Embedded Software-Development
arvero
http://www.arvero.de

# Meine SPS kann Linux, und nun?

Ein Linux-basierter Software-Stack für
industrielle Eingebettete Systeme

**Autor: Christoph Stoidner**

# Zur Person / Firma

- Christoph Stoidner
- Diplom. Informatiker (FH)
- Embedded Software-Entwickler / Consultant
- Betriebsysteme, Laufzeitsysteme, Treiber im industriellen Umfeld
- Embedded Linux
- Gründer der arvero GmbH

# Motivation

Unser Entwicklungspartner BECK THE POINT OF SYNERGY möchte eine Linux-basierte Steuerung mit typischen SPS-Diensten. **Linux**, weil...

- Offener Standard
- Keine Hersteller-Abhängigkeit
- Alles im Griff dank Open-Source
- Quasi HW-unabhänig durch breit unterstütztes Architekturpalette
- Keine Lizenzkosten
- ...

# Motivation

## Wichtige SPS-Eigenschaften:

- Harte Echtzeit
- Feldbus-Protokolle
- Alarm-Management
- Logging-System
- Live-Daten (Diagnose, Wartung)
- Fernzugriff
- Update-Konzept (lokal und entfernt)
- GUI (für Browser und Display)
- SPS-typische Entwicklungsumgebung

# Motivation

## Wichtige SPS-Eigenschaften:

- Harte Echtzeit
- Feldbus-Protokolle
- Alarm-Management
- Logging-System
- Live-Daten (Diagnose, Wartung)
- Fernzugriff
- Update-Konzept (lokal und entfernt)
- GUI (für Browser und Display)
- SPS-typische Entwicklungsumgebung

**zusätzliche Gateway-Funktionalität**

# Rahmenbedingungen

- **Kleiner Footprint**

  → ab 32MB RAM / 32MB Flash
- **Niedrige Latenzzeiten**

  → unter ~60us (Context-Switch, ISR)
- **Schnelles Booten**

  → unter 10s
- **Auch ohne MMU verwendbar**

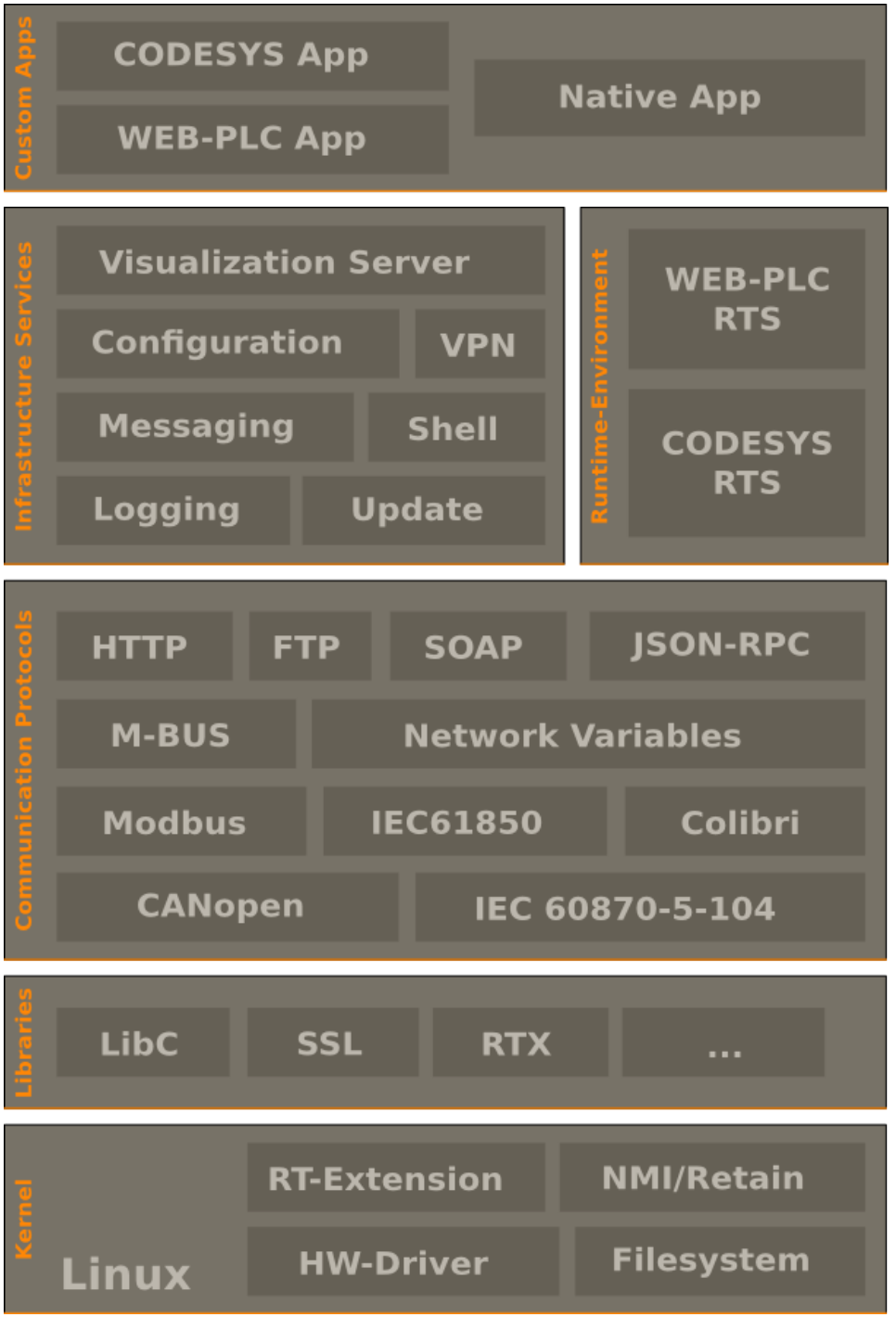
# Rahmenbedingungen

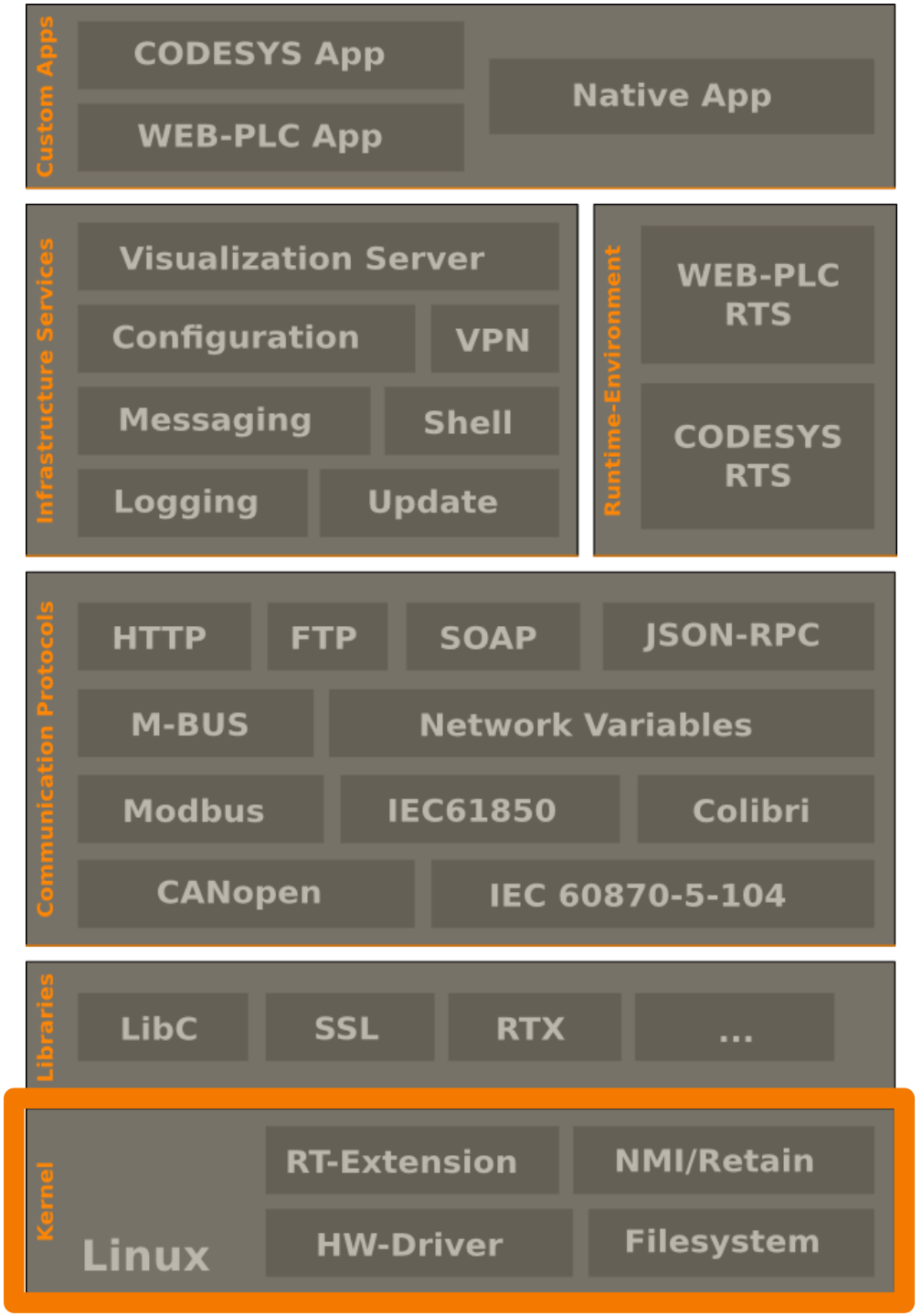
**High End Platform**

| | |
|---|---|
| **Prozessor:** | Freescale i.MX28 @ 450MHz (ARM9) |
| **Speicher:** | 128MB DDR2, 2GB e-MMC Flash, 64kB EEPROM |
| **Schnittstellen:** | 2xEthernet (10/100), 2xCAN, 5xUSB, 1xSDIO/MMC, 2xI²C, 2xI2S, 2xSPI, GPIO, 8xADC, 8xPWM, 24bit TFT-Interface |

**Low End Platform**

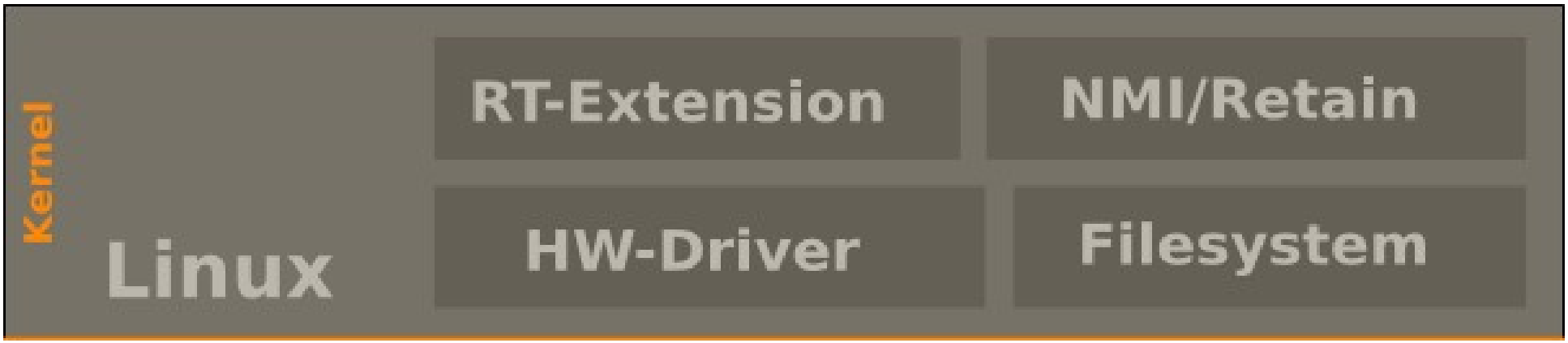
| | |
|---|---|
| **Prozessor:** | ARM Cortex M3 @ 180MHz |
| **Speicher:** | 32MB RAM, 32MB Flash |
| **Schnittstellen:** | ... |


Custom Apps
CODESYS App
WEB-PLC App
Native App
Infrastructure Services
Visualization Server
Configuration
VPN
Messaging
Shell
Logging
Update
Runtime-Environment
WEB-PLC RTS
CODESYS RTS
Communication Protocols
HTTP
FTP
SOAP
JSON-RPC
M-BUS
Network Variables
Modbus
IEC61850
Colibri
CANopen
IEC 60870-5-104
Libraries
LibC
SSL
RTX
...
Kernel
Linux
RT-Extension
NMI/Retain
HW-Driver
Filesystem



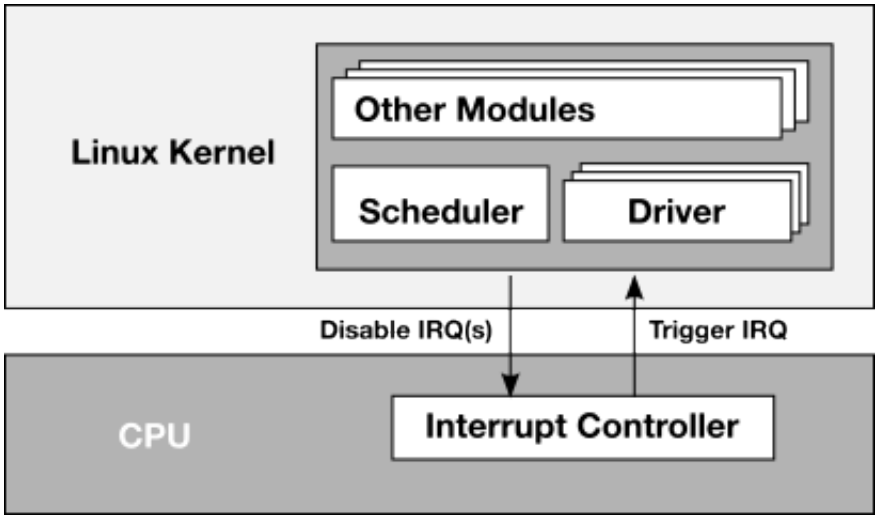
# Kernel

- Mainline
- NMI/Retain, z.B. via ARM FIQ Handler
- Hardware-Treiber (RS485, ...)
- Realtime-Extension für harte Echtzeit



# Realtime-Extension

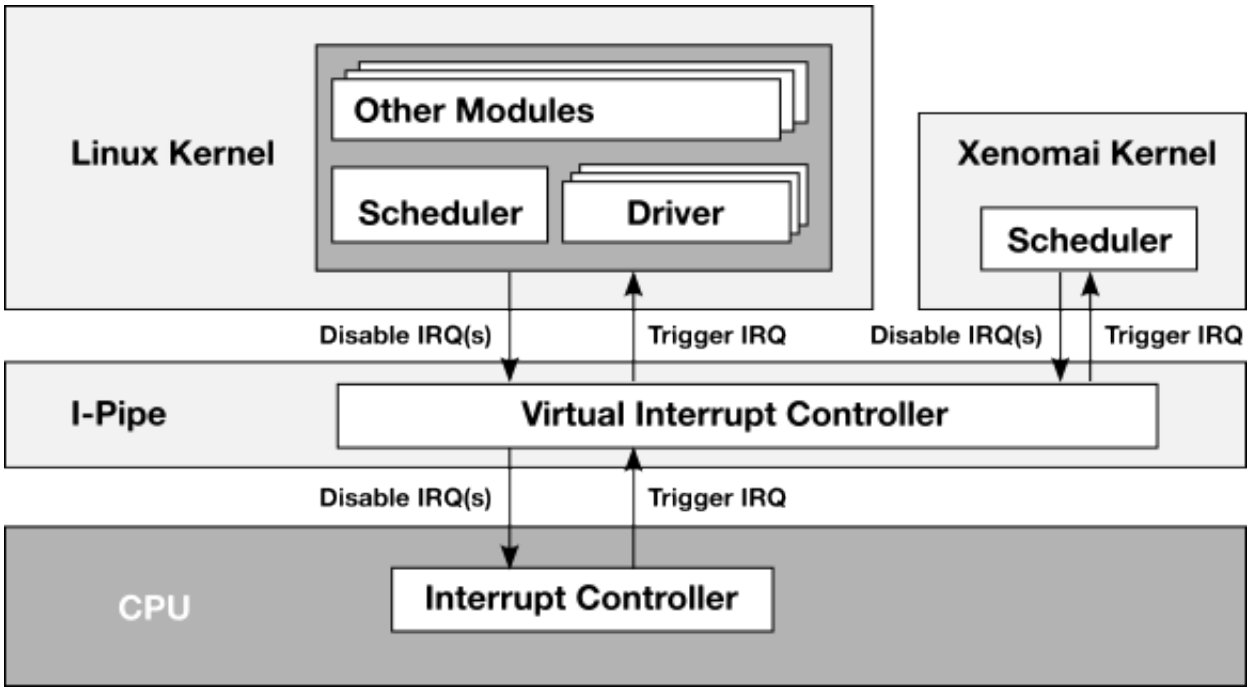
**Preemptive Linux Kernel mit *Preempt RT Patch***



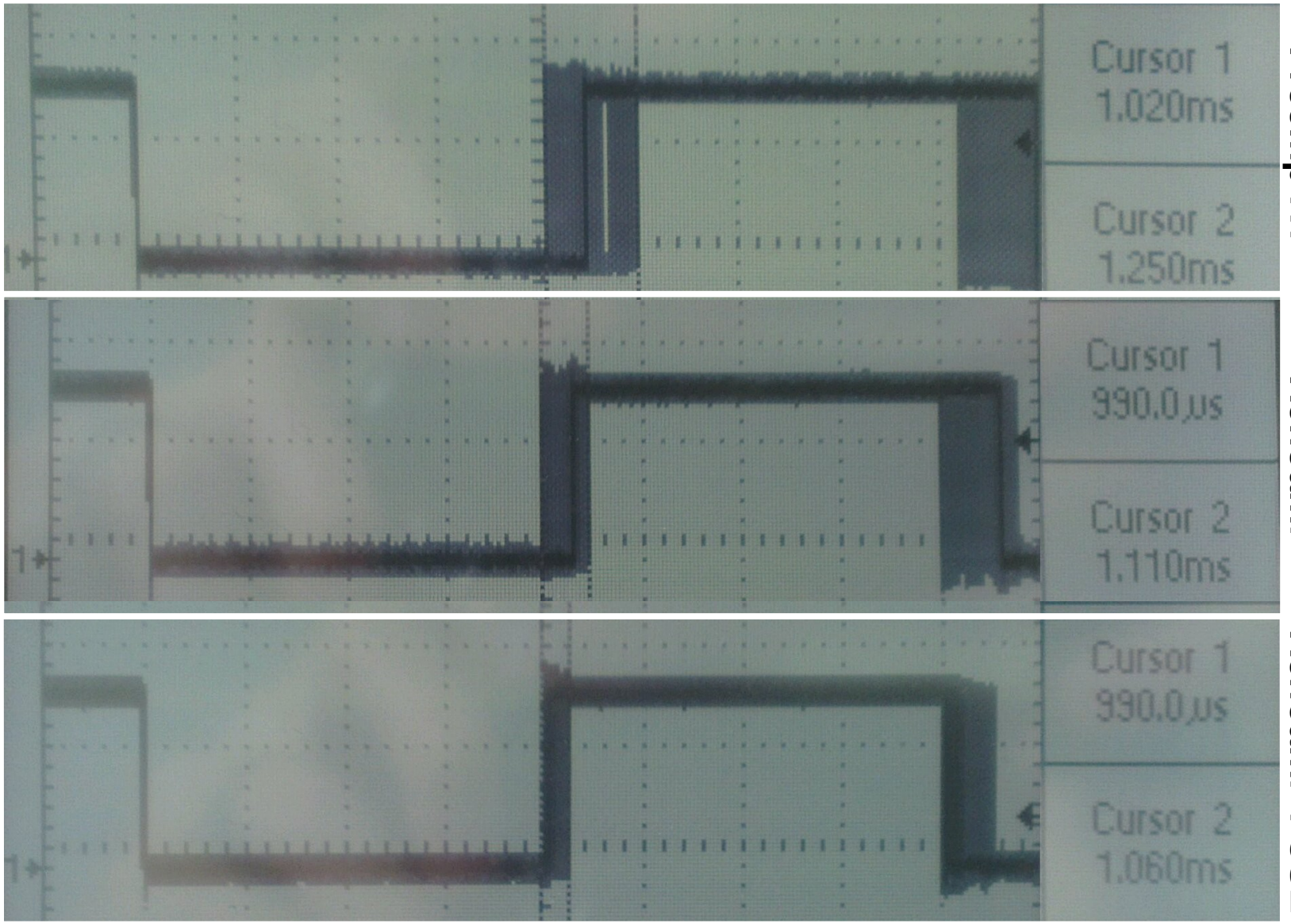

- Keine Interrupts sperren
- Lösung für Priority Inversion
- ISRs in ISR-Threads

# Realtime-Extension

**Co-Kernel mit *I-Pipe* und *Xenomai***



# Latenzzeit-Messungen



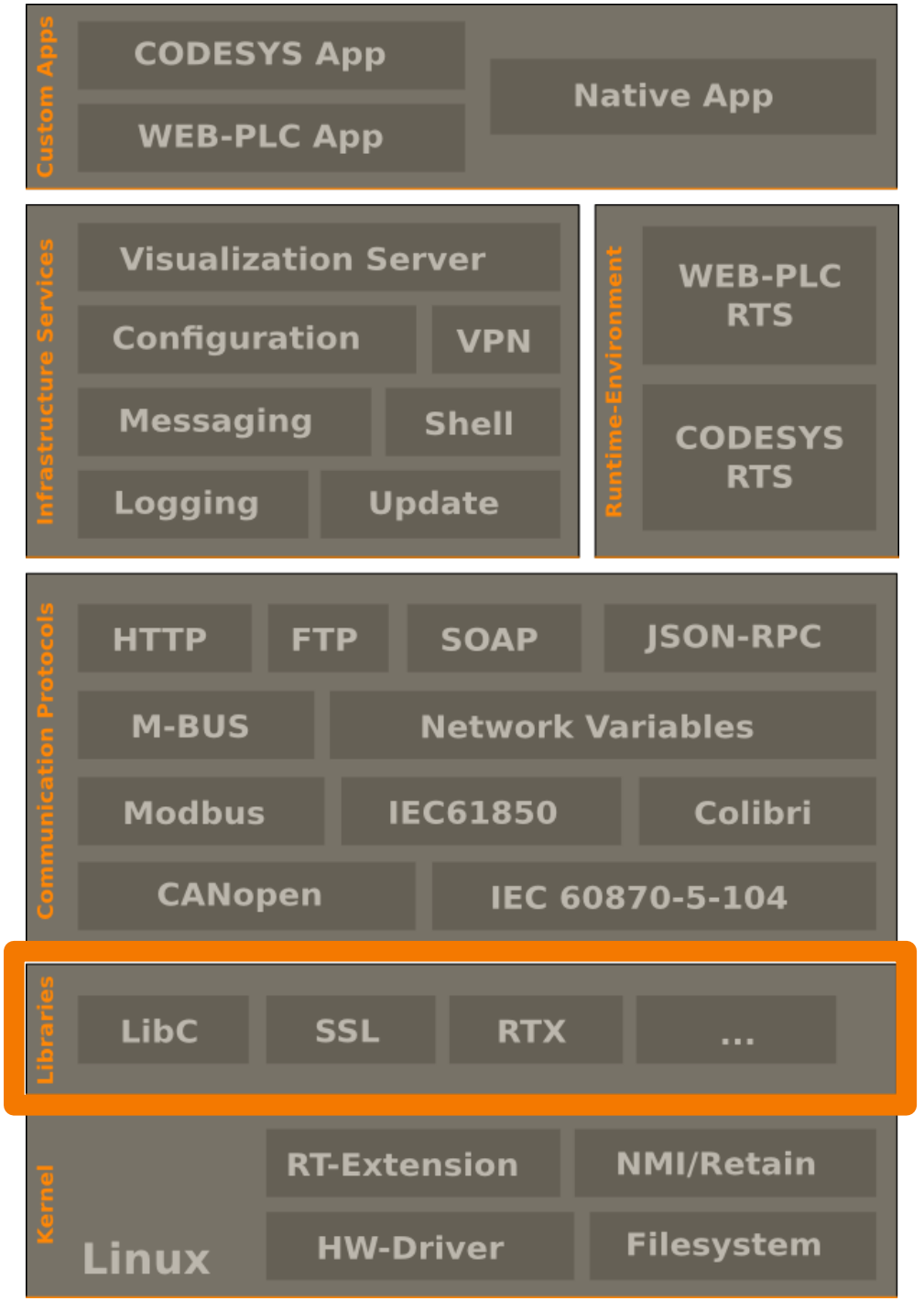
# Weitere Vorteile Co-Kernel

- Kleiner/überschaubarer Kernel
- Keine Beeinträchtigung durch Non-Realtime Module im Linux Kernel insbesondere durch Treiber möglich
- Komfortable API für Userspace (z.B. auch für Interrupts)
- RTOS-Typische Features vorhanden, z.B.
  - → Suspendieren einzelner threads anstelle von Prozessweiten pthread_kill()
  - → Prioritätsbasierte Semaphoren
  - → Sperren des Task-Schedulings
- Co-Kernel transparent für Userspace Application

# Motivation

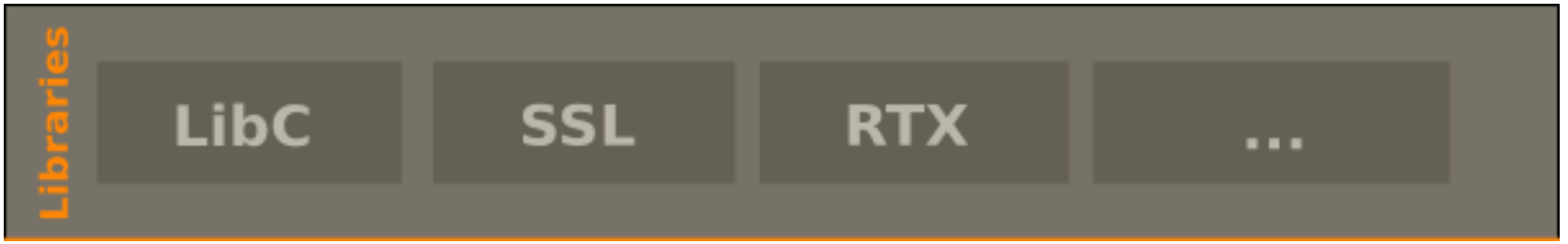
## Wichtige SPS-Eigenschaften:

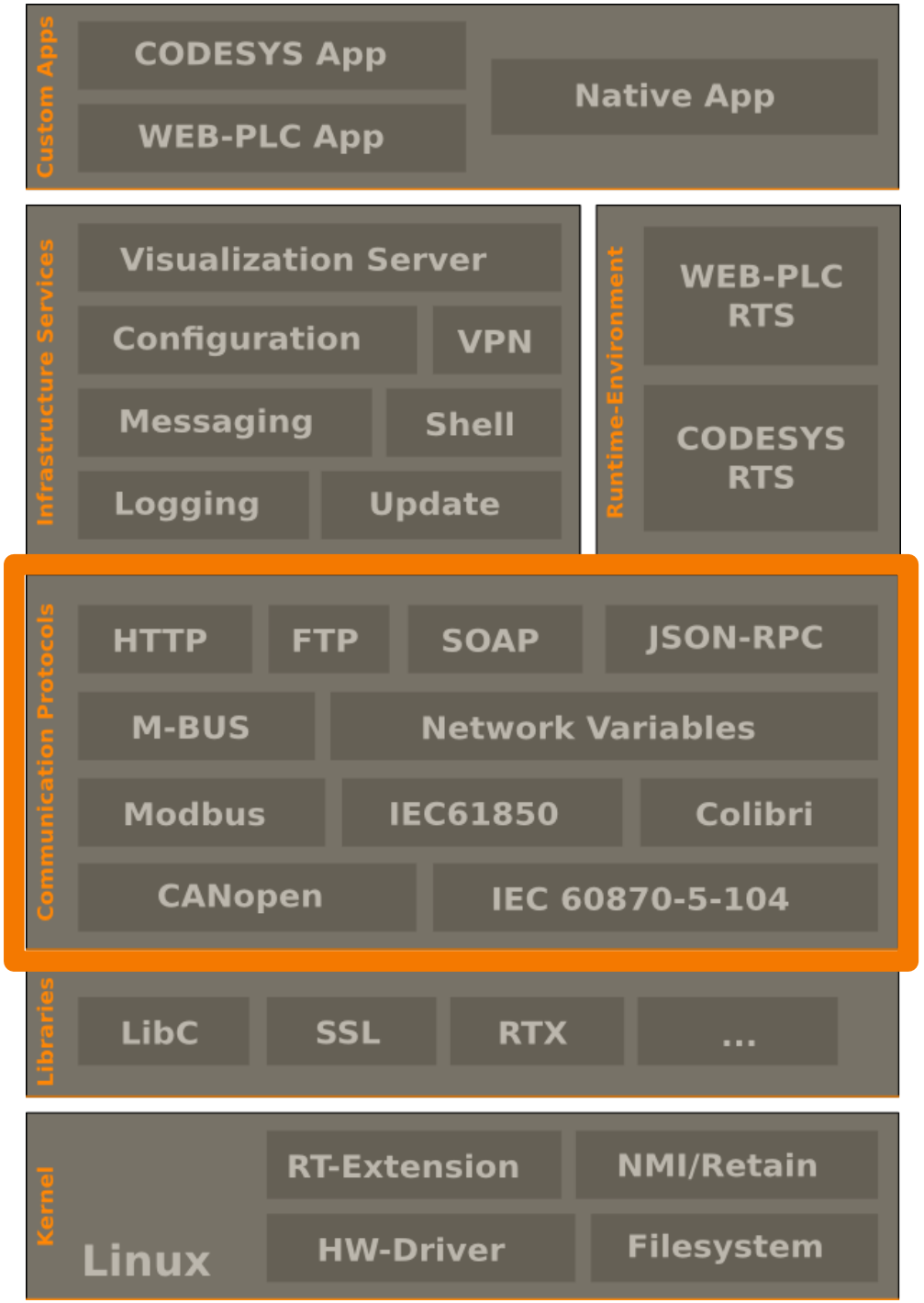
- ✓ Harte Echtzeit
- Feldbus-Protokolle
- Alarm-Management
- Logging-System
- Live-Daten (Diagnose, Wartung)
- Fernzugriff
- Update-Konzept (lokal und enfernt)
- GUI (für Browser und Display)
- SPS-typische Entwicklungsumgebung



# Libraries

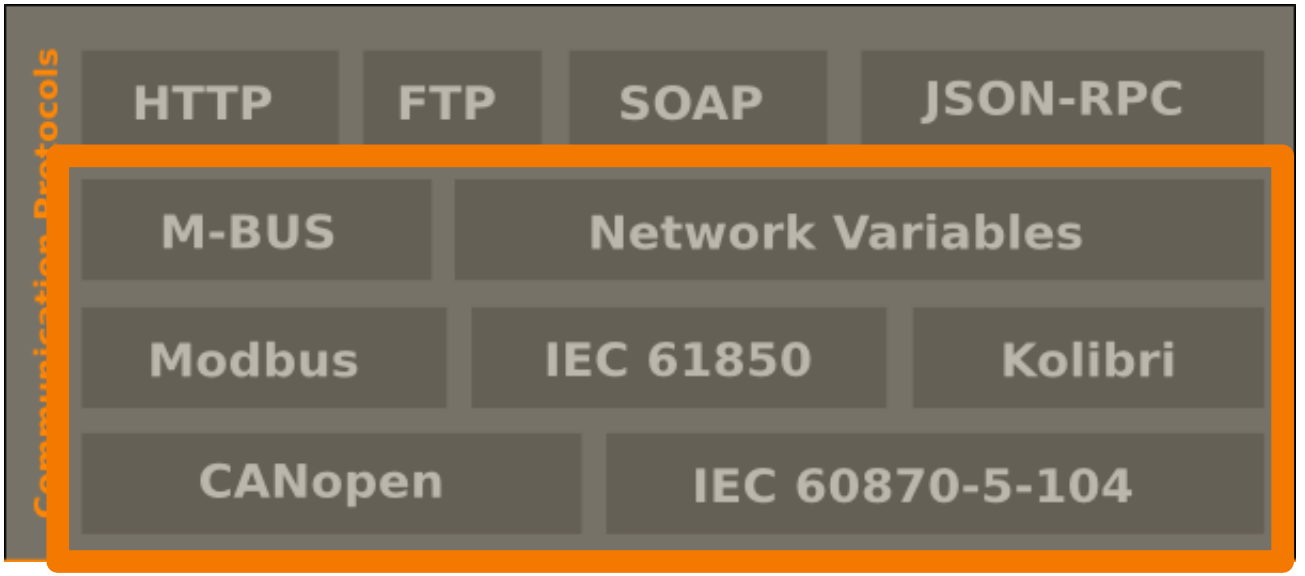
- Standard C-Bibliothek (glibc vs. ulibc)
- Basis Kommunikation (SSL, BSD, ...)
- RTX-Bibliothek (Realtime Multitasking-API)
- ACHTUNG: Keine *shared objects* ohne MMU!





# Communication Protocols

# Communication Protocols




Typische industrielle Protokolle

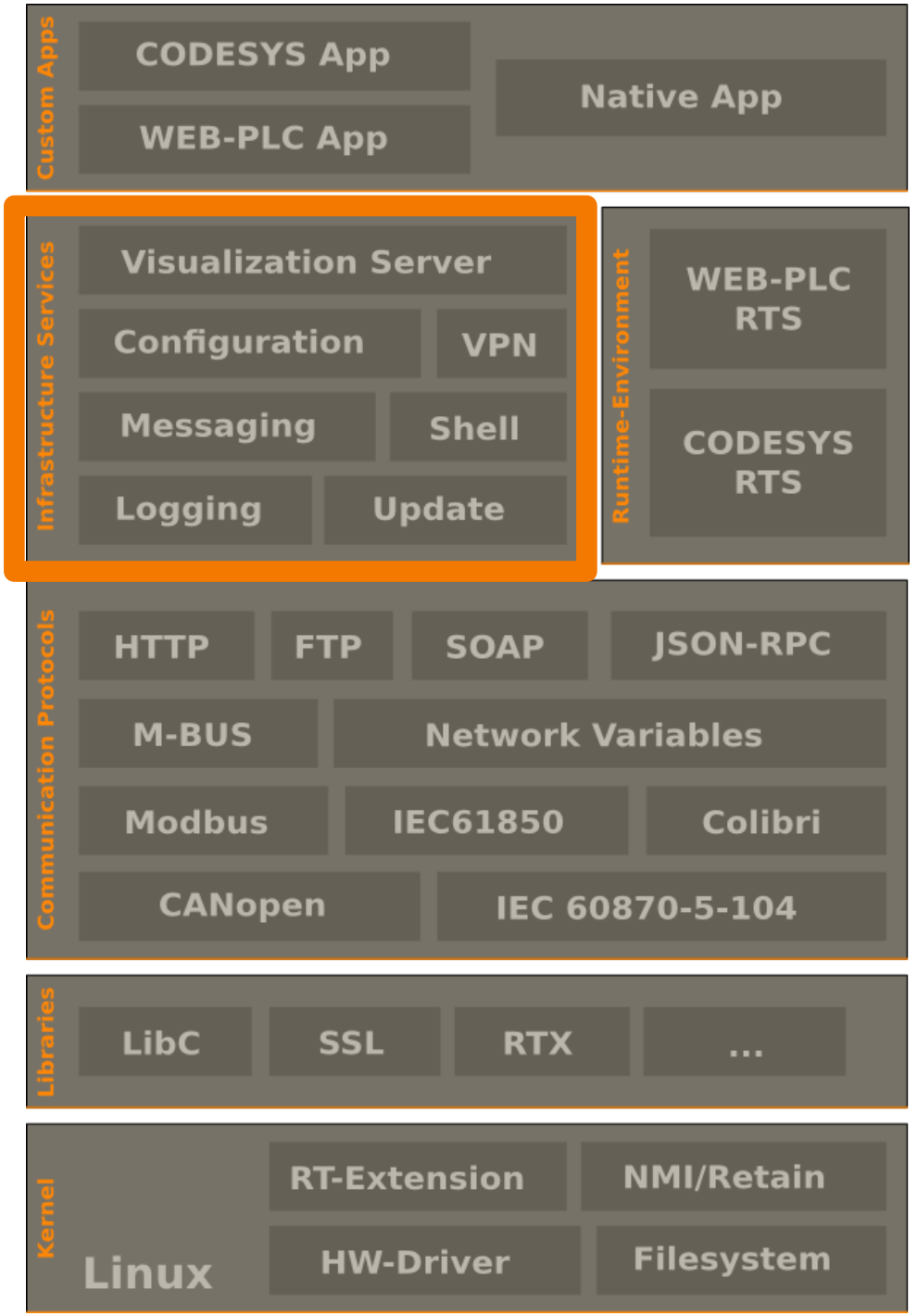
# Communication Protocols

## Typische IT-Protokolle

# Motivation

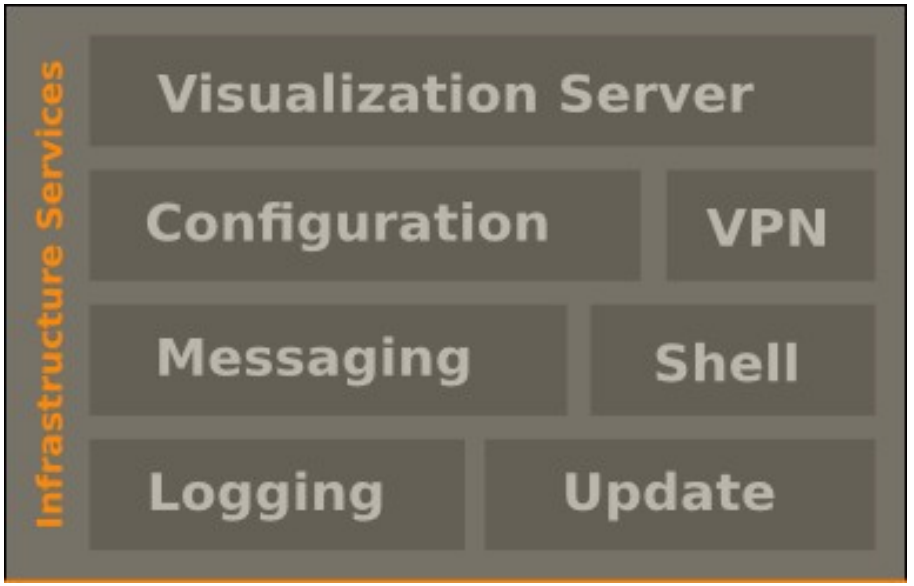
## Wichtige SPS-Eigenschaften:

- ✓ Harte Echtzeit
- ✓ Feldbus-Protokolle
- Alarm-Management
- Logging-System
- Live-Daten (Diagnose, Wartung)
- Fernzugriff
- Update-Konzept (lokal und entfernt)
- GUI (für Browser und Display)
- SPS-typische Entwicklungsumgebung


Custom Apps
CODESYS App
WEB-PLC App
Native App
Infrastructure Services
Visualization Server
Configuration
VPN
Messaging
Shell
Logging
Update
Runtime-Environment
WEB-PLC RTS
CODESYS RTS
Communication Protocols
HTTP
FTP
SOAP
JSON-RPC
M-BUS
Network Variables
Modbus
IEC61850
Colibri
CANopen
IEC 60870-5-104
Libraries
LibC
SSL
RTX
...
Kernel
Linux
RT-Extension
NMI/Retain
HW-Driver
Filesystem

# Infrastructure Services

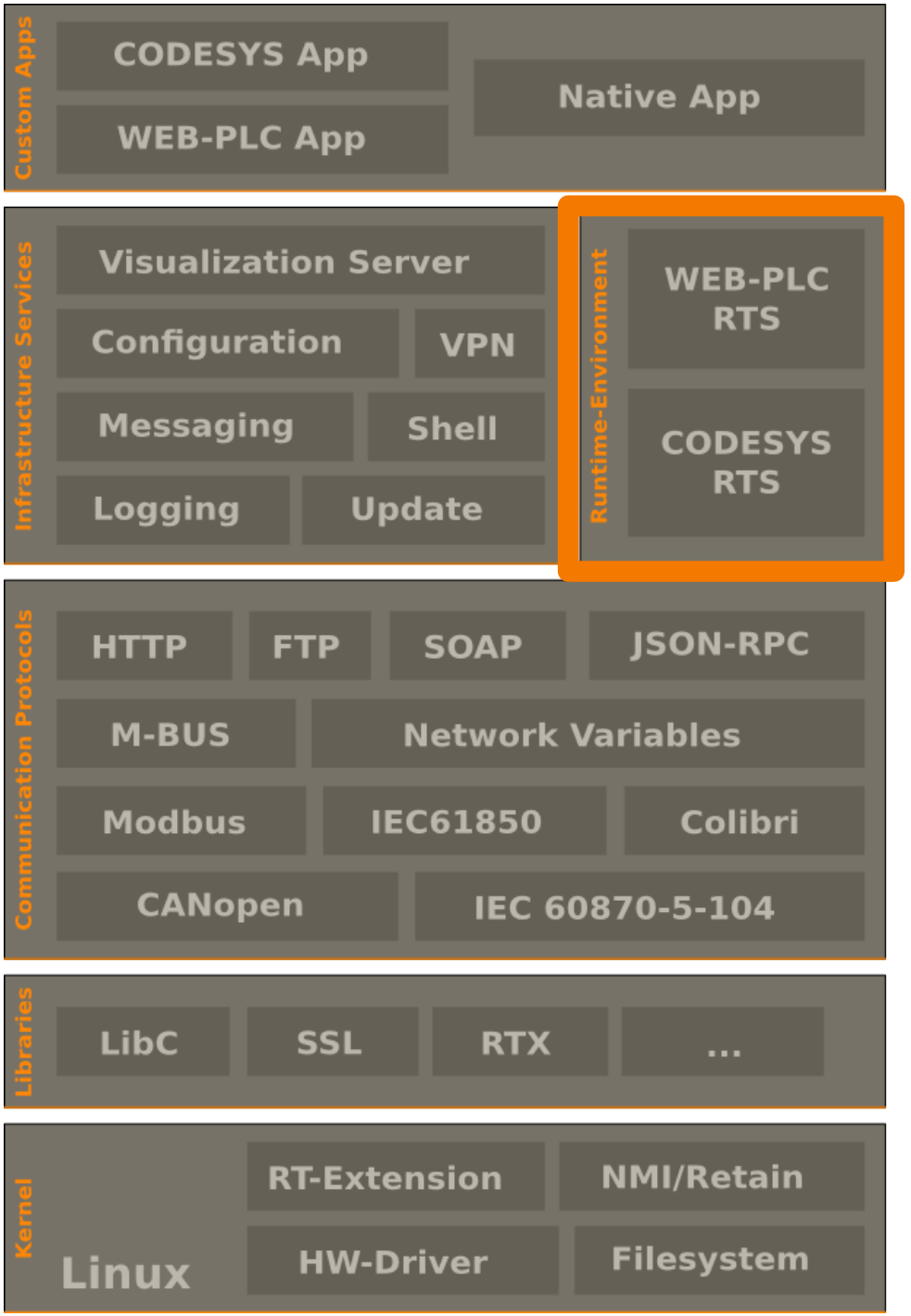
- Logging-Manager sammelt Laufzeitmeldungen
- Firmware/App-Update lokal oder von entfernt
- Messaging informiert bei Alarmen per SMS/Email
- Shell (Unix- oder Windows-like) erlaubt technische Kontrolle
- Konfiguration aller Komponenten per Browser
- Frei gestalterbare Visu im Browser oder LCD
- Vollzugriff per VPN
- Live-Daten per Cloud-Adapter



# Motivation

## Wichtige SPS-Eigenschaften:

- ✓ Harte Echtzeit
- ✓ Feldbus-Protokolle
- ✓ Alarm-Management
- ✓ Logging-System
- ✓ Live-Daten (Diagnose, Wartung)
- ✓ Fernzugriff
- ✓ Update-Konzept (lokal und entfernt)
- ✓ GUI (für Browser und Display)
- SPS-typische Entwicklungsumgebung



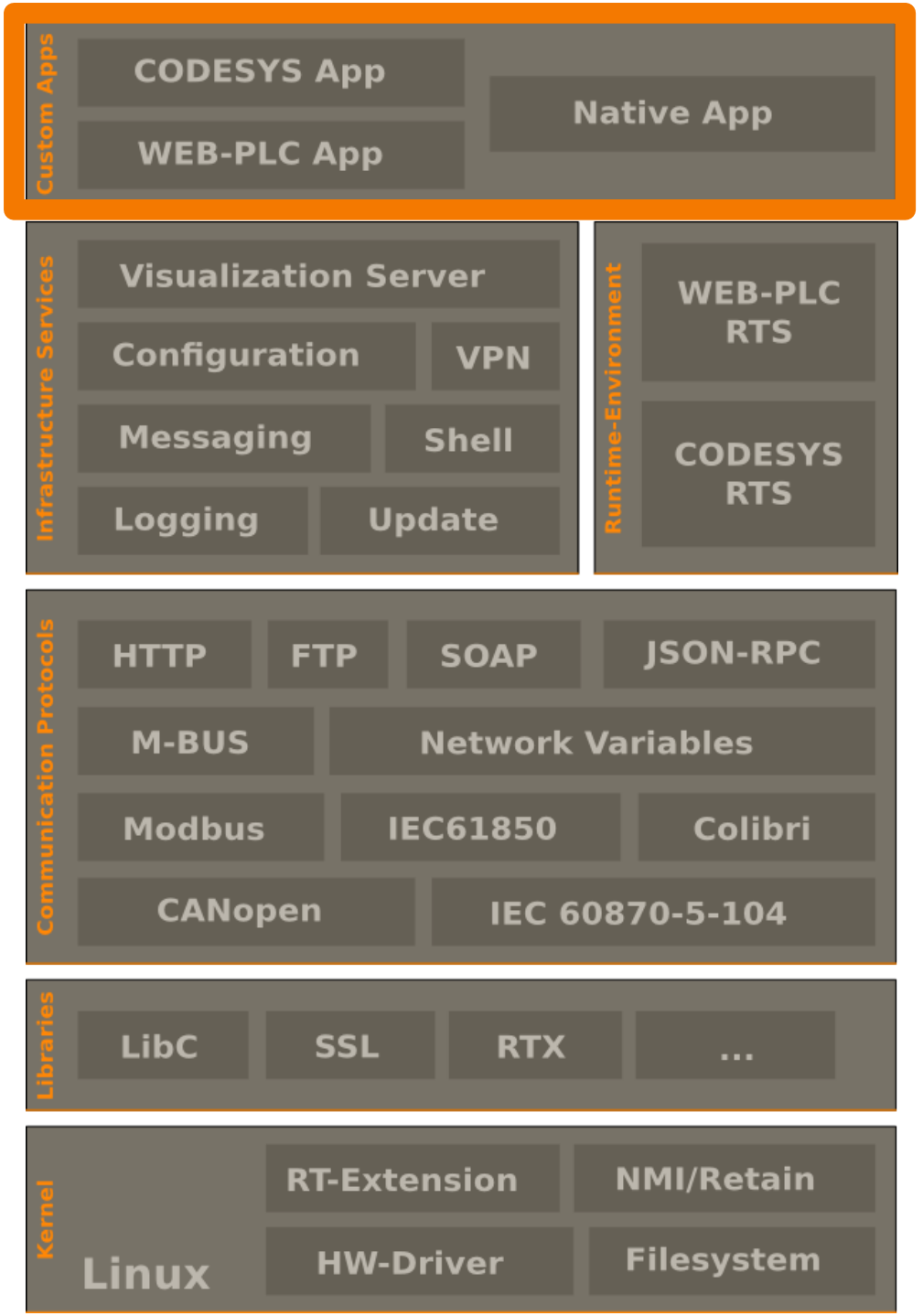
# Runtime-Environment

- IEC61131-3 Programmiersystem ***CODESYS*** (marktführer in Deutschland) mit typische SPS-Sprachen, wie FBD, ST, SFC, IL, ...
- Einfaches Browser-basiertes Programmier-system ***WEB-PLC*** für kleine Aufgaben.

# Motivation

## Wichtige SPS-Eigenschaften:

- ✓ Harte Echtzeit
- ✓ Feldbus-Protokolle
- ✓ Alarm-Management
- ✓ Logging-System
- ✓ Live-Daten (Diagnose, Wartung)
- ✓ Fernzugriff
- ✓ Update-Konzept (lokal und entfernt)
- ✓ GUI (für Browser und Display)
- ✓ SPS-typische Entwicklungsumgebung


Custom Apps
CODESYS App
WEB-PLC App
Native App
Infrastructure Services
Visualization Server
Configuration
VPN
Messaging
Shell
Logging
Update
Runtime-Environment
WEB-PLC RTS
CODESYS RTS
Communication Protocols
HTTP
FTP
SOAP
JSON-RPC
M-BUS
Network Variables
Modbus
IEC61850
Colibri
CANopen
IEC 60870-5-104
Libraries
LibC
SSL
RTX
...
Kernel
Linux
RT-Extension
NMI/Retain
HW-Driver
Filesystem

# Fazit

- Offenes System mit typische SPS-Eigenschaften
- Starker Fokus auf Kommunikation & Gateway
- Schnelle Updates dank Mainline-Kernel
- Linux als Basis, nicht aber als Frontend
- Spannend, denn wir müssen *das Letzte* aus der Hardware *rausholen*

# Vielen Dank!

# Fragen..?

Oder per Email: **c.stoidner@arvero.de**