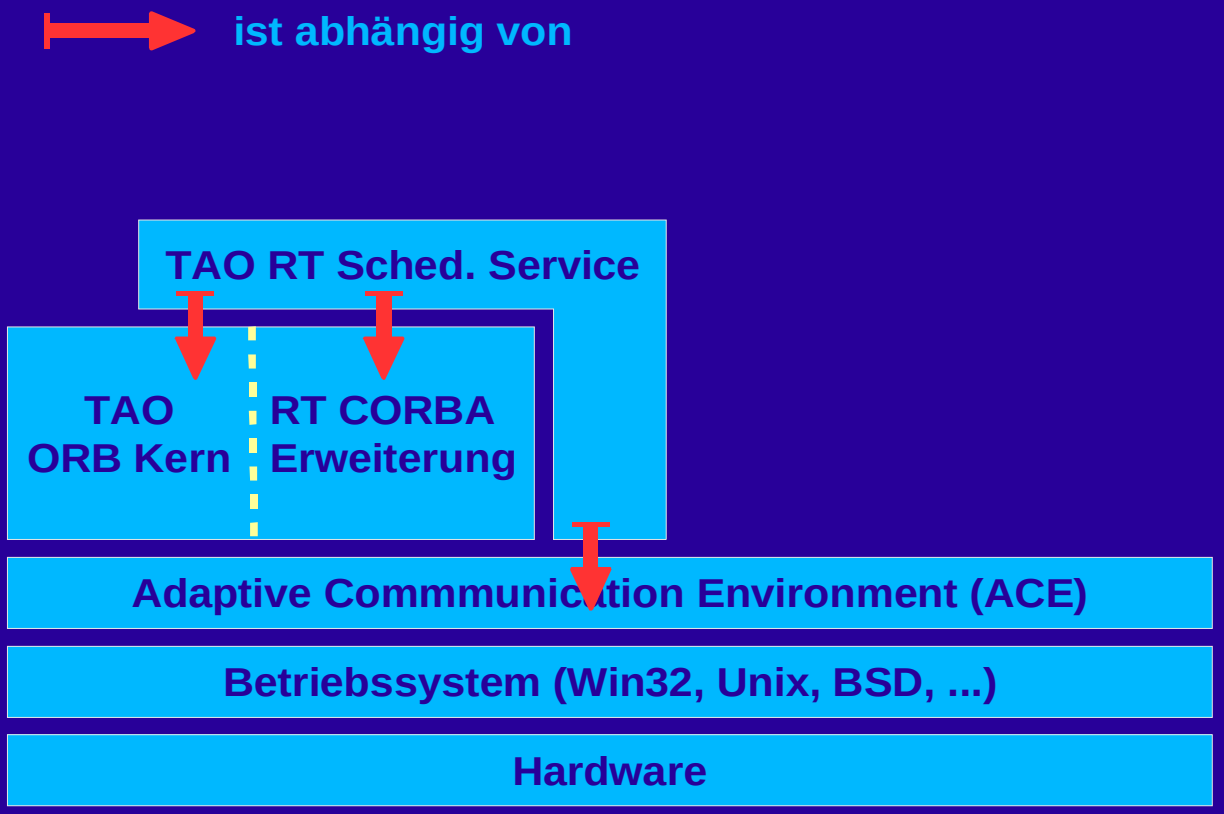
Systemarchitektur: TAO ORB-Kern



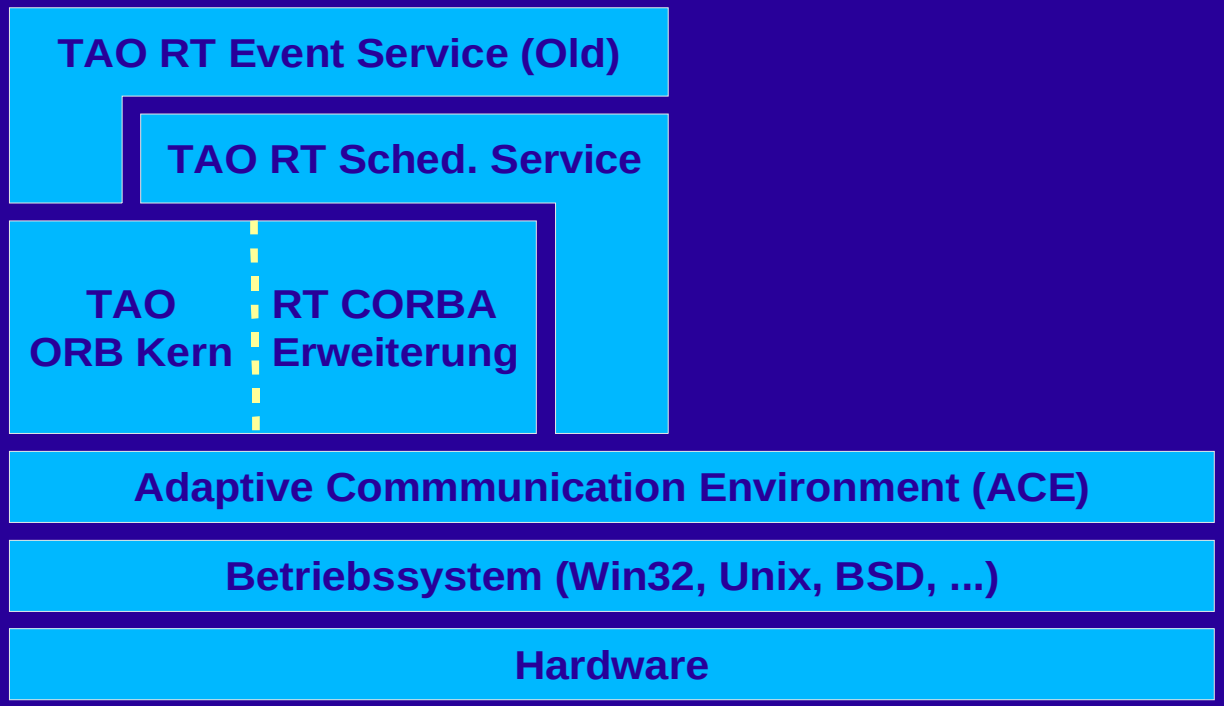
TAO Real Time Scheduling Service



TAO Real Time Scheduling Service

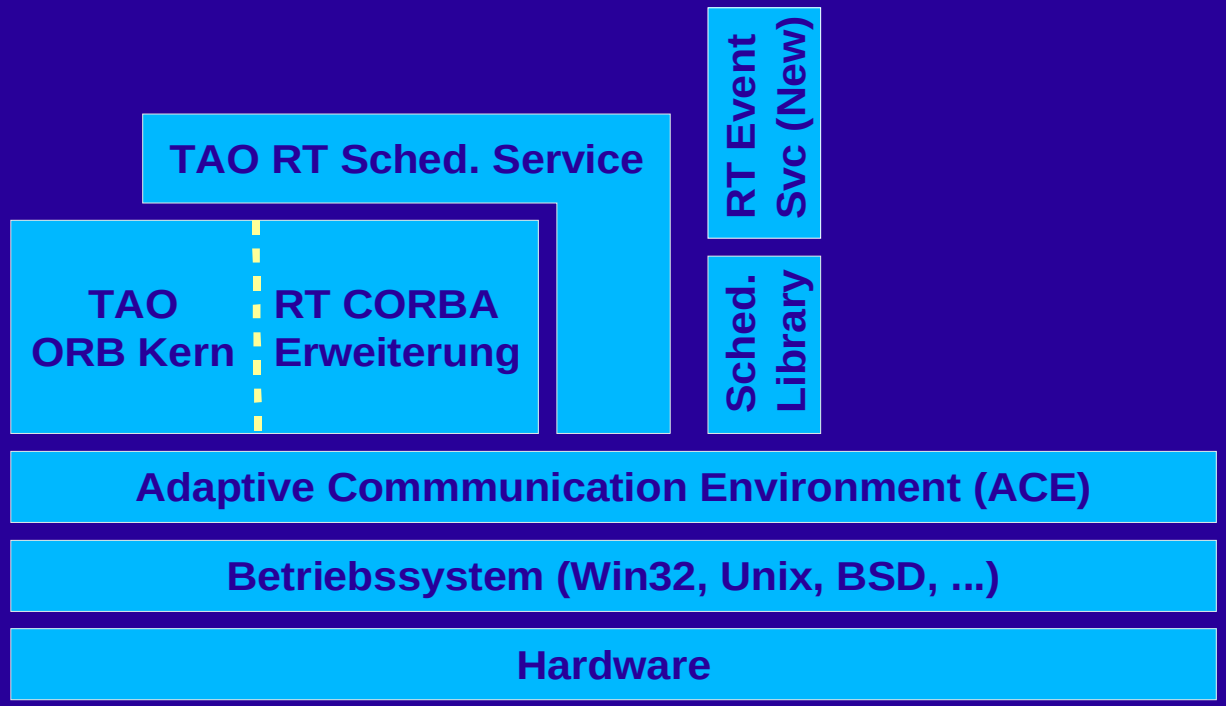


TAO Real Time Event Service (alt)

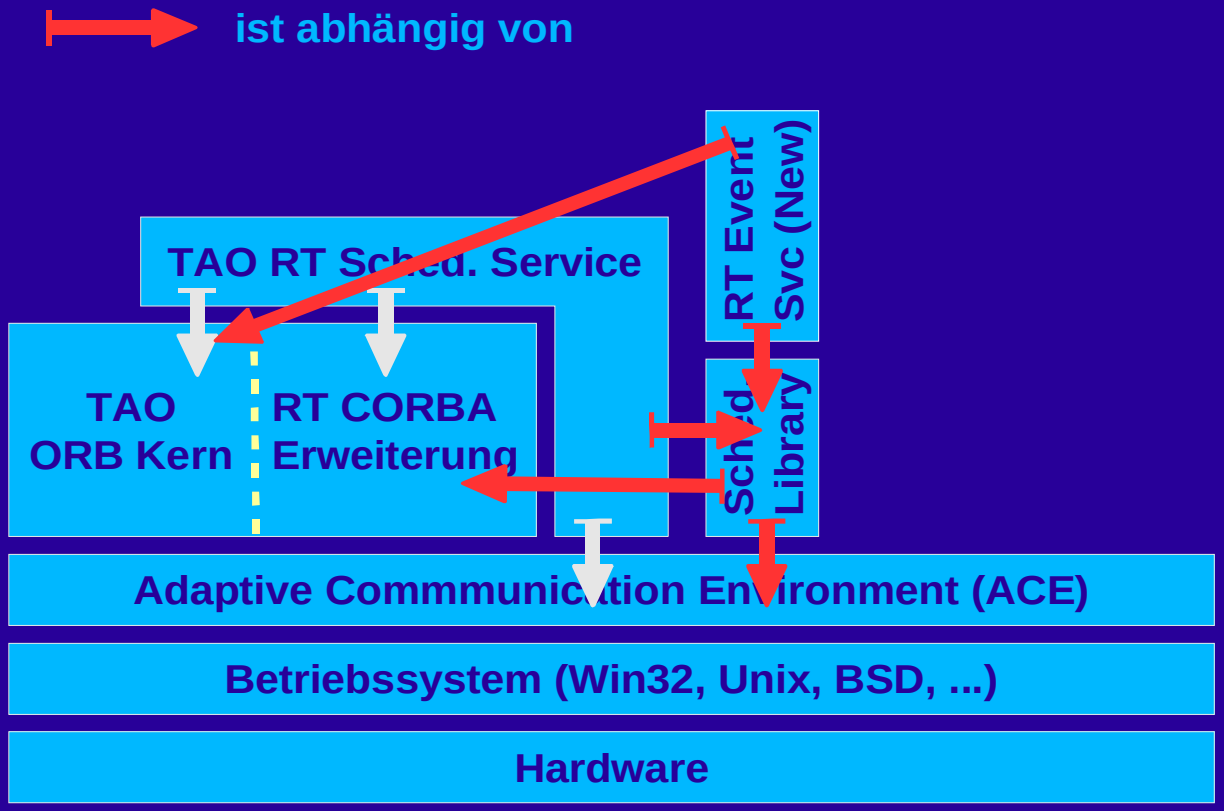


TAO Real Time Event Service (alt)

TAO Real Time Event Service (neu)

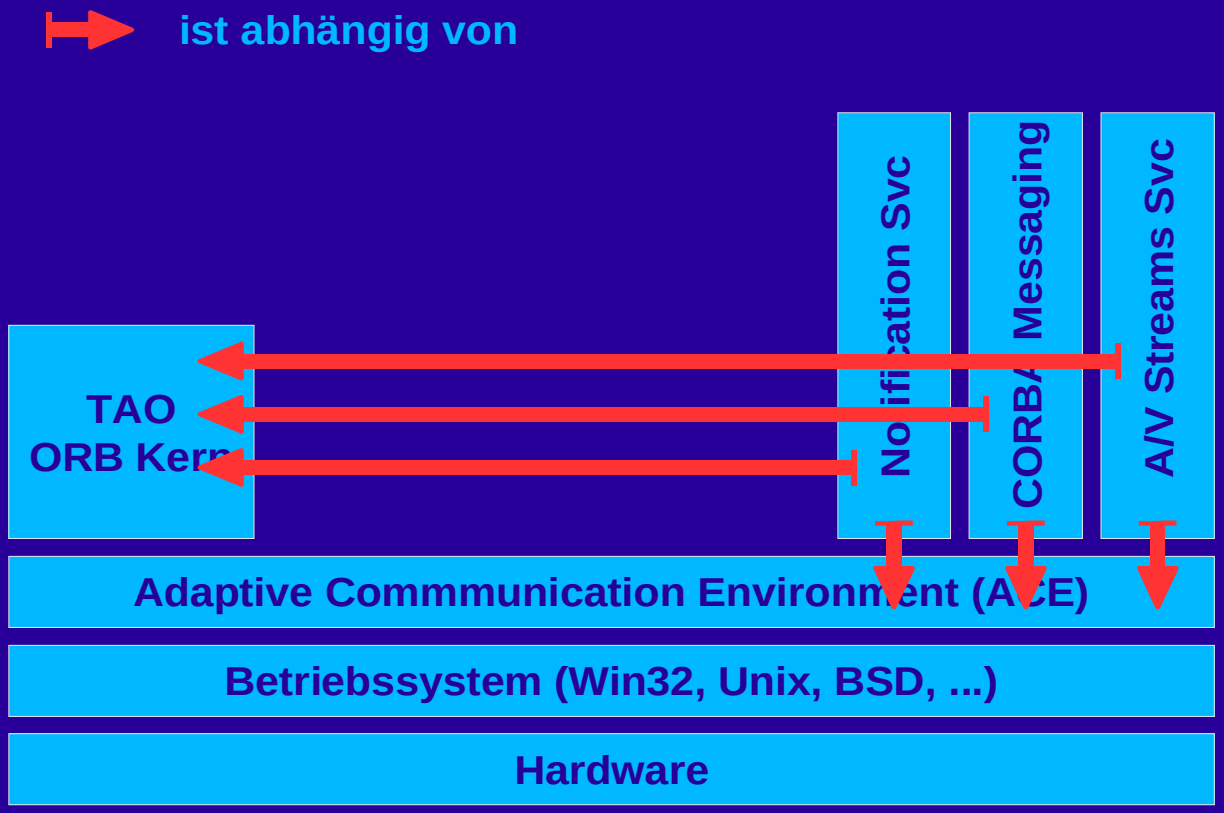


TAO Real Time Event Service (neu)

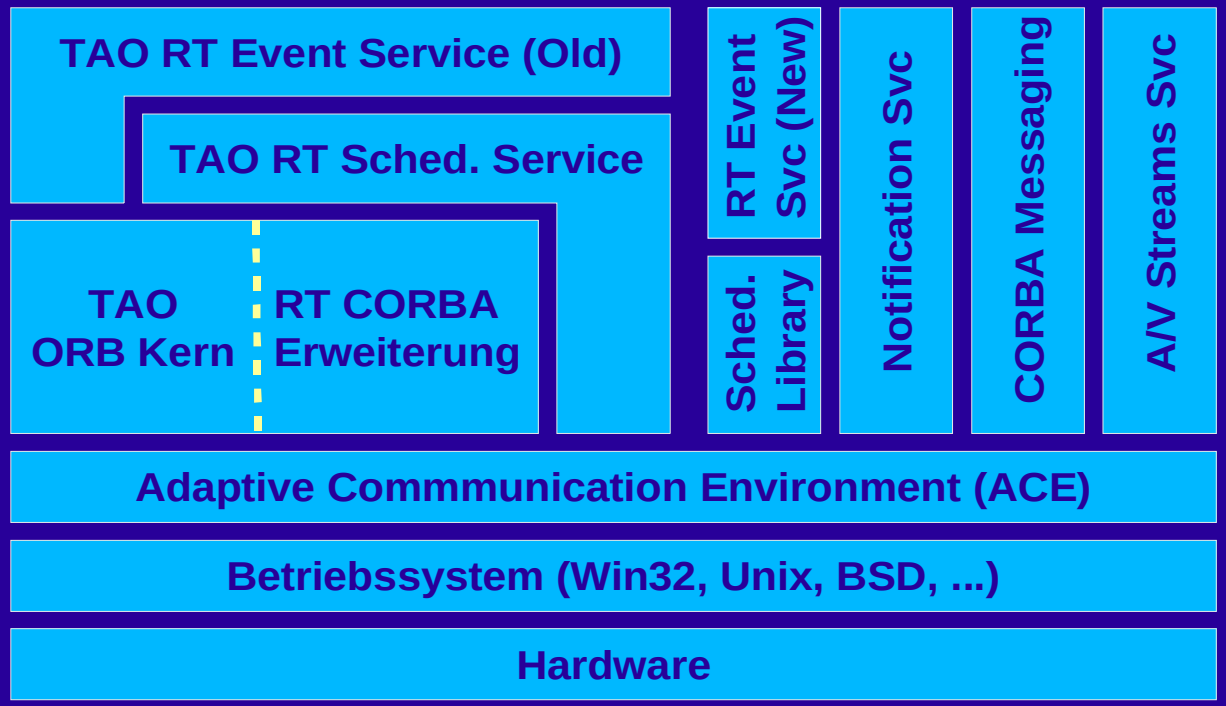


Weitere CORBA-Dienste von TAO

Weitere CORBA-Dienste von TAO



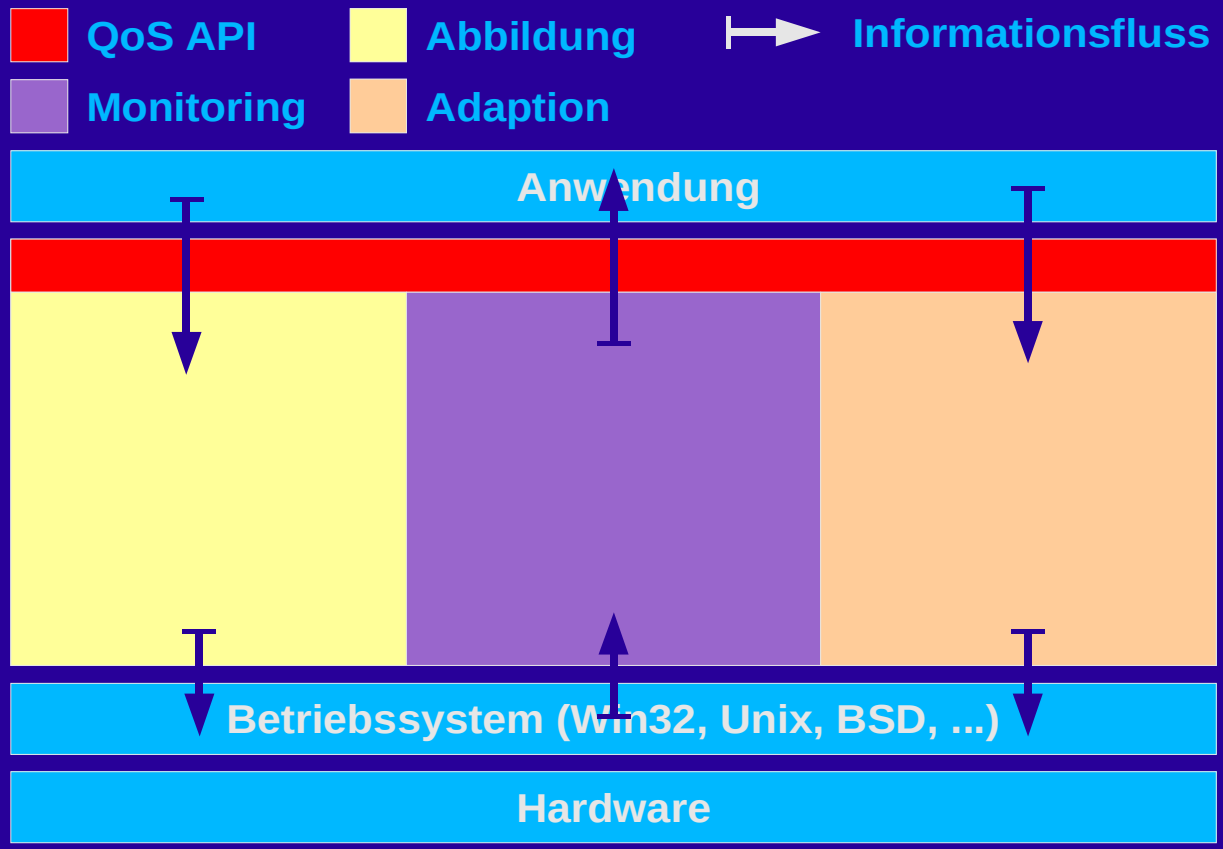
Systemarchitektur: Gesamtübersicht



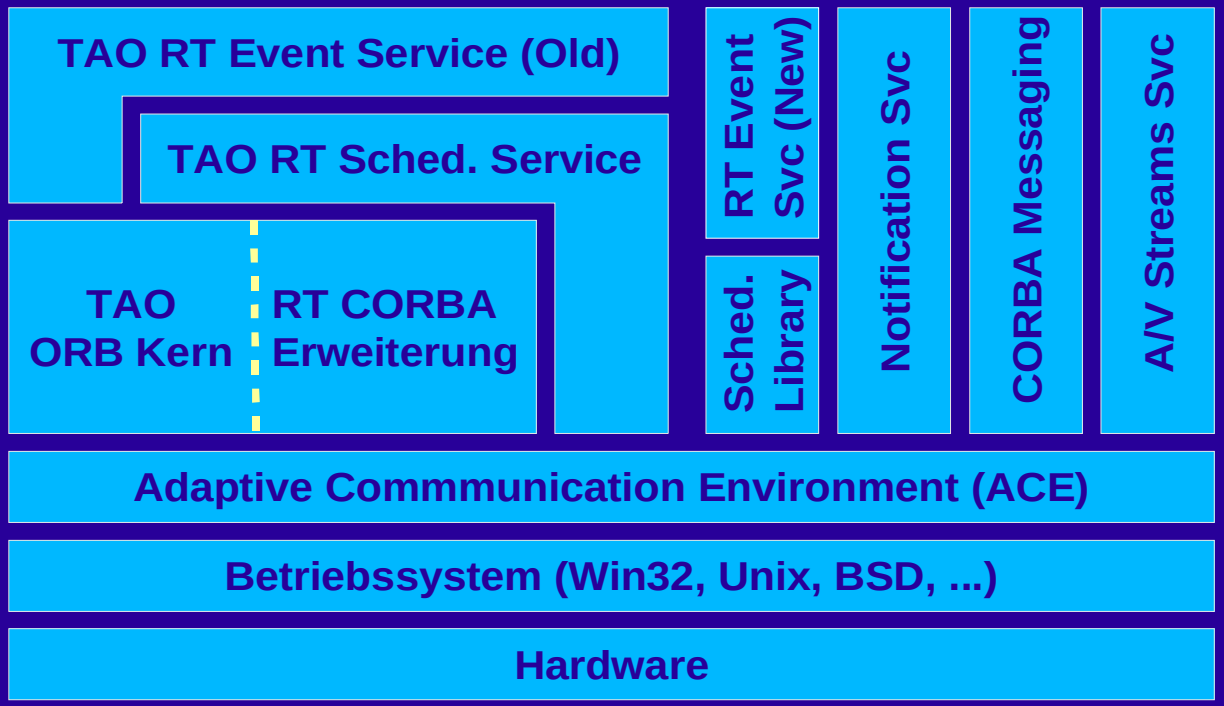
Vergleich ideal vs. real

- Abbildung ideale auf reale Architektur
  - Wo sind Entsprechungen der idealisierten Architekturkomponenten in der realen Architektur?
- Vergleich
  - Ähnlichkeiten und Unterschiede der beiden Architekturen?
- Begründung:
  - Warum ist die TAO-Architektur so, wie sie ist?
- Konsequenzen für Bearbeitung der Aufgabenstellung

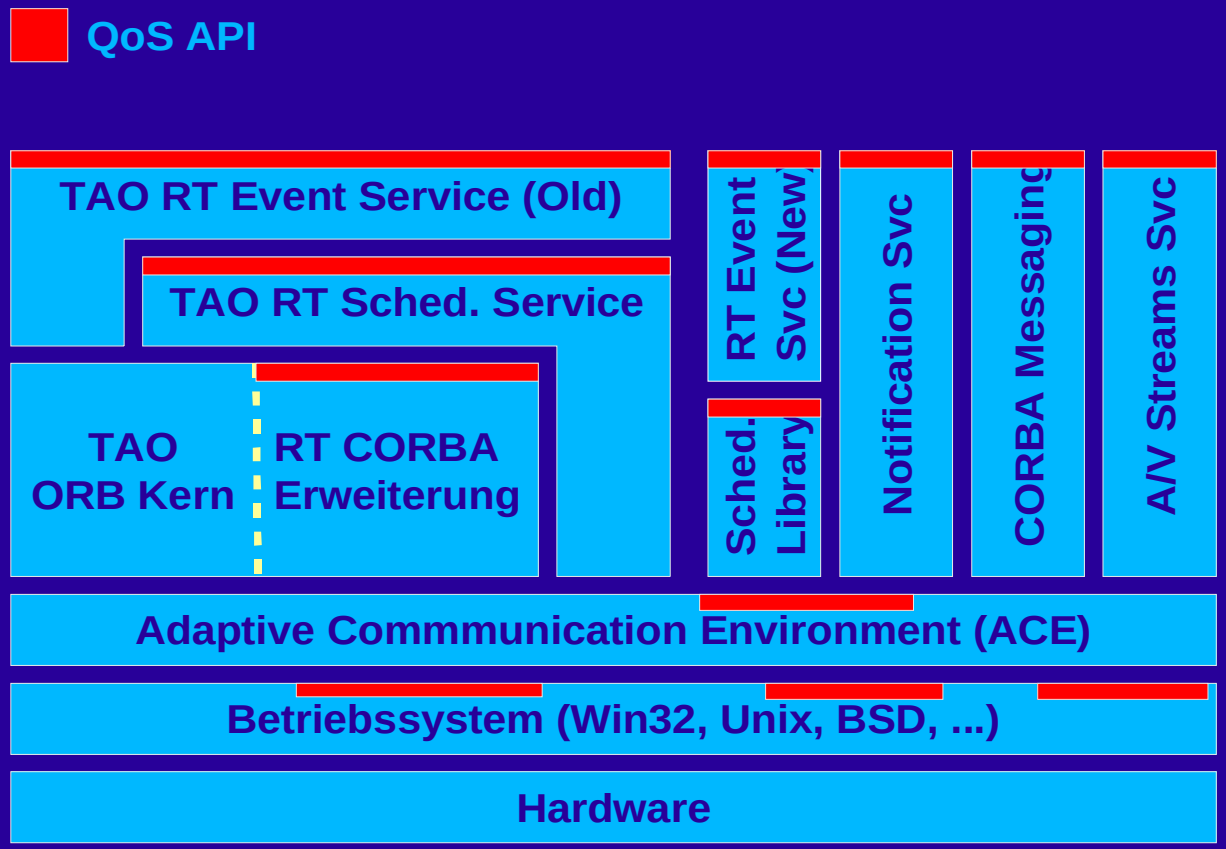
Idealisierte QoS-Architektur (wdh.)



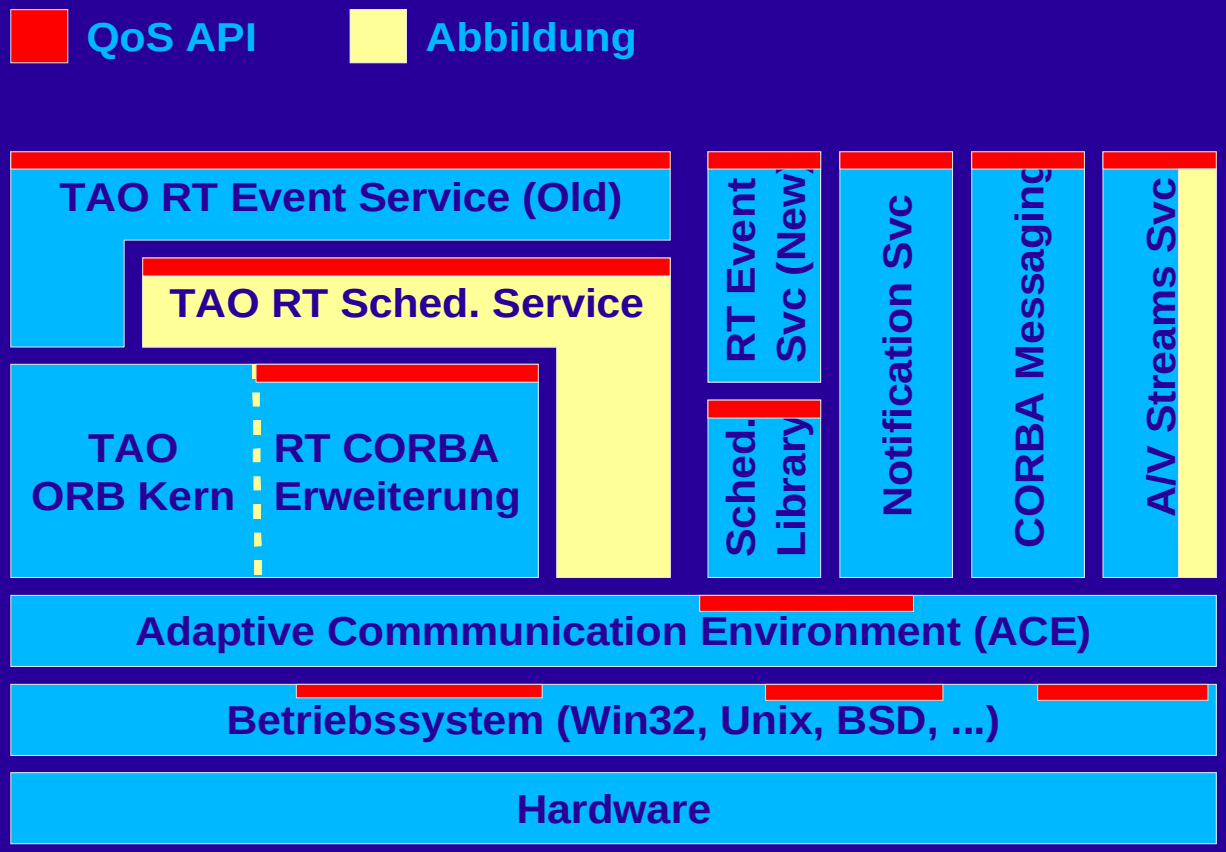
Reale Systemarchitektur



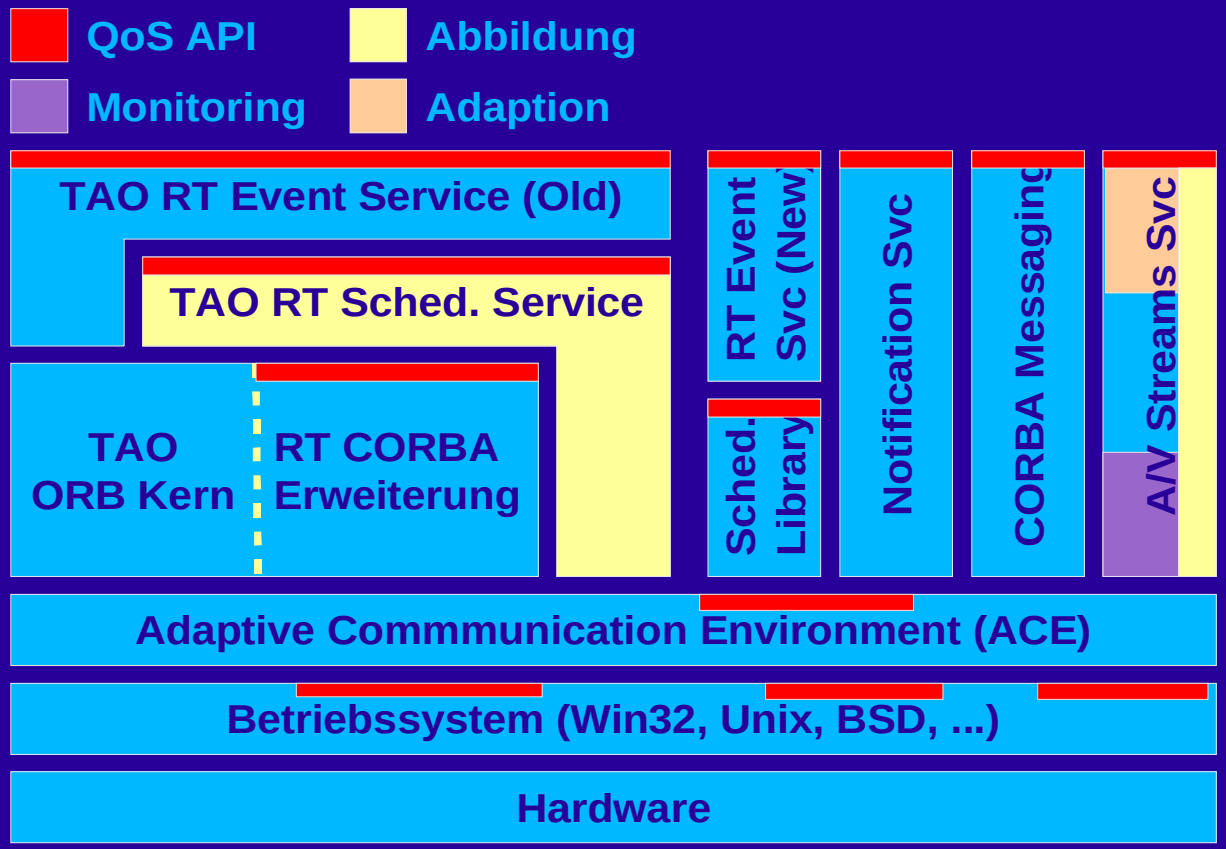
QoS-APIs in TAO und ACE



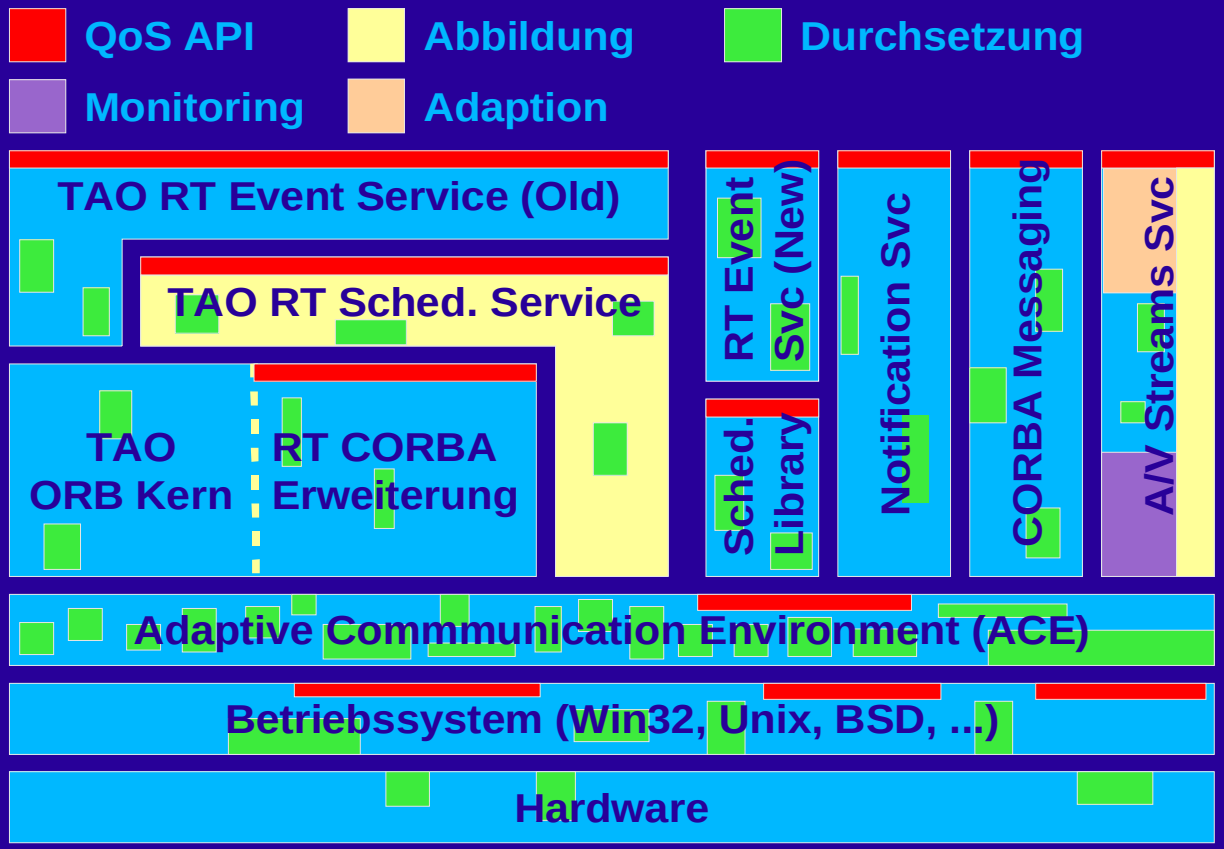
Abbildungskomponenten für QoS



Monitoring und Adaption von QoS



Grobarchitektur TAO und Dienste



TAO vs. ideale Architektur (1)

- Keine *einheitliche* QoS-Architektur
  - Vielzahl von Einzeldiensten und -komponenten
  - Dienstarchitektur meist durch CORBA-Spezifikation vorgegeben
- *Lokalität:*
  - Lokale QoS-APIs
  - lokale Abbildungsvorschriften
  - lokale Durchsetzungsmechanismen

TAO vs. ideale Architektur (2)

- Durchsetzungsmechanismen:
  - Betreffen überwiegend Ablaufsicht, physikalische Sicht
  - Zahlreiche Implementierungsdetails
  - Einsatz von Mustern und C++ Templates aus ACE
- Beispiele für Durchsetzungs*linien*:
  - CORBA A/V Streams Service
  - Demo-Anwendung

Gründe für *reale* TAO-Architektur

- Forschungsziele der TAO-Gruppe:
  - Unterstützung von QoS und RT
  - Beibehaltung von Standardarchitekturen
- *„Sponsored research“*:
  - Ziel: *funktionierende* Lösungen für *konkrete* Szenarien
- Empirisches Vorgehen:
  - System bauen, Probleme erkennen, System verbessern
- QoS-Unterstützung := Problem des *Feinentwurfs* und der *Implementierung*

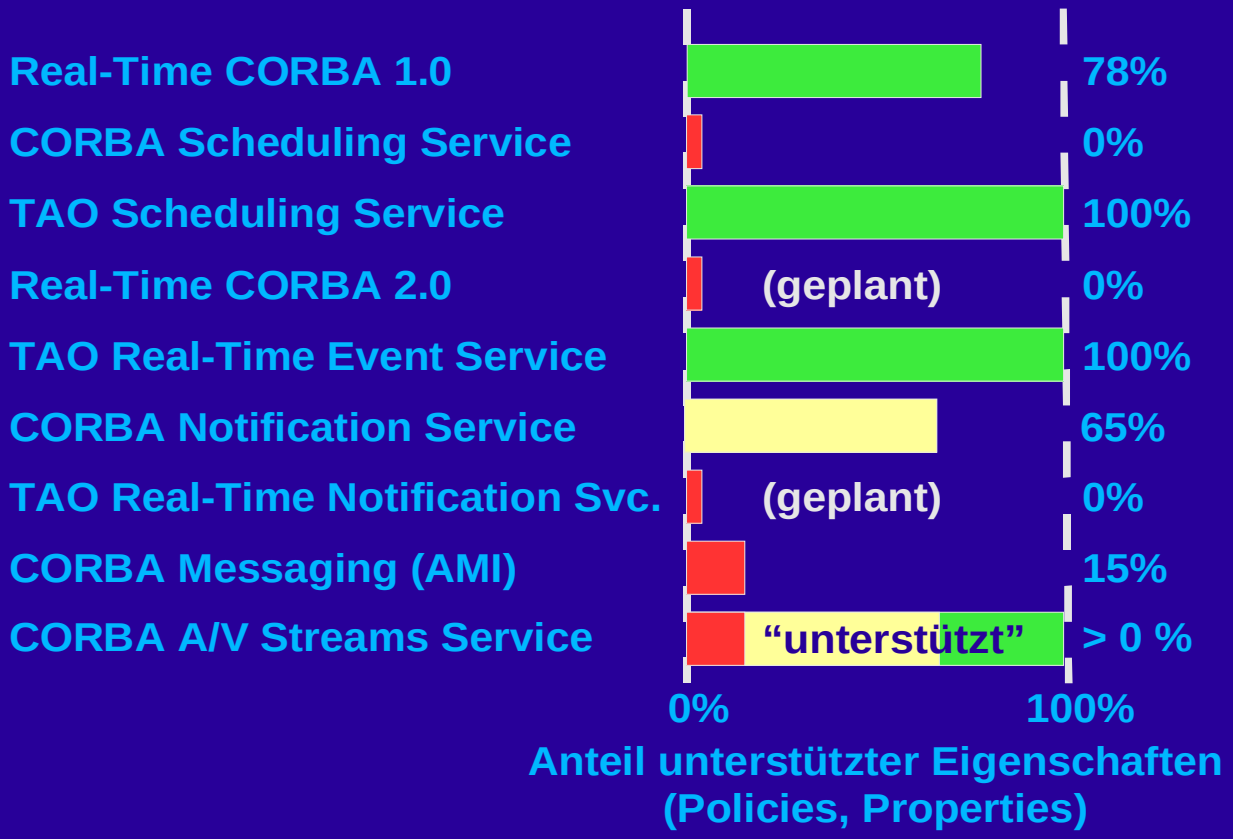
Konsequenzen

- *Getrennte* Betrachtung aller TAO-Systemkomponenten notwendig
- QoS-Durchsetzung in TAO untersuchen heißt:
  - *Details sammeln:*
    - Implementierungsdetails
    - Details des Feinentwurfes
- Nicht nach *QoS-Systemarchitektur* suchen!
  - Stattdessen: Ablauf- und physische Architektursicht betonen

Gliederung

- Aufgabenstellung
- Idealisierte QoS-Architektur
- QoS in ACE und TAO
- Unterstützungsgrad für QoS in TAO
- Beispielprogramm zur QoS-Durchsetzung
- Zusammenfassung, Ausblick

Unterstützungsgrad für QoS in TAO



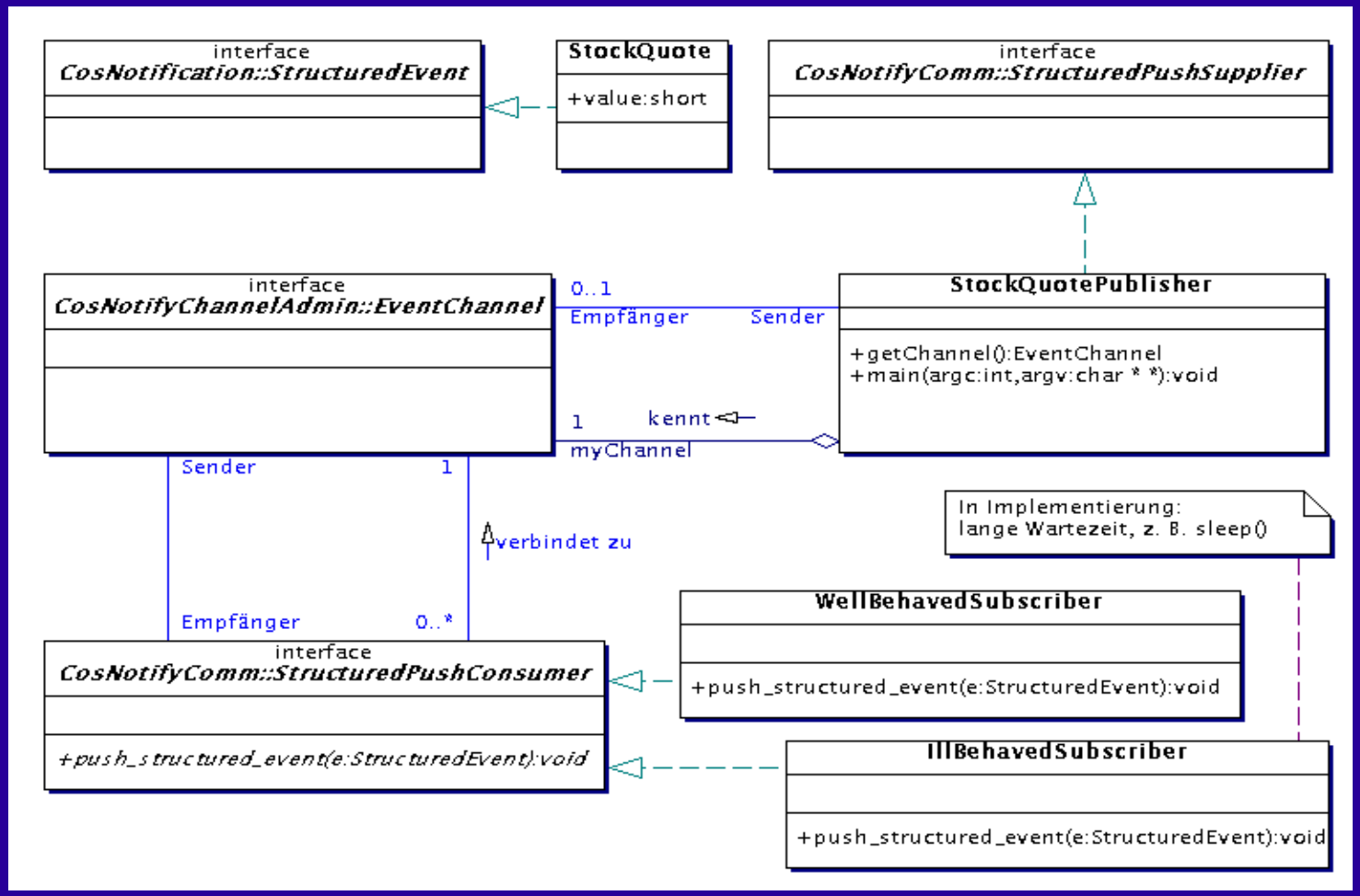
Gliederung

- Aufgabenstellung
- Idealisierte QoS-Architektur
- QoS in ACE und TAO
- Unterstützungsgrad für QoS in TAO
- Beispielprogramm zur QoS-Durchsetzung
- Zusammenfassung, Ausblick

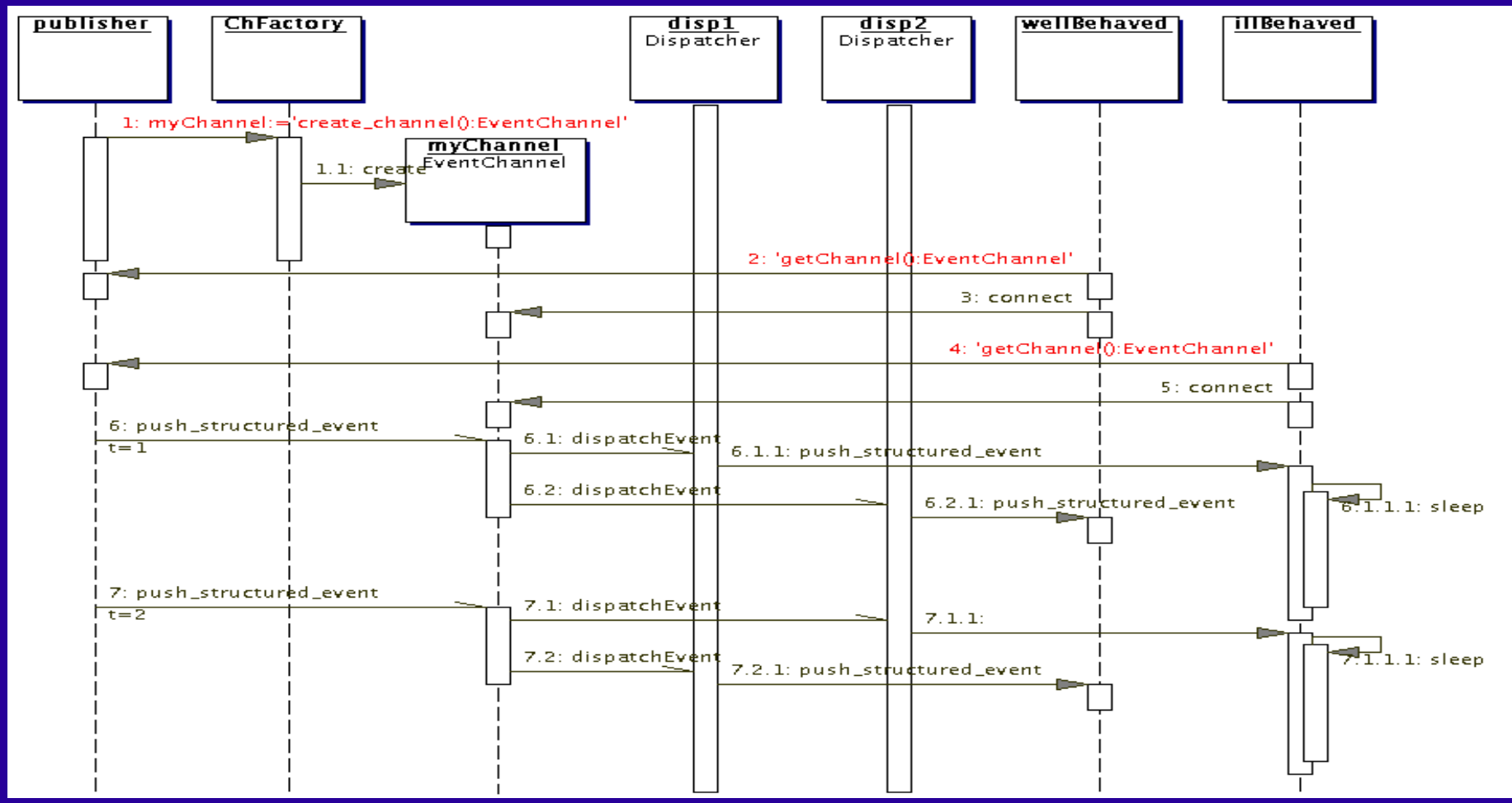
Beispielprogramm

- Szenario: Börseninformationsdienst
- Implementierung nutzt Notification Service
- Problem: bösartige Clients (Konkurrenzsituation)
- Lösungsversuche:
  - Verwendung von Active Objects zur Ereigniszustellung
  - Nachrichten mit Timeout versehen
  - Warteschlangen-Länge des Ereigniskanals begrenzen

Architektur Beispielanwendung



Active Objects: Sequenzdiagramm



Beispielprogramm: Experimente

- Komplexe Konfiguration erforderlich:
  - Beeinflussung der Durchsetzungsmechanismen
  - geforderte Dienstgüte nur bei korrekter Konfiguration
- Auslieferung stockt nach $t \div (b \cdot i)$ Nachrichten
  - t: Anzahl Ablauffäden (aktive Objekte)
  - b: Anzahl bösartiger Clients
  - i: Anzahl noch nicht ausgelieferter Informations-Ereignisse
- Je Empfänger und Ereignis ein AO blockiert

Beispielprogramm: Experimente (2)

- Versuch: Blockierung vermeiden
  - Begrenzung der Ereignis-Lebenszeit: Timeout
  - Begrenzung der Warteschlangenlänge je Empfänger
- Ergebnis: keine Verbesserung
- Folgerung:
  - Implementierung belegt AO für jedes *eintreffende* Ereignis
  - Keine weiteren Prüfungen nach Reservierung des AO
- Bewertung: QoS-Zusage ungenügend umgesetzt

Gliederung

- Aufgabenstellung
- Idealisierte QoS-Architektur
- QoS in ACE und TAO
- Unterstützungsgrad für QoS in TAO
- Beispielprogramm zur QoS-Durchsetzung
- Zusammenfassung, Ausblick

Zusammenfassung

- Sehr komplexes System
- Uneinheitliche QoS-Architektur
- *Lokale* Spezifikation und Durchsetzung von QoS
- Umfassende Konfigurierbarkeit
  - Zur korrekten Konfiguration Wissen über Durchsetzungsmechanismen in TAO notwendig (!)
- Mangelhafte Dokumentation
- Forschungsinstrument (nicht: *Produkt)*

Zusammenfassung (2)

- Experimentelles Prototyping vor TAO-Einsatz
- Grad der QoS-Unterstützung:
  - differiert je nach Dienst/Komponente stark
  - zum Teil nur „bedingt bekannt“

Ausblick: COMQUAD (1)

- Profitieren von der Erfahrung des TAO-Teams:
  - Nutzung der Muster und Implementierungsdetails
  - Vielzahl Muster „vorgefertigt“ in ACE enthalten
  - Nutzung von ACE als Implementierungsgrundlage sinnvoll
  - Nutzung geeigneterer Dokumentation (Bücher)
- TAO-Distribution der DOC-Gruppe:
  - in der Regel *nicht* einsetzbar für Produktionsumgebungen
  - eigene Distribution abspalten (hoher Aufwand)
  - alternativ: Nutzung kommerzieller Version

Ausblick: COMQUAD

- Genauer betrachten: A/V Streams Service
  - Nah an „idealer“ Architektur
  - kann eher Hinweise für COMQUAD-Architektur liefern als die vorgefundene TAO-Systemarchitektur

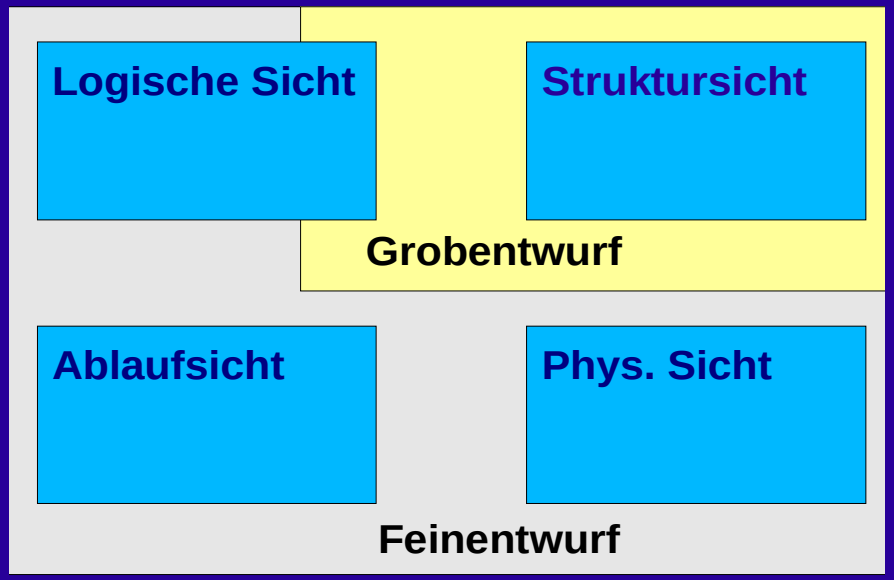
Fragen

?

Es folgen Ergänzungsfolien

- Falls erforderlich oder gewünscht.

4+1 View Model of Architecture



- Logische Sicht:
  - Funktionalität
- Struktursicht:
  - Subsysteme, Pakete
- Ablaufsicht:
  - Prozesse, Koordination
- Physische Sicht:
  - Komponenten, Hardware, Netzwerke