
FRAUNHOFER-INSTITUT FÜR SOLARE ENERGIESYSTEME ISE

Architekturvorstellung und Performanceanalyse der offenen IEC 61850
Implementierung openIEC61850



Stefan Feuerhahn

Fraunhofer-Institut für
Solare Energiesysteme ISE

Leipzig, 25.9.2012

www.ise.fraunhofer.de

Stefan Feuerhahn

© Fraunhofer ISE

Agenda

- › IEC 61850: kurze Einführung
- › Entwicklung von openIEC61850 – Ziele und Design
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Performancevergleich der Implementierungen:
 - › openIEC61850 (in Java)
 - › Triangle MicroWorks Stack (in C)
- › Fazit

IEC 61850 Einführung am Beispiel eines BHKWs

Netzleitwarte /
Virtuelles Kraftwerk /
BHKW Besitzer

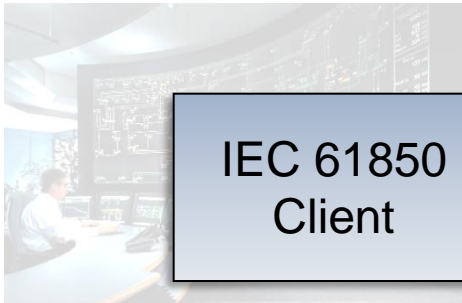


Blockheizkraftwerk
(BHKW)



IEC 61850 Einführung – Client-Server-Modell

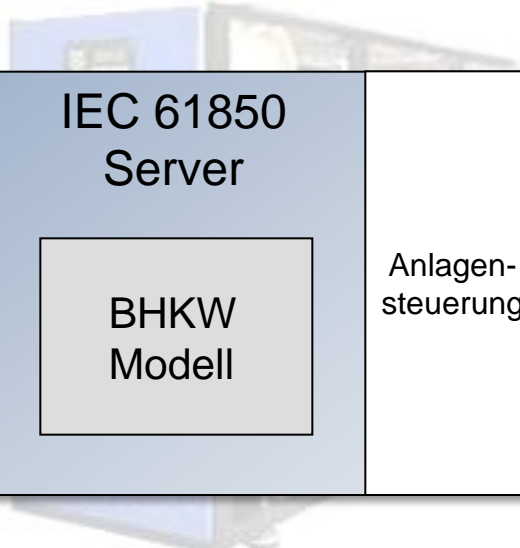
Netzleitwarte /
Virtuelles Kraftwerk /
BHKW Besitzer



IEC 61850
Client



Blockheizkraftwerk
(BHKW)



IEC 61850
Server

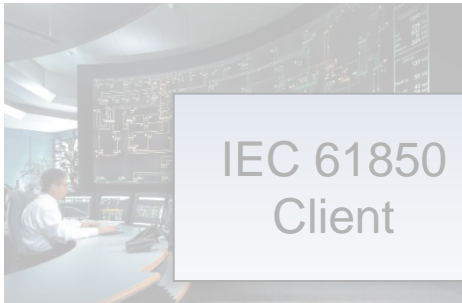
BHKW
Modell

Anlagen-
steuerung

IEC 61850 Einführung – Hauptteile des Standards


- › IEC 61850 definiert:
 1. Modellkomponenten (LN,DO,DA) aus denen das BHKW-Modell zusammengestellt werden kann
 2. Eine Sprache um die Modelle zu beschreiben (SCL/XML)
 3. Abstrakte Services über die der Client auf das Modell im Server zugreifen kann
 4. Konkrete Mappings auf Protokolle, die die Services realisieren

Netzleitwarte /
Virtuelles Kraftwerk /
BHKW Besitzer



IEC 61850
Client

Blockheizkraftwerk
(BHKW)

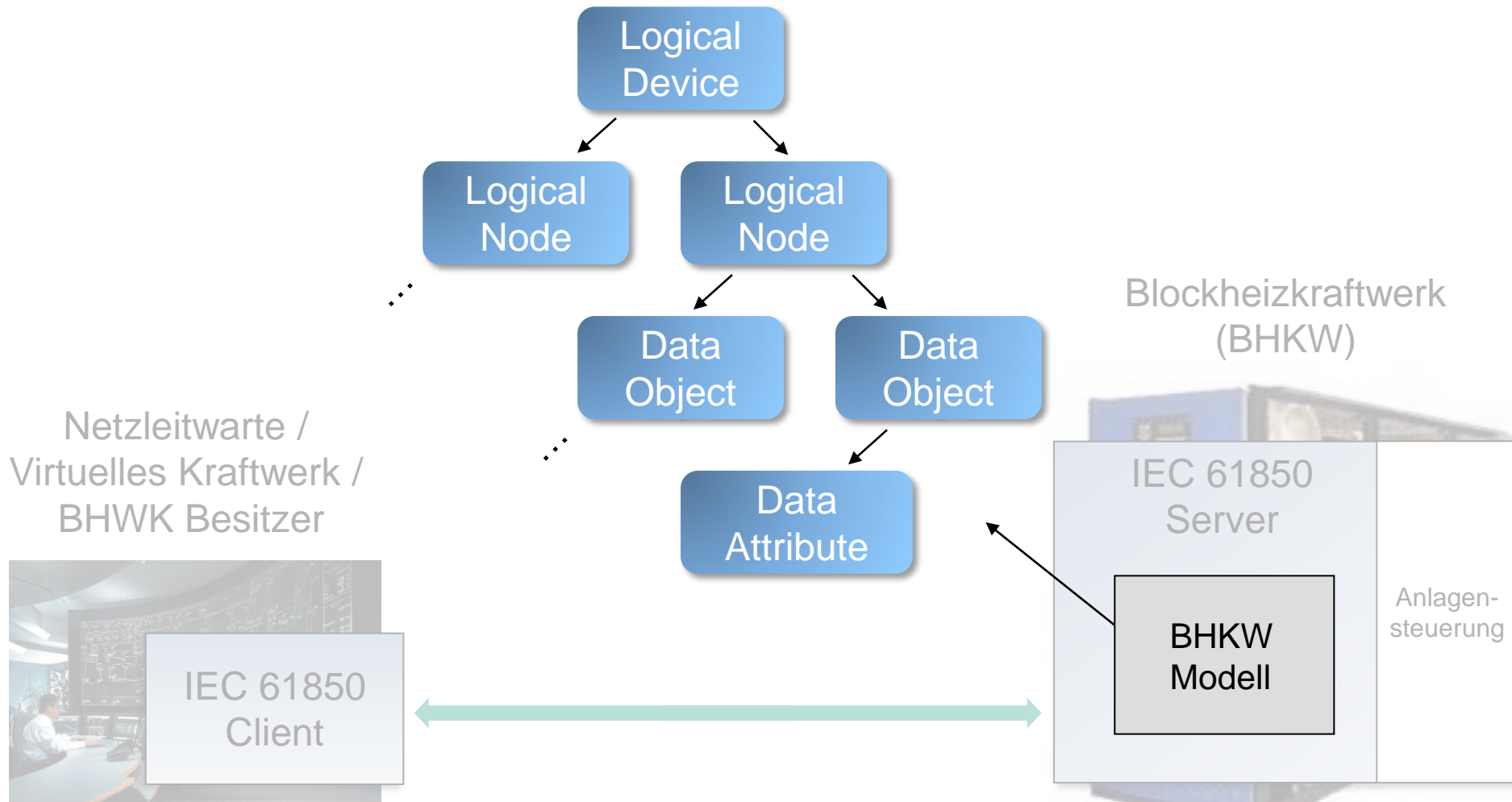


IEC 61850
Server

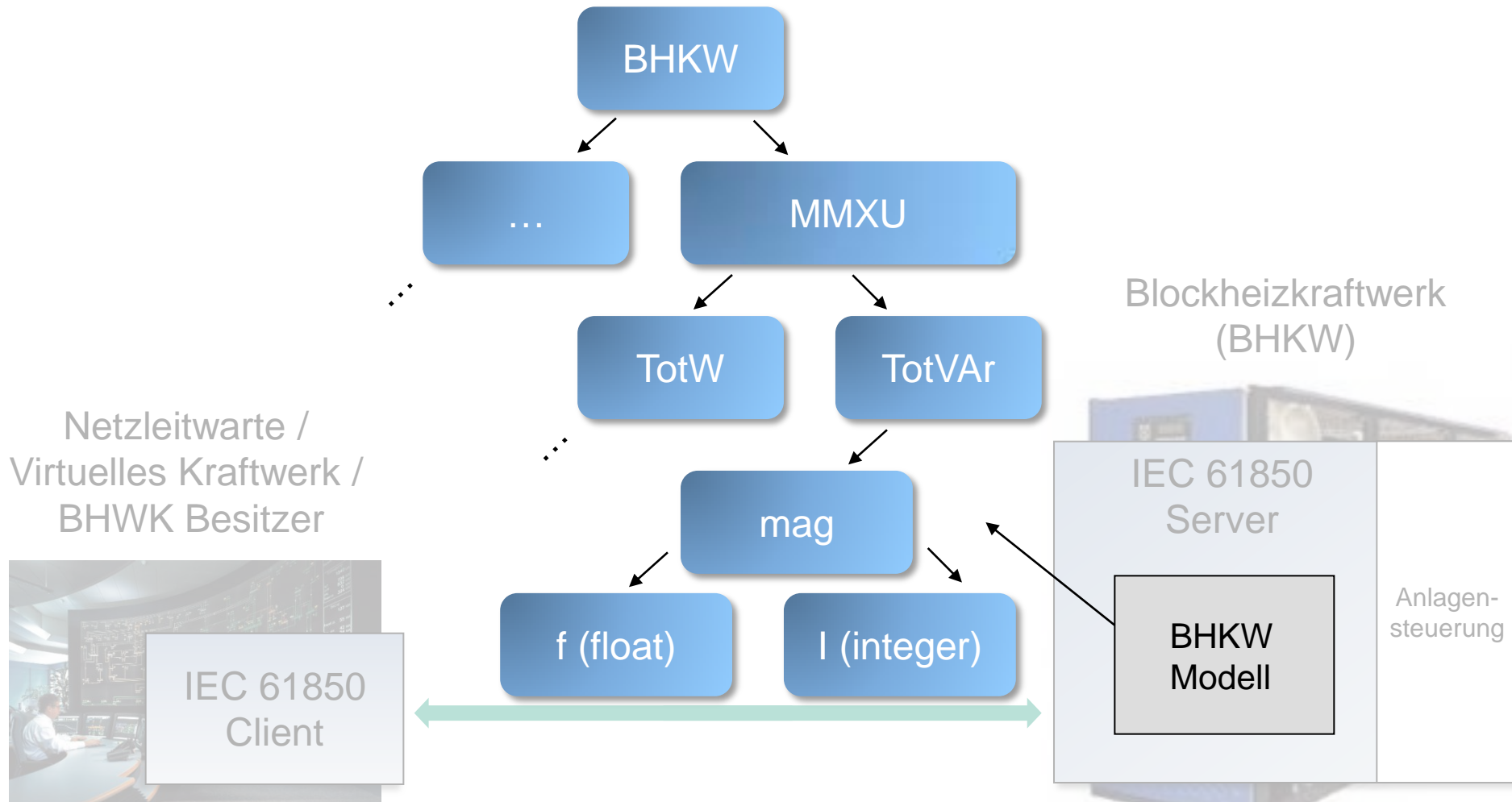
BHKW
Modell

Anlagen-
steuerung

IEC 61850 Einführung – Struktur des Datenmodells

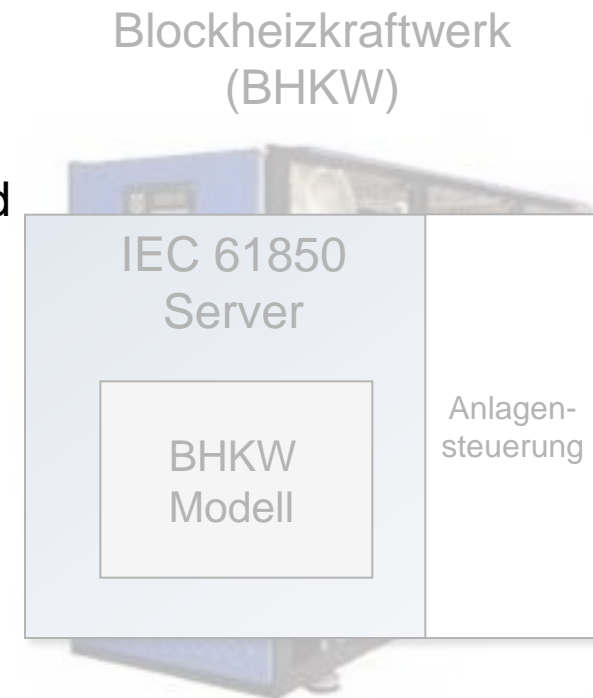


IEC 61850 Einführung – Modellbeispiel



IEC 61850 Einführung – Abstrakte Services

- › Abstrakte Services die ein IEC 61850 Server anbieten kann:
 - › Verbindung mit dem Server
 - › Abruf des Datenmodells
 - › Auslesen und Setzen von Mess-, Status- und Steuerwerten
 - › Gruppieren von Daten zum schnelleren Auslesen und Setzen
 - › Reporting und Konfiguration von Reports
 - › Logging und Konfiguration von Logs
 - › Datei-Übertragung
 - › Zeitsynchronisation
 - › Echtzeitservices (GOOSE)



IEC 61850 Einführung – Konkrete Mappings

- › Ein Mapping definiert wie die abstrakten IEC 61850 Services mittels eines konkreten Protokolls realisiert werden können
- › De facto wird das MMS-Mapping (IEC 61850-8-1) fast immer verwendet
 - › In bisherigen Umspannwerken etc.
- › Aber es gibt weitere Mappings:
 - › z.B. SOAP (IEC 61400-25-4)
- › Und zahlreiche Vorschläge für neue Mappings
 - › OPC-UA, RESTful Services etc.

IEC 61850 Einführung – Schwachstellen des Standards

- › Unnötige Komplexität des Standards
 - › viele anwendungsspezifische Details sind Teil des Standards
 - › Schlechte Beschreibung des Standards
 - › Viele ungenaue Aussagen
 - › Keine einheitliche Terminologie
 - › Erstellung von neuen IEC 61850 Modellen ist unflexibel
 - › Hierarchie: LD->LN->DO[->DO]->DA[->DA] muss eingehalten werden
 - › Es werden keine einheitlichen Modelle forciert
- › Aber IEC 61850 ist bereits ein Standard und in Verwendung
 - › -> Gute Chancen sich als Standard für Steuerung und Monitoring von dezentralen Erzeugern durchzusetzen

Agenda

- › IEC 61850: kurze Einführung
- › Entwicklung von openIEC61850 – Ziele und Design
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Performancevergleich der Implementierungen:
 - › openIEC61850 (in Java)
 - › Triangle MicroWorks Stack (in C)
- › Fazit

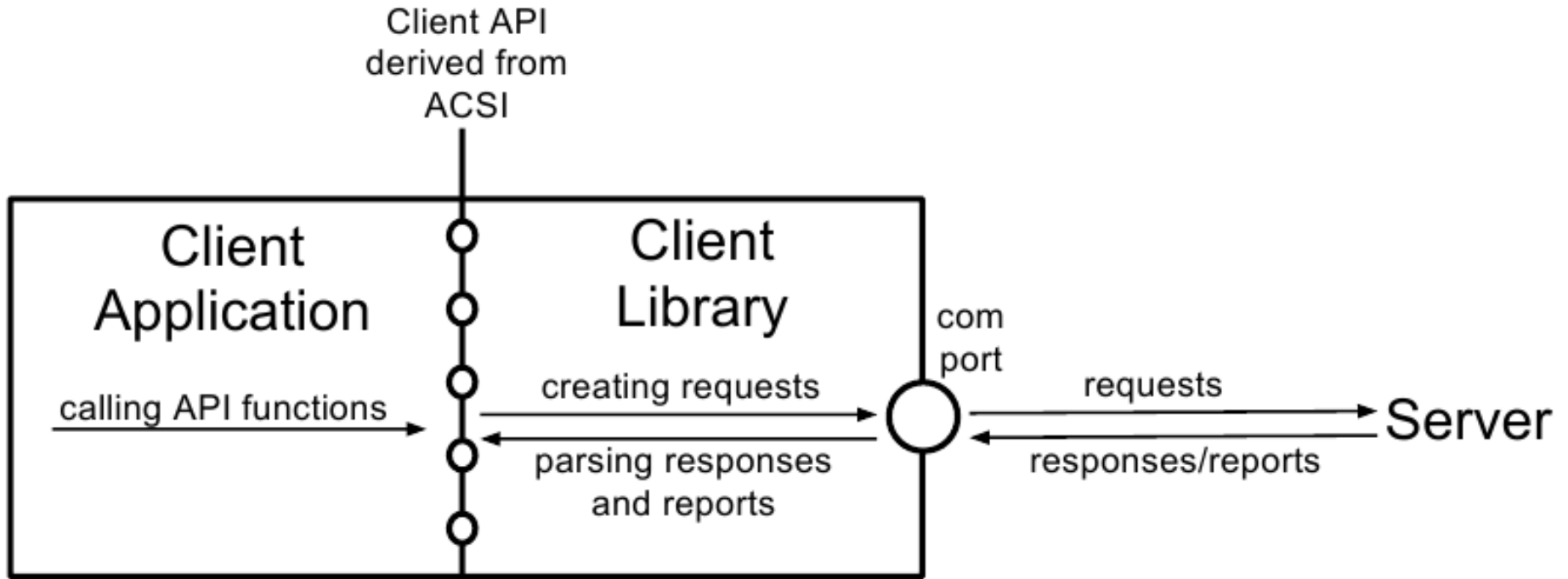
openIEC61850

- › openIEC61850 – IEC 61850 MMS Stack in Java
- › Start der Entwicklung im Rahmen des eTelligence-Projekts
- › Erste Veröffentlichung erfolgte Januar 2012 auf openmuc.org
- › Ziele:
 - › Alternative zu den sehr teuren kommerziellen Lösungen schaffen
 - › Mittels der offenen Implementierung wissenschaftliche Fragestellungen beantworten

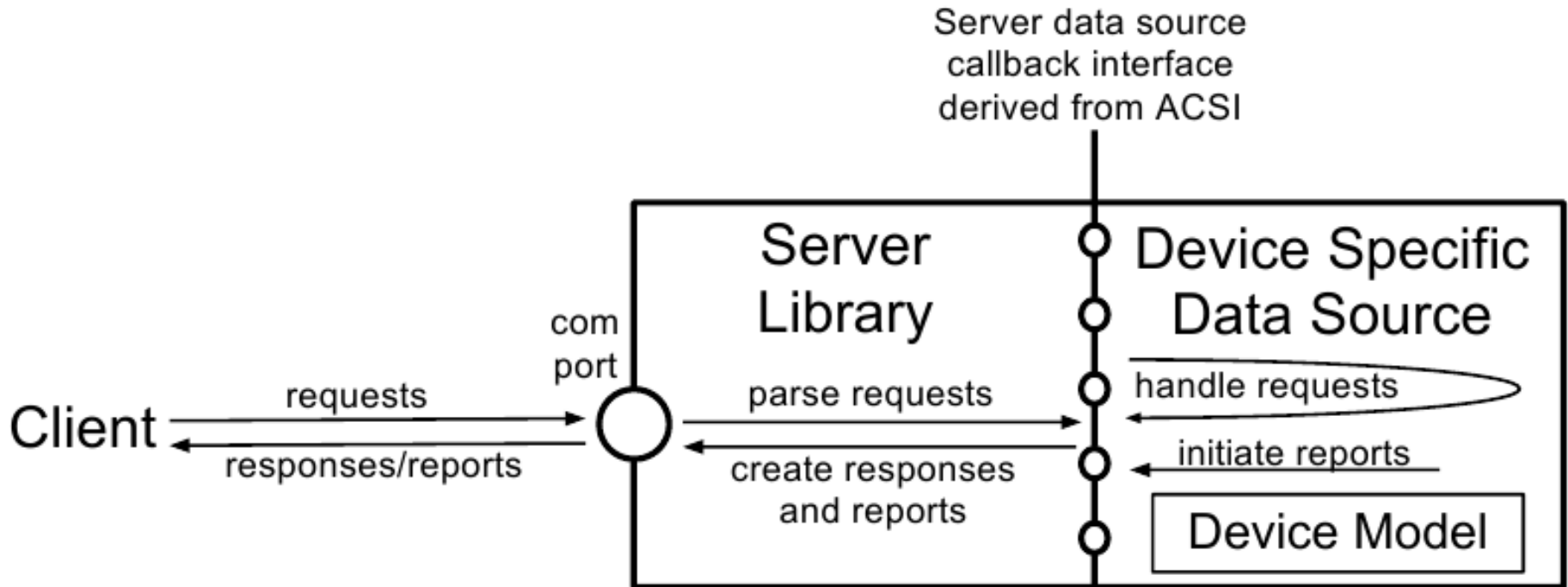
IEC 61850 ~~Ä~~ openIEC61850 - Fragestellungen

- › Welche Schwachstellen hat der Standard -> Verbesserungsvorschläge
- › Wie könnten herstellerübergreifende Modelle/Profile für dezentrale Erzeuger aussehen
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Vergleich der Implementierungen:
 - › openIEC61850
 - › Triangle MicroWorks (TMW)

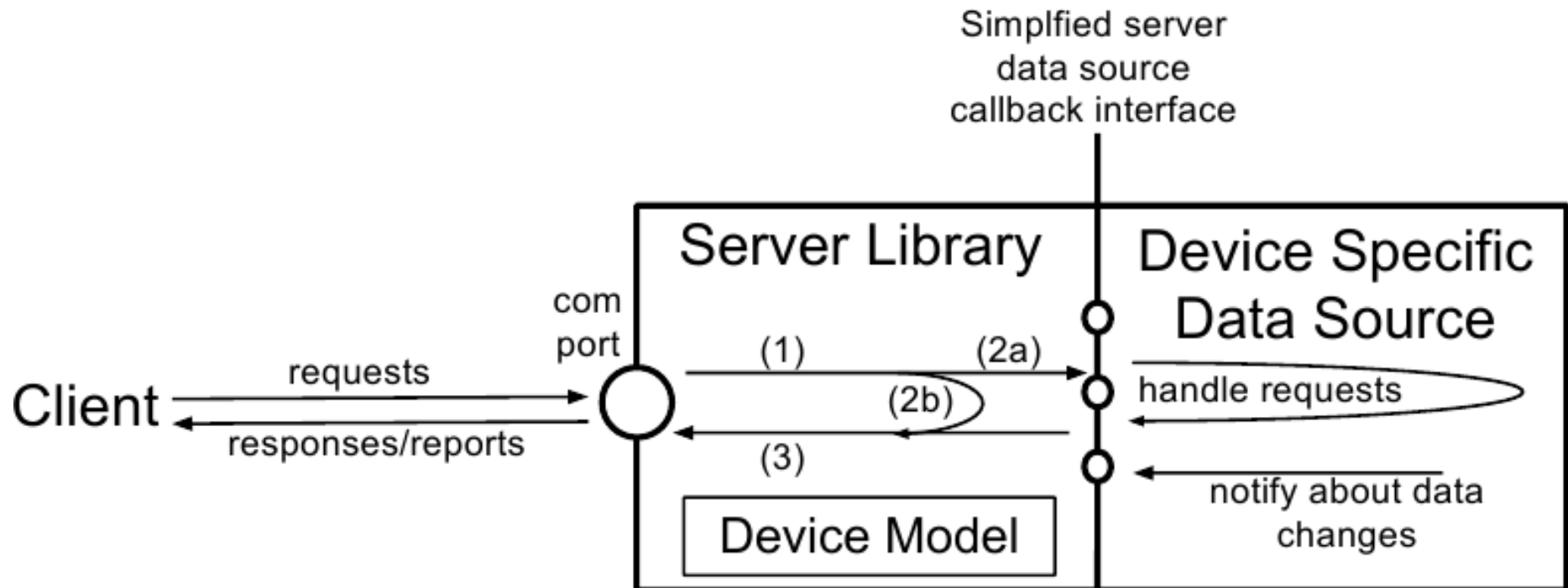
openIEC61850 – Design des Clients



openIEC61850 – Design des Servers – Eine Möglichkeit

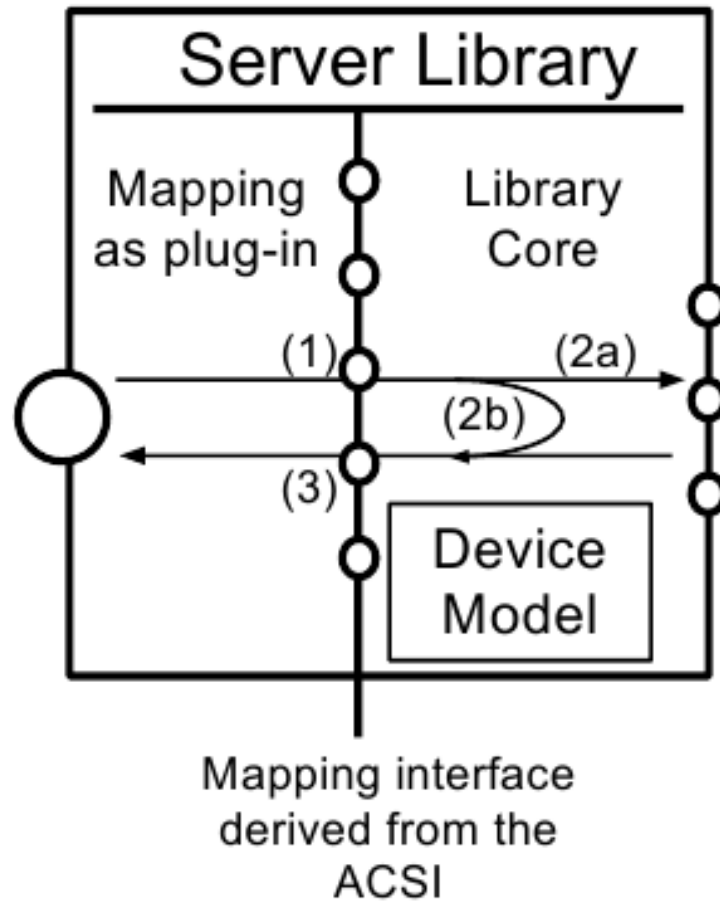


openIEC61850 – Design des Servers – Bessere Lösung



- › 1.) Decode Request
- › 2a) Leite vereinfachte Anfrage an Data Source weiter
- › 2b) oder beantworte Request wenn möglich selber
- › 3) encode Response

openIEC61850 – Design des Servers – Mapping as Plugin

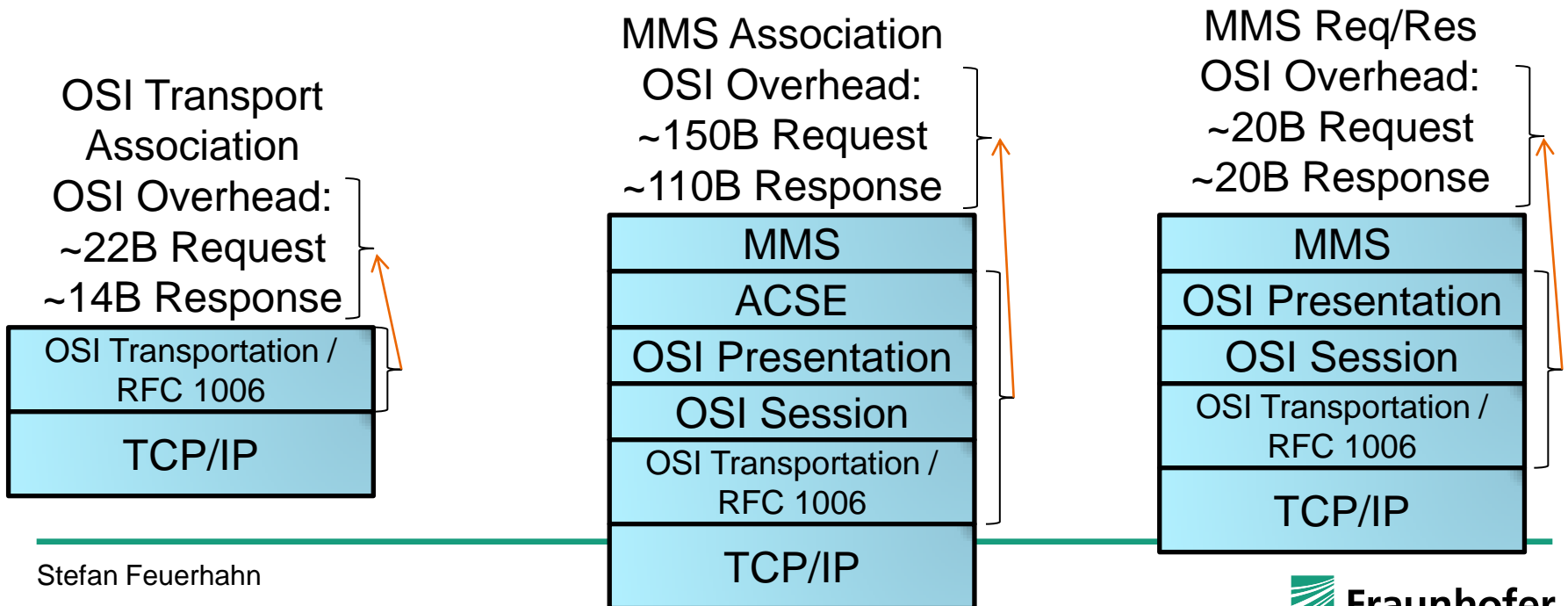


Agenda

- › IEC 61850: kurze Einführung
- › Entwicklung von openIEC61850 – Ziele und Design
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Performancevergleich der Implementierungen:
 - › openIEC61850 (in Java)
 - › Triangle MicroWorks Stack (in C)
- › Fazit

IEC 61850 Æ MMS-Mapping

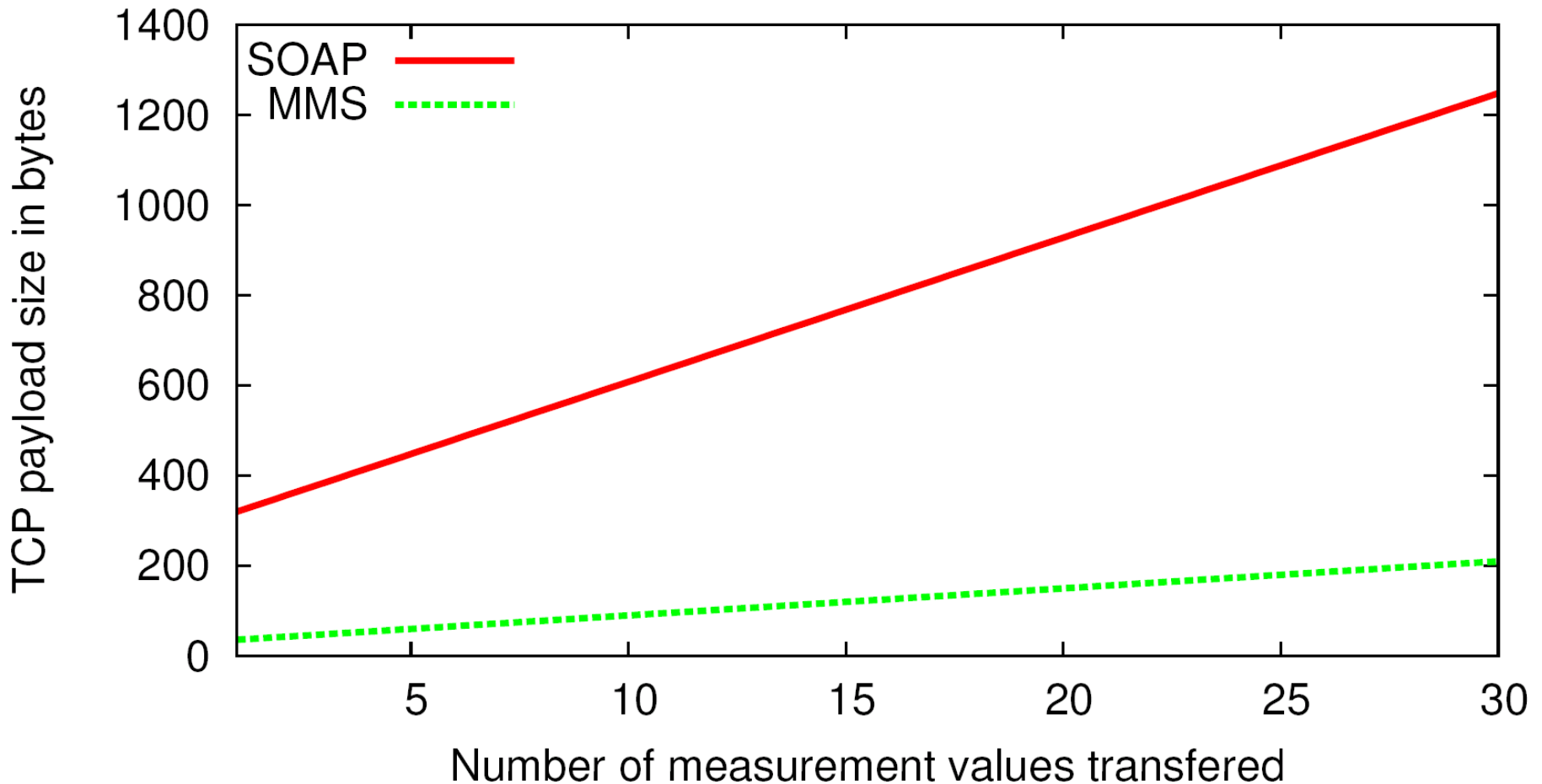
- › MMS – Manufacturing Message Specification (ISO 9506)
 - › Application Layer Protokoll für die Automatisierungstechnik aus den 80ern
 - › MMS Paket-Aufbau wird in ASN.1 definiert. Theoretisch wird kein konkretes Encoding vorgeschrieben, aber de facto wird immer BER eingesetzt
 - › Setzt aus historischen Gründen auf dem ISO/OSI Schichtenmodell auf (eigentlich unnötig)



IEC 61850 Æ SOAP-Mapping

- › SOAP-Mapping (IEC 61400-25-4)
 - › Teil von IEC 61400: Communications for monitoring and control of wind power plants
 - › Noch jung im Vergleich zu MMS
 - › Wird in diversen Forschungsprojekten eingesetzt

MMS & SOAP Ȧ Vergleich PaketgröÙe



IEC 61850 – Vergleich MMS und SOAP Mapping

- › Vorteile des MMS-Mappings:
 - › Deutlich effizientere Kodierung
 - › Unterstützt Reporting über permanente TCP/IP Verbindung
 - › Wird bereits in großem Maßstab verwendet
- › Vorteile des SOAP-Mappings:
 - › Association-Aufbau benötigt nur einen Handshake (nicht 2 wie MMS)
 - › Keine permanente TCP/IP Verbindung nötig
 - › Deutlich einfacher zu implementieren (XML Parser sind bereits Teil der Java API)



openIEC61850 implementiert MMS

Agenda

- › IEC 61850: kurze Einführung
- › Entwicklung von openIEC61850 – Ziele und Design
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Performancevergleich der Implementierungen:
 - › openIEC61850 (in Java)
 - › Triangle MicroWorks Stack (in C)
- › Fazit

IEC 61850 \ddot{E} ASN.1 und BER

- › Wichtigste Komponente des MMS-Mappings:
 - › ASN.1-BER Encoding/Decoding der MMS-PDUs
- › Der Aufbau der MMS-Pakete ist in Abstract Syntax Notation One (ASN.1) beschrieben, diese werden mittels der Basic Encoding Rules kodiert/dekodiert
- › ASN.1/BER
 - › Erste Version von 1984
 - › Wird u.a. für digitale Zertifikate verwendet (z.B. X.509)

IEC 61850 Æ ASN.1 und BER

- › Für openIEC61850 wurde benötigt
 - › ASN.1 Tool zur Erzeugung von Java Klassen aus dem ASN.1 Syntax
 - › Java Klassen können anschließend zum einfachen Kodieren und Dekodieren der BER Datenströme verwendet werden
 - › Möglichst unter OpenSource-Lizenz (LGPL oder freier)

IEC 61850 – Vergleich von Java ASN.1/BER Bibliotheken

ASN.1/BER Bibliotheken	Beinhaltet ASN.1 Compiler zur Erzeugung von Java-Klassen	Versteht MMS ASN.1 Syntax	Lizenz
BouncyCastle	nein	n.a.	Permissive Open Source
CoDec	nein	n.a.	LGPL
JAC	ja	ja	GPL
BinaryNotes	ja	ja nach Modifikation	Apache

- › Zunächst wurde BinaryNotes für openIEC61850 verwendet
- › Schlechte Performance -> BinaryNotes wurde ersetzt durch eigene Implementierung: „jASN1“

ASN.1/BER Bibliotheken Æ Performance-Vergleich

- › Vergleich für der benötigten Zeit zum Kodieren und Dekodieren des Beispiels aus ITU-T X.690 (07/2002)
- › Parameter:
 - › Java HotSpot VM (Server Modus)
 - › Intel Core2 Duo, P8700 @ 2.53GHz
 - › 40.000 Runs, davon Durchschnittswert der Runs 30.001-40.000:

ASN.1/BER Bibliotheken	Zeit kodieren/ dekodieren in ns	Zeit kodieren/ dekodieren in % relativ zu jASN1
JAC	23753/11656	280%/252%
BinaryNotes	11464/17278	135%/374%
jASN1	8458/4613	100%/100%

IEC 61850 È jASN.1

- › Common Coding-Rules vs. Performance
- › Häufige Java Performance-Faktoren
 - › Function calls (Vererbung checken)
 - › Initialization of new Objects
- › Wie jASN.1 probiert diese Kosten zu minimieren:
 - › Kodier- und Dekodierintelligenz steckt in den erzeugten Klassen selber
 - › Die meisten ASN.1 Compiler erzeugen Klassen, die von externen Decodern / Encodern „geliefert“ werden
 - › Kodierung erfolgt „rückwärts“
 - › So können Sequenzelemente sofort in einen Stream geschrieben werden, ohne auf darauf zu warten, dass die Länge bekannt ist

jASN1

ASN.1 Beispiel:

```
Person ::= SEQUENCE {  
    name OCTET STRING,  
    age INTEGER  
}
```

jASN1 Compiler
generiert

Person.java

beinhaltet encode Funktion

BerInteger.java

BerOctetString.java



Schreibt rückwärts
in einen ByteArray

```
encode(BerOutputStream os)  
{  
    int length = 0;  
    length += age.encode(os);  
    length += name.encode(os);  
    os.write(length);  
    os.write(tag);  
}
```

openIEC61850 – Relevanz von ASN.1/BER Kodierung

› Wie relevant ist die Optimierung des BER Kodierers?

› Profiling des Servers

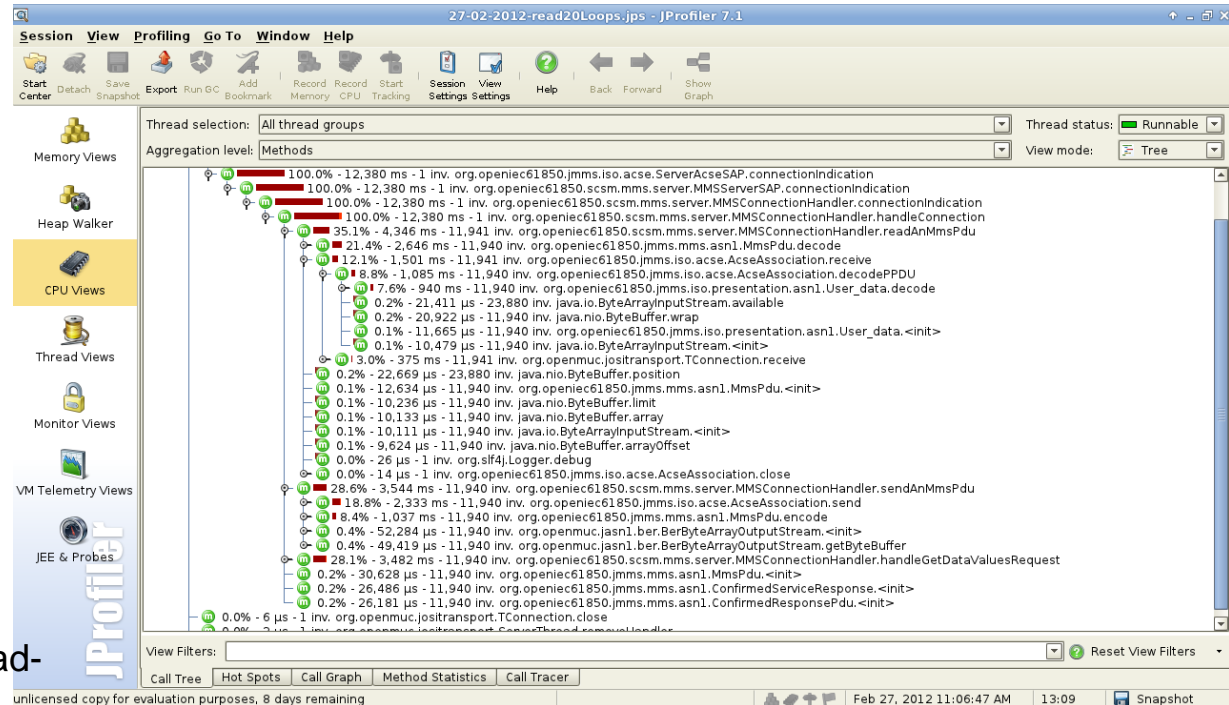
› JProfiler 7.1

› jASN1 v. 1.1.1

› HotSpot Java VM in Server Mode

› Etwa 12000 getDataValues Aufrufe für verschiedene BasicDataAttributes

› CPU: AMD Phenom 9600 Quad-Core



› Profiling zeigt, dass etwa 23,0% mit Dekodierung und 28% der Zeit mit Kodierung von ASN.1/BER Information verbracht wurde (Kodierung beinhaltet in diesem Fall nicht die Erzeugung der Java-Objekte die kodiert werden)

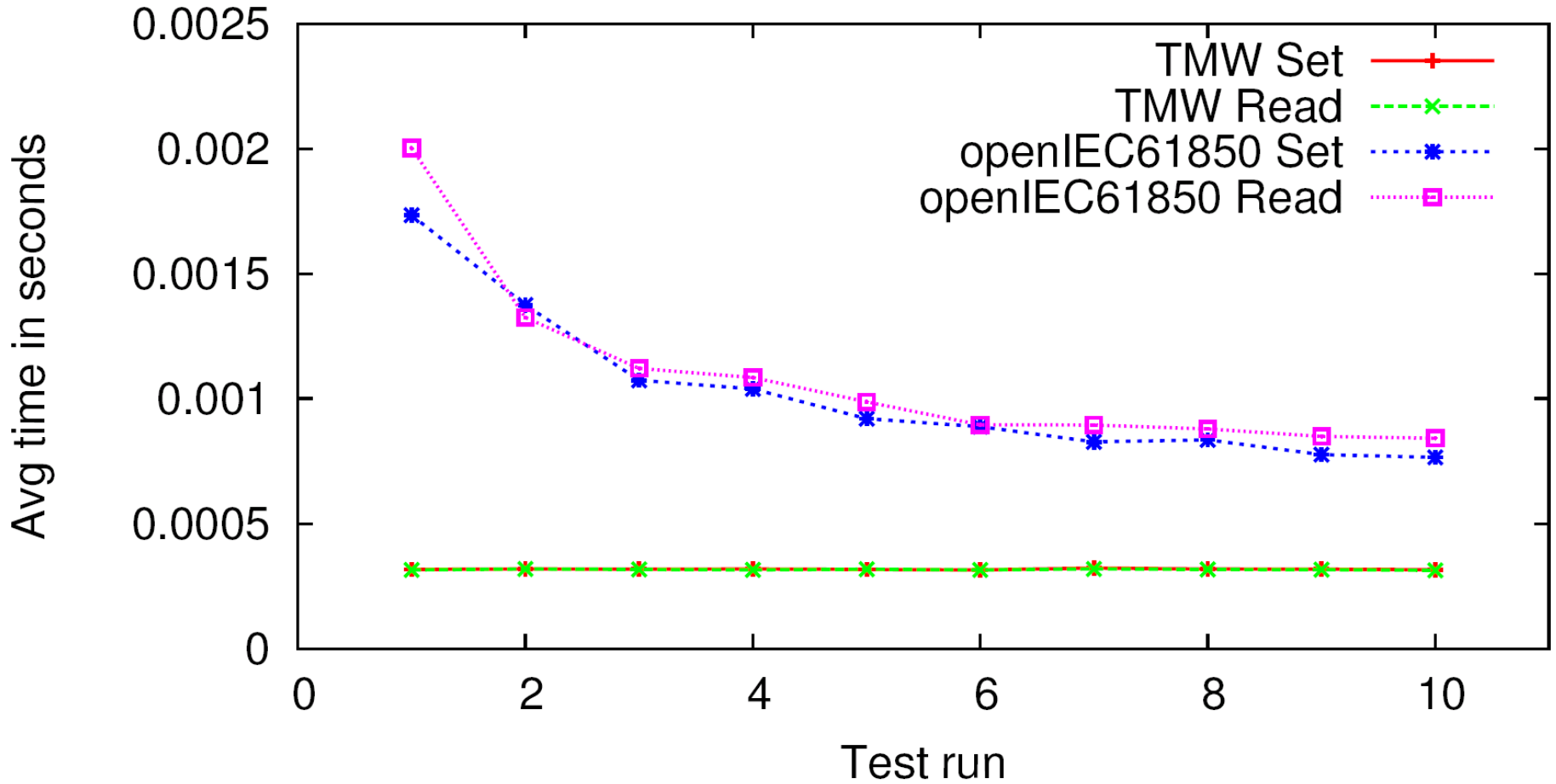
Agenda

- › IEC 61850: kurze Einführung
- › Entwicklung von openIEC61850 – Ziele und Design
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Performancevergleich der Implementierungen:
 - › openIEC61850 (in Java)
 - › Triangle MicroWorks Stack (in C)
- › Fazit

openIEC61850 ̈ Performance-Vergleich

- › Vergleich von openIEC61850 mit IEC 61850 Stack von Triangle MicroWorks
- › 10 Testläufe
- › Jeder Testlauf besteht aus 498 SetDataValues und GetDataValues service requests von verschiedenen BasicDataAttributes
- › Server nutzen beide identische Modelle und liefern identische Dummywerte zurück
- › Ermittlung der Reaktionszeit
 - › TcpDump läuft auf Servermaschine
 - › Reaktionszeit = Timestamp des eingehenden Requests – Timestamp der ausgehenden Response
- › Hot Spot VM (Server Modus) , Geode™

openIEC61850 Ę Performance-Vergleich



Agenda

- › IEC 61850: kurze Einführung
- › Entwicklung von openIEC61850 – Ziele
- › Analyse
 - › Vergleich von MMS und SOAP Mappings
 - › Performanceanalyse und Profiling der openIEC61850 MMS Implementierung
 - › Performancevergleich der Implementierungen:
 - › openIEC61850 (in Java)
 - › Triangle MicroWorks Stack (in C)
- › Fazit

Fazit

- › Der größte Nachteil von IEC 61850: unnötig komplex und schwer zu Implementieren
- › Größter Vorteil: schon jetzt ein international anerkannter Standard und wird bereits eingesetzt (vor allem in Umspannwerken)
- › MMS-Mapping ist deutlich effizienter bezüglich Paketgröße als SOAP-Mapping aber Implementierungshürde ist deutlich höher
- › Fokus der Implementierung auf ein Mapping ist weniger Aufwändig und effizienter
- › ASN.1/BER Kodierung ist der wichtigste Faktor bezüglich der Performance von IEC 61850
- › Mit jASN1 konnte eine deutliche Performancesteigerung im Vergleich zu anderen offenen ASN.1/BER Bibliotheken geschaffen werden
- › Nach Optimierung durch die HotSpot VM ist openIEC61850 immer noch etwa 2,5 langsamer als der proprietäre TMW-Stack
- › Viele Ideen für weitere Optimierungen existieren bereits
- › Weitere Ideen für Analysen: Speicherbedarf, Multi-Thread-Tests