EASY-RES

SUMMER SCHOOL
"ENABLING DRES TO OFFER ANCILLARY SERVICES"
20TH – 24TH SEPTEMBER 2021

ICT Architecture for Enabling Ancillary Services

Prof. Hermann de Meer, Armin Stocker // 23.09.2021

This project has received funding from the European Union´s Horizon 2020 Programme for research and innovation under Grant Agreement no 764090.

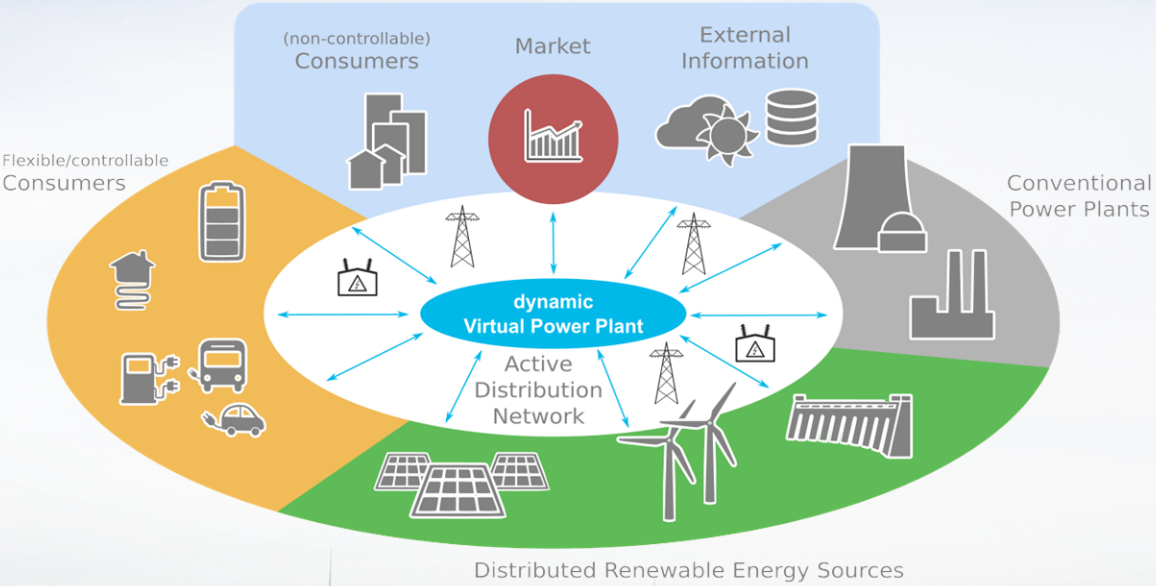
Agenda

- Motivation
- Distributed Systems
- Virtualization
- EASY-RES ICT Architecture

Agenda

- Motivation
- Distributed Systems
- Virtualization
- EASY-RES ICT Architecture

What we want to achieve!



Virtual Power Plant

- Grid Monitoring
- Resource Management
- Grid Protection
- Grid Control/Optimization
- Ancillary services
  (as a virtual synchronous generator)

How to build such a system?

Motivation

Systems

System

- “a **combination of interacting elements** organized to **achieve one more stated purpose**” [ISO/IEC/IEEE 15288]

Software-intensive systems

- …are systems in which **software development and/or integration are dominant considerations** (i.e., most complex systems these days). This includes computer-based systems ranging from individual software applications, information systems, embedded systems, software product lines and product families and systems-of-systems. [ISO/IEC/IEEE 42010]

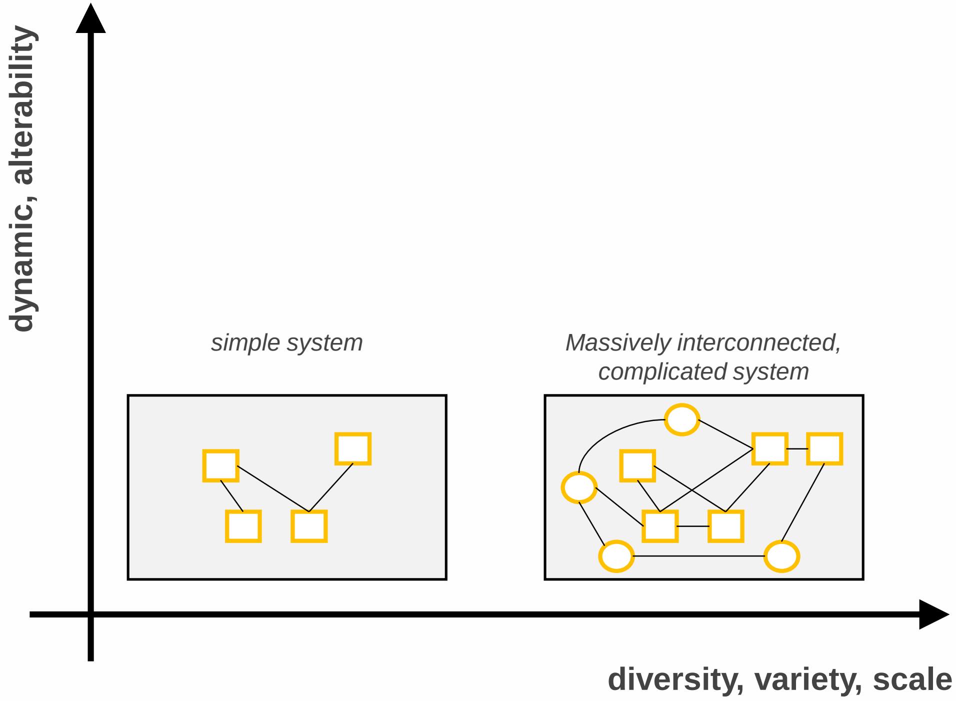
Aspects of systems

- Man-made
- Created and utilized to provide services in defined environments for the benefit of users and other stakeholders
- Configured with one or more of the following:
  - hardware
  - software
  - humans
  - processes (e.g., review process)
  - procedures (e.g., operator instructions)
  - facilities
  - naturally occurring entities

ISO/IEC/IEEE 15288 2015; ISO/IEC/IEEE 42010

Motivation

Classification of Systems



Simple system: comprises only a few **statically** interconnected elements with a low intensity of interaction. They can be described in their entirety explicitly by means of mathematical and/or analytical methods.

Example: Q(U) voltage controller at a DRES

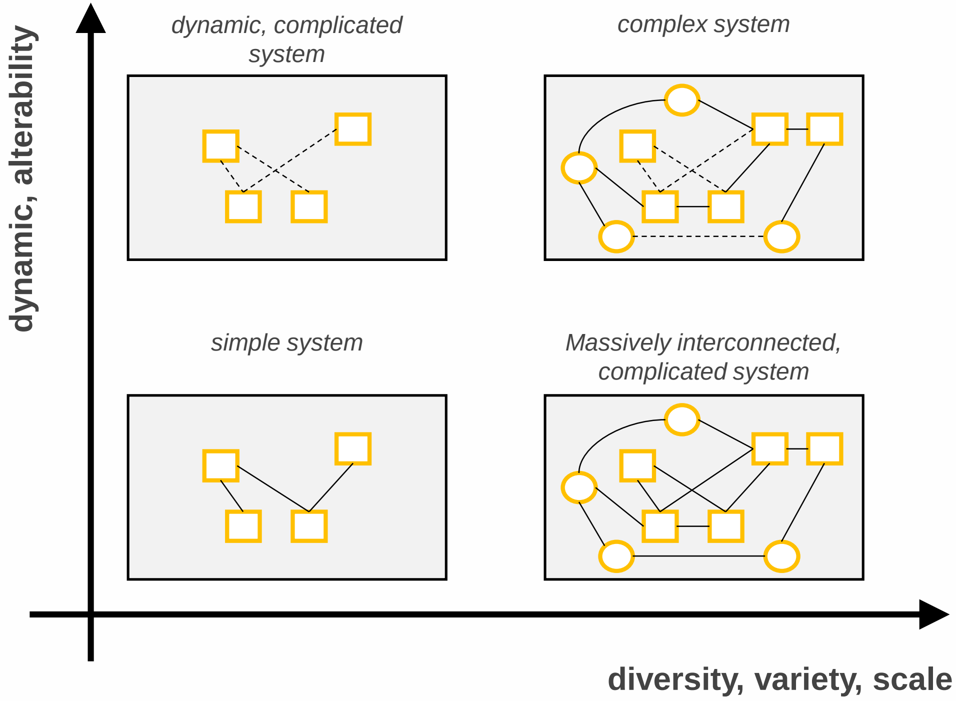
Massively interconnected, complicated system: Characterized by huge number of **statically** connected elements and their variety. Explicit description is hard to obtain. Description of system‘s behavior mostly only by computer simulations.

Example: Interconnected power system

[Haberfellner 2019]

Motivation

Classification of Systems



[Haberfellner 2019]

Dynamic, complicated system: characterized by their **temporal**, often **non-linear** variety between elements in respect to their type of interaction, intensity and structure. Amount and intensity of inter-dependencies is limited.

Example: Different AS at a DRES using different components

Complex system: characterized by a **huge number** of elements and connections with a high variety and **dynamically established connections**. System-wide interdependencies between elements exist.

Example: Dynamic virtual power plant coordinating different AS at different locations

Motivation

System-of-Systems (SoS)

Definition

- "Interoperating collection of component systems that produce results unachievable by the individual systems alone" [Strickland 2011]

System of systems

- Combination of power system and ICT system
- Massively interconnected
- Dynamically interacting effects

Two fundamental paradigms

- **Abstraction** („Divide and Conquer“)
  - Hide details on higher levels
  - Treat on lower levels

  → Decide on abstraction levels and granularity
- **Separation of Concerns**
  - Deal with different aspects in different views

  → Decide on views

Motivation

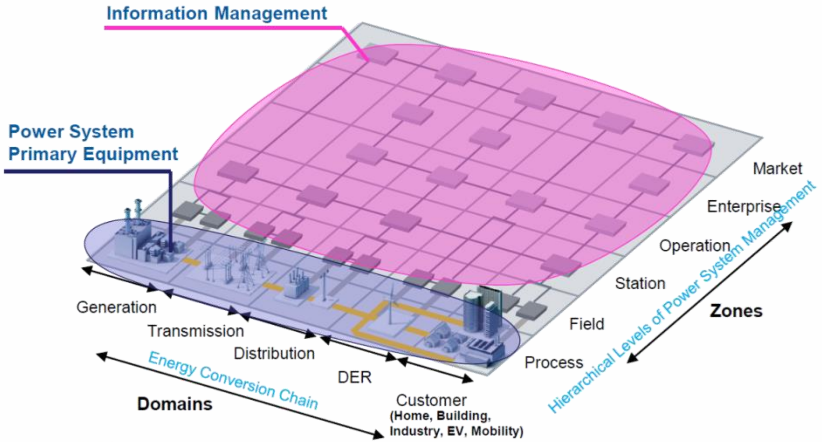
Smart Grid Architecture Model (SGAM)

SGAM-Plane: problem decomposition

- Domains: Energy conversion chain
- Zones: Automation pyramid

Examples

- Smart meter: Customer, Field
- Grid control center: Distribution, Operation
- dVPP: Distribution/Transmission, Operation



CEN-CENELEC-ETSI 2012

Motivation

Smart Grid Architecture Model (SGAM)

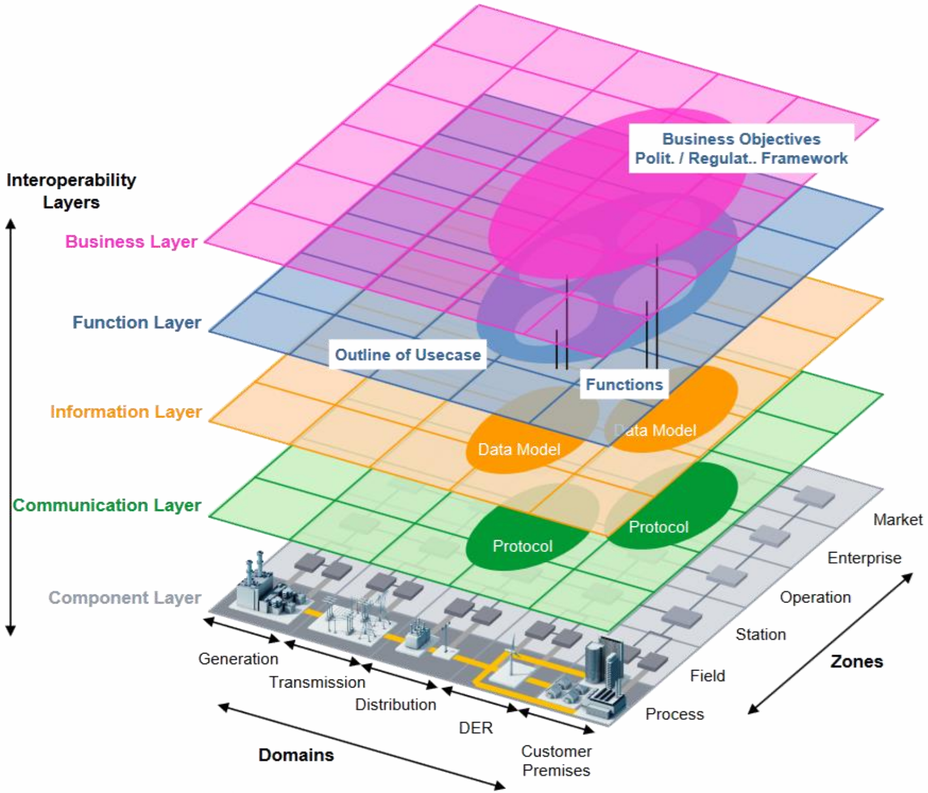
SGAM-Plane: problem decomposition

- Domains: Energy conversion chain
- Zones: Automation pyramid

Interoperability on different layers („SGAM Layers“)

- Business, *e.g. use cases*
- Function, *e.g. algorithms*
- Information, *e.g. data representation*
- Communication, *e.g. encryption protocols*
- Component

Reuse of standards, e.g. data models, protocols, etc.



CEN-CENELEC-ETSI 2012

Agenda

- Motivation
- Distributed Systems
- Virtualization
- EASY-RES ICT Architecture

Distributed Systems

Definition

- Before 1985: computers were large, expensive, independent
- Starting in the mid-1980s:
  - "A Distributed System is a **collection of autonomous and cooperative computers** that appears to users as a **single coherent system**" [vanSteen 2017]

    reads,
  - Invention of high speed computer networks

    "A dVPP is a **collection of autonomous and cooperative CI-DRES and ICT units** that virtually appears to AS customers as a **single coherent synchronous generator**"

    nected
- The... of networked and geographically dispersed computers → called Distributed Systems

Distributed Systems

Characteristics

dVPP

- **Collection of autonomous computing elements** → **Collection of autonomous CI-DRES and ICT units**
  - Different types of nodes:
    - Hardware device and software process → E.g. CI-DRES and central optimization algorithms
    - Act independently
    - Programmed to achieve common goals
  - Management of group membership (e.g., admission control) → ICT and related quality of service
  - Organized as overlay networks → E.g. Overlay network with increasing voltage level for voltage control optimization

Distributed Systems

Characteristics

dVPP

- **Single coherent system**
  - Coherent:
    - System behaves according to the expectations → E.g. fault current is provided according to the configurations
  - Distribution transparency
    - Location of data storage is not concerned → Exact location of inertia provisioning is not concerned
    - Location of the web service might be relevant to performance → Voltage control is location-dependent
  - Main challenge: failure masking
    - Partial failures of system components are unavoidable
    - In case of failure:
      - Some applications continue. For example, an ICT replica takes over → E.g. shifting inertia response to anther CI-DRES in the same area.
      - Some applications halt → E.g. lack of resources for voltage control

Design Goals

dVPP

- **Accessibility and Resource Sharing**
  - Makes it easy for users to access and share remote resources → Using uniform AS & CI-DRES interfaces
  - Resources can be virtually anything, e.g., storage facilities, data, files, services, and networks. → E.g. batteries, and CI-DRES
  - Reasons for resource sharing: Economics, collaboration and information exchange → E.g. optimal inertia allocation, ramp-rate limitation
  - Security and privacy issues → E.g. secure accounting transactions, integrity and confidentiality of AS monitoring and control
- Transparency
- Openness
- Scalability

Distributed Systems

Design Goals

dVPP

- Accessibility and Resource Sharing
- **Transparency:**
  - **Access**: Hides differences in data representation and how a resource is accessed → Using uniform AS & CI DRES interfaces
  - **Location**: Hide where a resource is located → E.g. hiding where inertia is provided
  - **Migration**: Hide that a resource may move to another location → E.g. shifting inertia response to another CI-DRES in the same area
  - **Replication**: Hide that a resource is replicated → E.g. replicated batteries for ramp-rate limitation
  - **Concurrency**: Hide that a resource may be shared by several competitive users → E.g. CI-DRES provides multiple AS
  - **Failure**: Hide the failure and recovery of a resource → E.g. shifting inertia response to another CI-DRES in the same area
- Openness
- Scalability

Distributed Systems

Design Goals

dVPP

- Accessibility and Resource Sharing
- Transparency
- **Openness**: Components follow standard rules for syntax and semantics of the services provided
  - **Interoperability**: The possibility for two different implementations to cooperate
  - **Portability**: The degree to which an application for a distributed system A can be run on another distributed system B, which has implemented the same interfaces
  - **Extensibility**
    - Easy configuration and modifications
    - Easy integration or replacement of components
    - Defining internal interfaces is crucial
- Scalability

→ Using uniform vendor-independent AS & CI-DRES interfaces

Distributed Systems

Design Goals

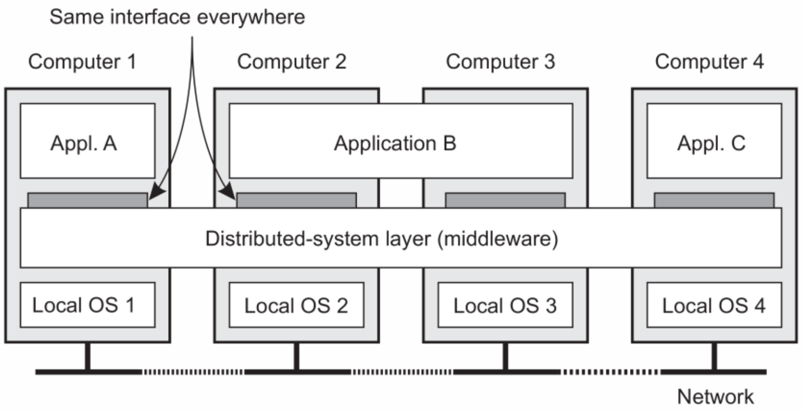
dVPP

- Accessibility and Resource Sharing
- Transparency
- Openness
- **Scalability**
  - **Size Scalability**: easily add more users and resources to the system without any noticeable loss of performance → E.g. adding more CI-DRES by several orders of magnitude
  - **Geographical Scalability**: users and resources may lie far apart, but the fact that communication delays may be significant is hardly noticed → E.g. system can scale to a whole synchronous zone
  - **Administrative Scalability**: easy management of a system even if it spans many independent administrative organizations → E.g. multiple system operators and aggregators

Distributed Systems

Middleware

- For distributed communication, often a middleware is used (abstraction layer)
- Provides resource management, communication, transaction management, service composition, security services, masking failures, ...
- It is logically placed between 2 layers:
  - Higher level layer for users and applications
  - Underlying layer for operation systems and communication facilities

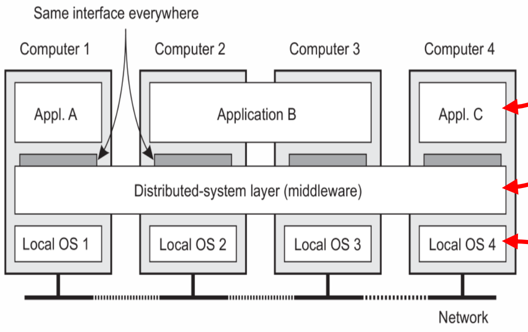


A distributed system organized in a middleware layer, which extends over multiple machines, offering each application the same interface

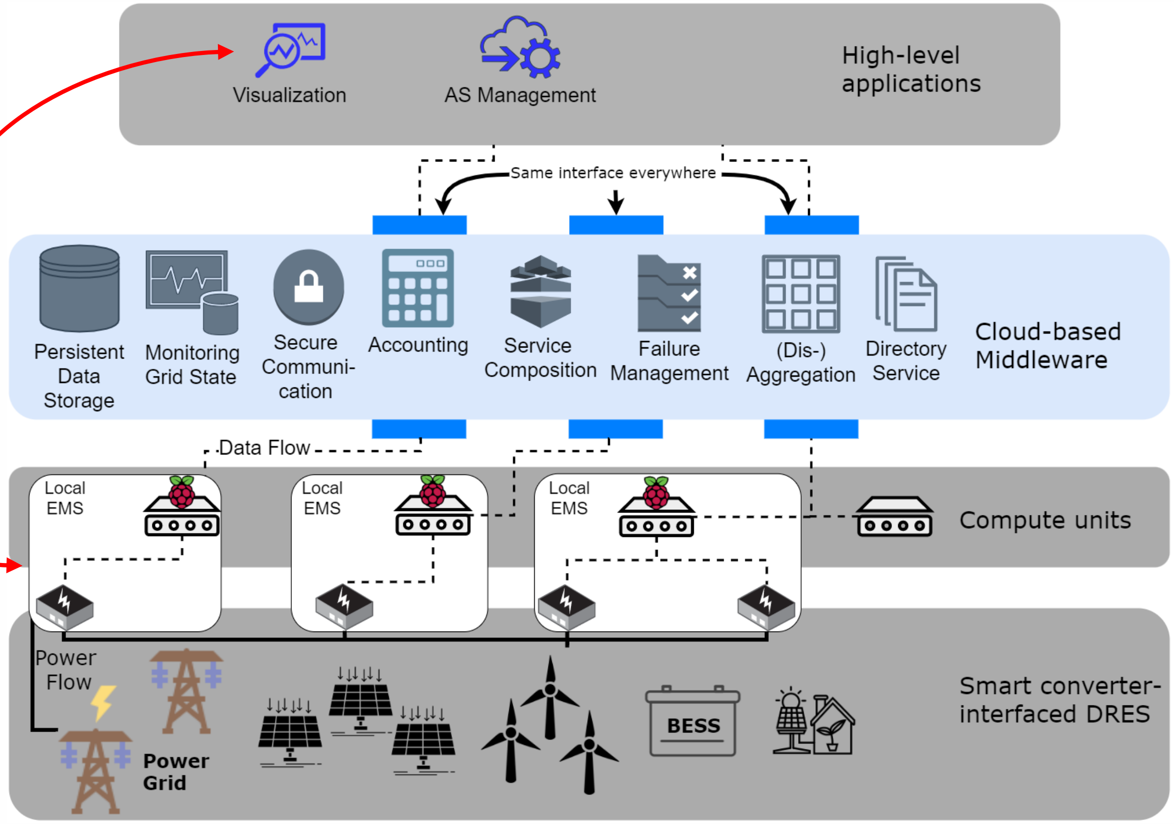
[vanSteen 2017]

Distributed Systems

Middleware



Dynamic Virtual Power Plant



Distributed Systems

Q&A

- The nodes of a distributed system are

A. Autonomous and non-cooperative

B. Autonomous and cooperative

C. Non-autonomous and cooperative

D. Non-autonomous and non-cooperative

Distributed Systems

Q&A

- Which transparency type does enable identical operation on local and remote resources?

A. Access transparency

B. Location transparency

C. Replication transparency

D. Concurrency transparency

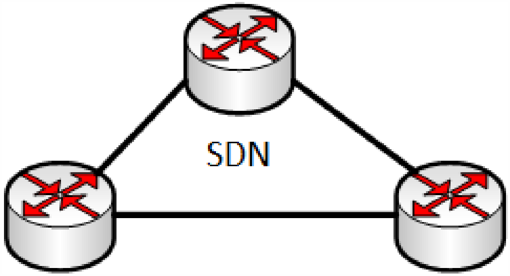
Agenda

- Motivation
- Distributed Systems
- Virtualization
- EASY-RES ICT Architecture

Virtualization

ICT virtualization

- Original virtualization concept: dynamic allocation of computing resources (CPU, memory, disk,..) in the form of virtual machines.
- Enables the flexible deployment of the distributed system (processes).
- Flexibility is critical in dense and heterogeneous environments (such as smart grids)
- Virtualization categories:
  - Connectivity virtualization:
    - Deploying data streams with certain specifications while providing QoS (Quality of Service) guarantees.
    - SDN (Software-defined Networking) architecture.
    - IEEE 802.1 TSN (Time-sensitive Networking).



Connectivity virtualization

dVPP

Communication within the middleware (compute units, CI-DRES)

Virtualization

ICT virtualization

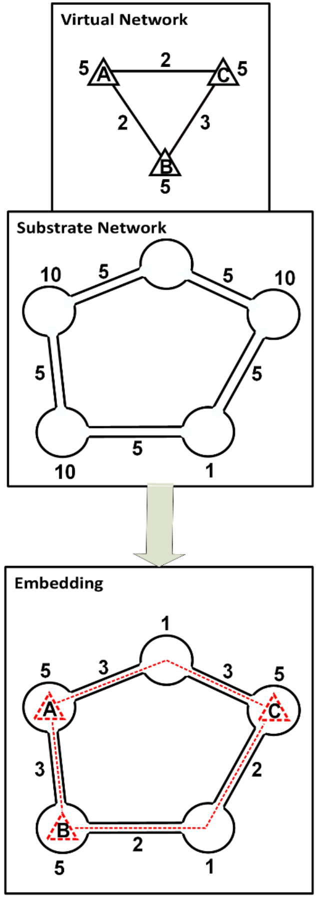
dVPP

- Virtualization categories (cont..):
  - Functionality virtualization:
    - NFV (Network Function Virtualization) architecture: Decoupling network and application functions from hardware to standard servers
    - High availability (replication and migration)
    - Performance challenges
  - Resource virtualization:
    - VNE (Virtual Network Embedding): Graph theory-based methods to allocate resources

Example: decoupling optimization and security functions from the proprietary hardware.

For example, allocating the communication services between the optimization functions and set-pointing functions

Resource virtualization



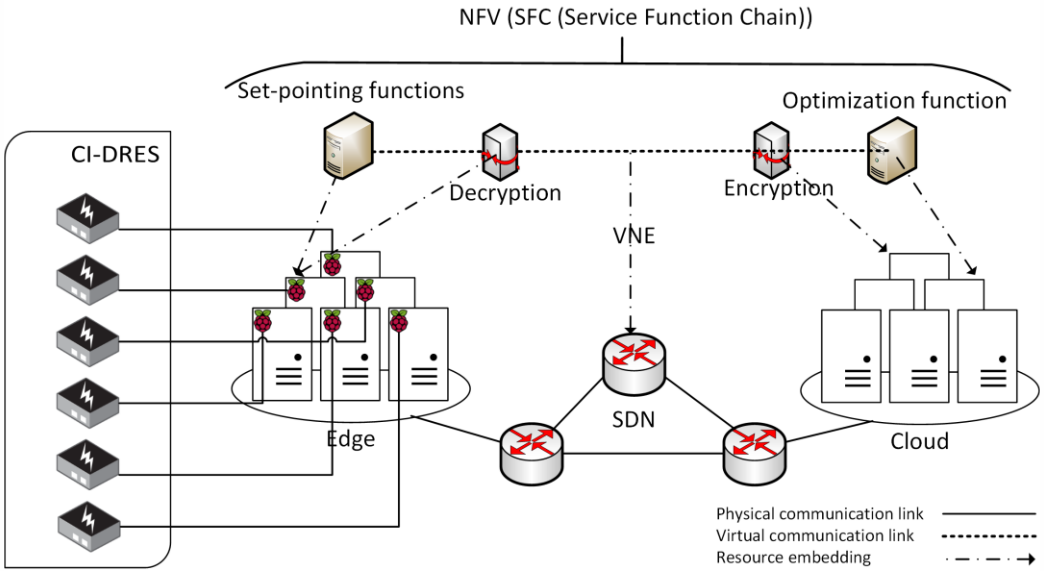
Virtualization

Virtualization Example

dVPP

Using virtualization to deploy a function chain for AS management with QoS guarantees

Functionality: Optimization → Encryption → Decryption → Set-pointing
Connectivity: SDN for deploying the virtual communication links
Resource: VNE for allocating set-pointing functions to the edge (Raspberry PIs), and optimization functions to the cloud servers



Virtualization

Q&A

- In our virtualization example, which virtualization methodology is used for allocating set-pointing functions to the edge (Raspberry PIs)?

A. SDN

B. NFV

C. VNE

Agenda

- Motivation
- Distributed Systems
- Virtualization
- EASY-RES ICT Architecture

EASY-RES ICT Architecture

Role of ICT System

Synchronous Generator	Distribution System	Active Distribution System
• (Re-)Active power generation • Ancillary service provision • Control & scheduling • Reliability	• Fluctuating load • Frequency & inertia balancing (External) • Little control over DRES • Unreliable standalone sources	• Fluctuating load & generation • Ancillary service provision • Control & scheduling of DRES • Reliability by aggregating unreliable sources → **Managed by ICT**

→Key takeaway: Enable distribution system AS capabilities by ICT

EASY-RES ICT Architecture

Role of Virtualization

- Management via ICT to improve distribution system capabilities requires:
  - Representation of the power grid for various use cases
  - Secure, reliable, low latency communication & computation
  - Representation of ICT resources for computation and communication

ICT Virtualization	Power Grid Virtualization

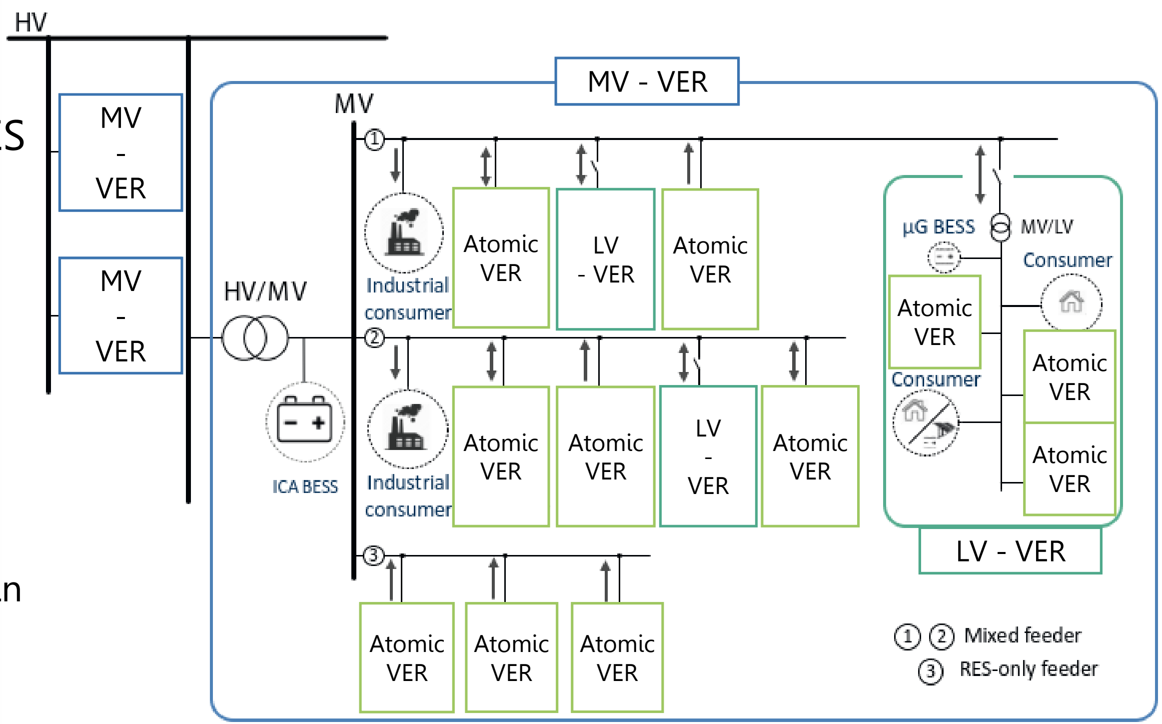
• Security via VPN, PKI
• Reliable computations via NVF
• Reliable communication via SDN
• (Dis-)Aggregation
	• UVSG applied to CI-DRES
	• Hide irrelevant aspects of:
	  • Device characteristics
	  • Measurements
	  • Grid topology
	  • See also DDGM (from part 7c)

Combined in Virtual Energy Resource (VER): • Use ICT virtualization & Power grid virtualization • Represent one DRES or grid area • Provides interface to request ancillary services • Ensures proper & optimal operation internally	

EASY-RES ICT Architecture

VER example:

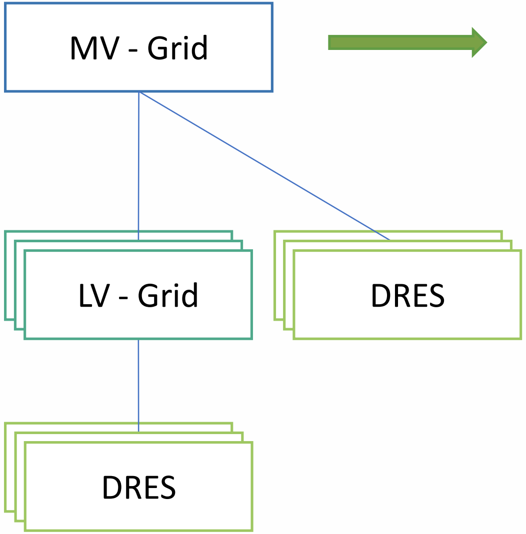
- Hide differences between DRES installation and aggregation:
  - Same virtual representation
  - → Atomic VER for DRES
  - → MV- / LV-VER at respective voltage level
- Aggregation, monitoring and other functions require communication:
  - → MV / LV – VER need to contain power & ICT resources

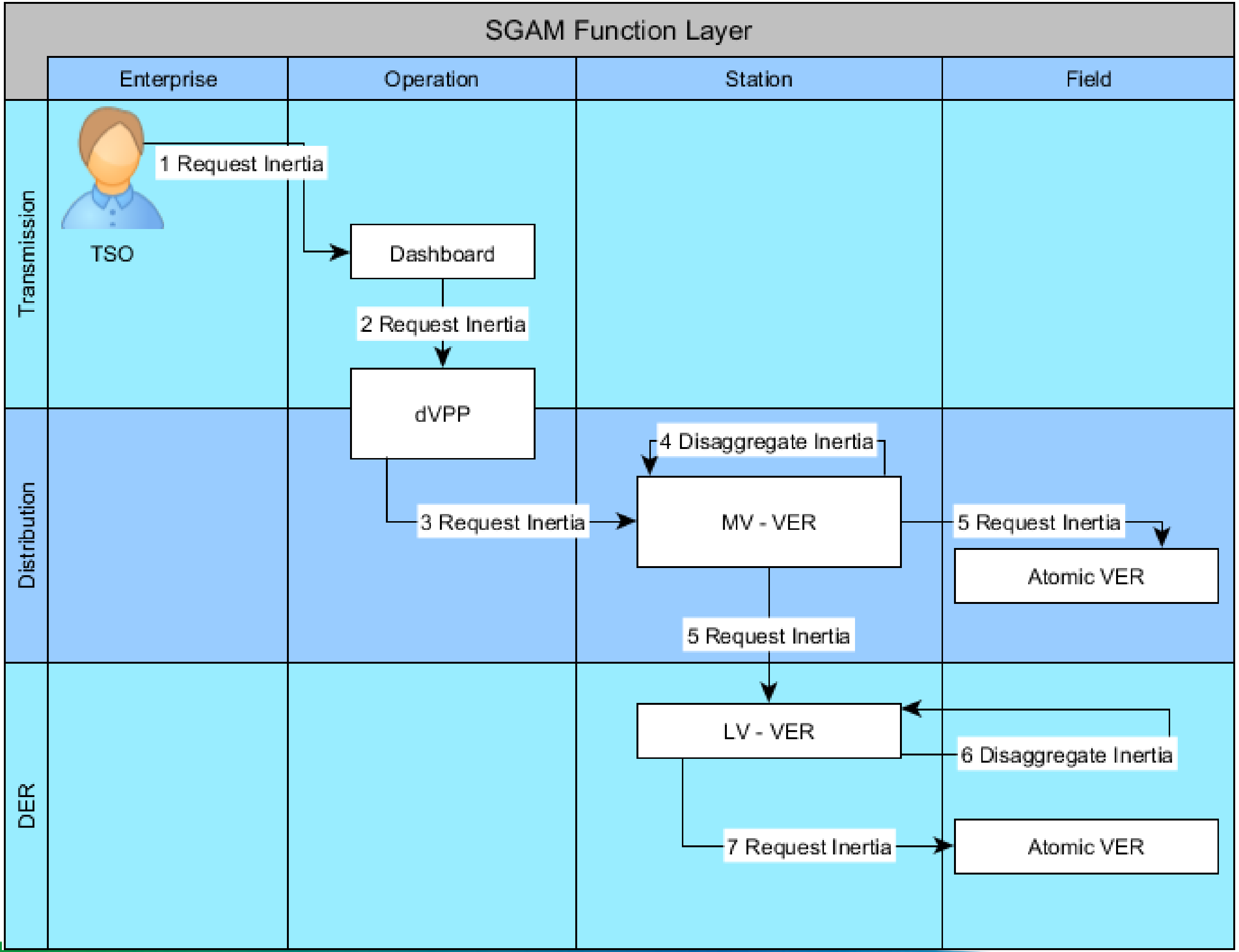


EASY-RES ICT Architecture

Scenario

2nd step of inertia Disaggregation (Talk 7b)



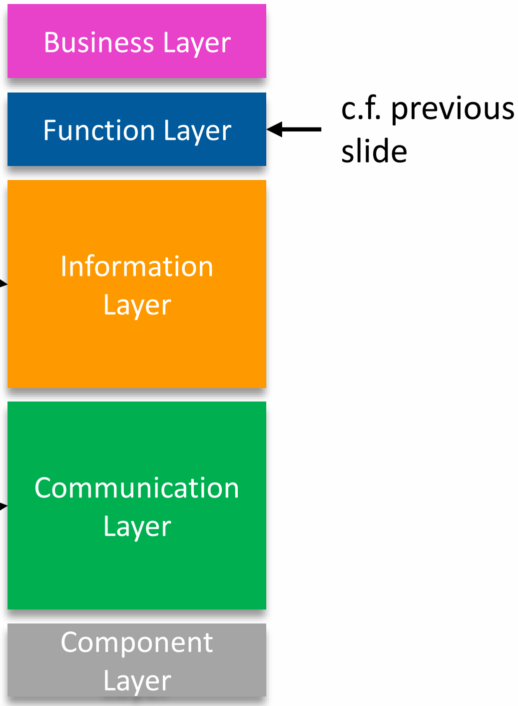


EASY-RES ICT Architecture

VER implemented as a "Thing"

- W3C Web of Things - Thing Description
  - Example: Atomic VER
- „What can it do?"
  - Interaction Affordances
    - Properties: e.g. maximum inertia capability
    - Actions: e.g. enable AS and set inertia constant
    - Events: e.g. accounting metrics
  - Data Schemas: e.g. inertia constant as floating number
- „How to communicate with it?"
  - Security Configuration: e.g. encryption, authorization
  - Protocol Binding, e.g. Modbus-TCP, MQTT…

SGAM Interoperability Layers



https://www.w3.org/TR/wot-thing-description

EASY-RES ICT Architecture

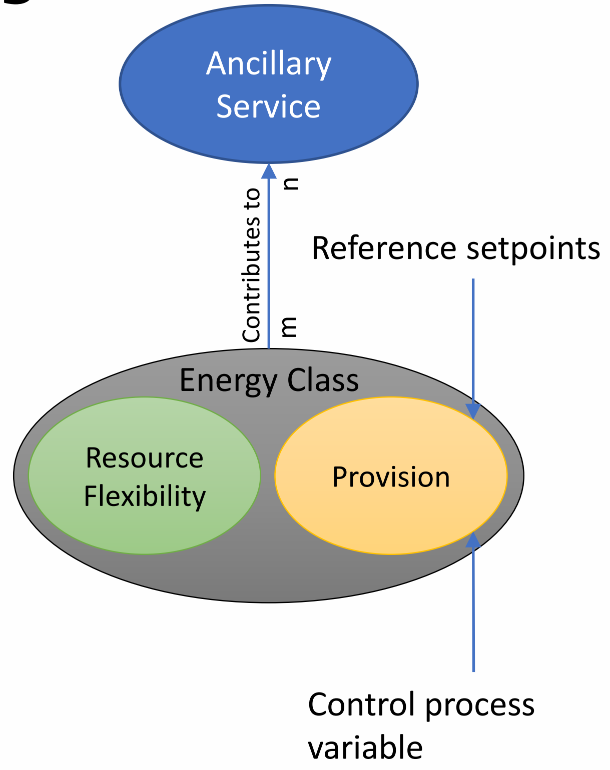
Achieving Design Goals: Openness

- Openness requires abstraction of internal differences
- Creation of standard interface to all DRES
  - Characterize required service for contribution to AS
  - Characteristics of the flexibility:
    - Ramp Rate
    - Capacity
    - Active or Reactive power (magnitude)
  - Characteristics of the provision:
    - Control process variable → PFR: Power output
    - Measured inputs → PFR: Frequency
    - Reference setpoints → PFR: P(f)-droop curve

dVPP

→ Result: Energy Class (EC)

- Device-agnostic
- Interchangeability of different generation technologies

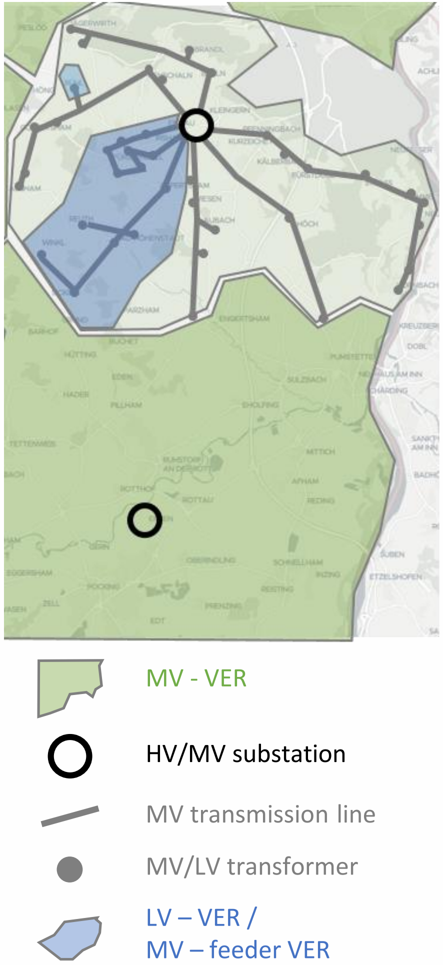


EASY-RES ICT Architecture

Achieving Design Goals: Transparency

- Need-to-know principle: Operations only have access to required data items
- Location transparency:
  - Only aggregated resources visible at higher hierarchy levels
- Access transparency:
  - Access CI-DRES with direct connection and aggregated resources in the same way
- Replication transparency:
  - Hide the existence of contingency resources to ensure service in case of (ICT -) failure
- Migration transparency:
  - Hide that service is shifted within a grid area while keeping total service the same
- Concurrency transparency:
  - Prevent conflicts when providing multiple services

→ Solution: Hierarchical (dis-)aggregation of Virtual Energy Resource each offering the Energy Class Interface



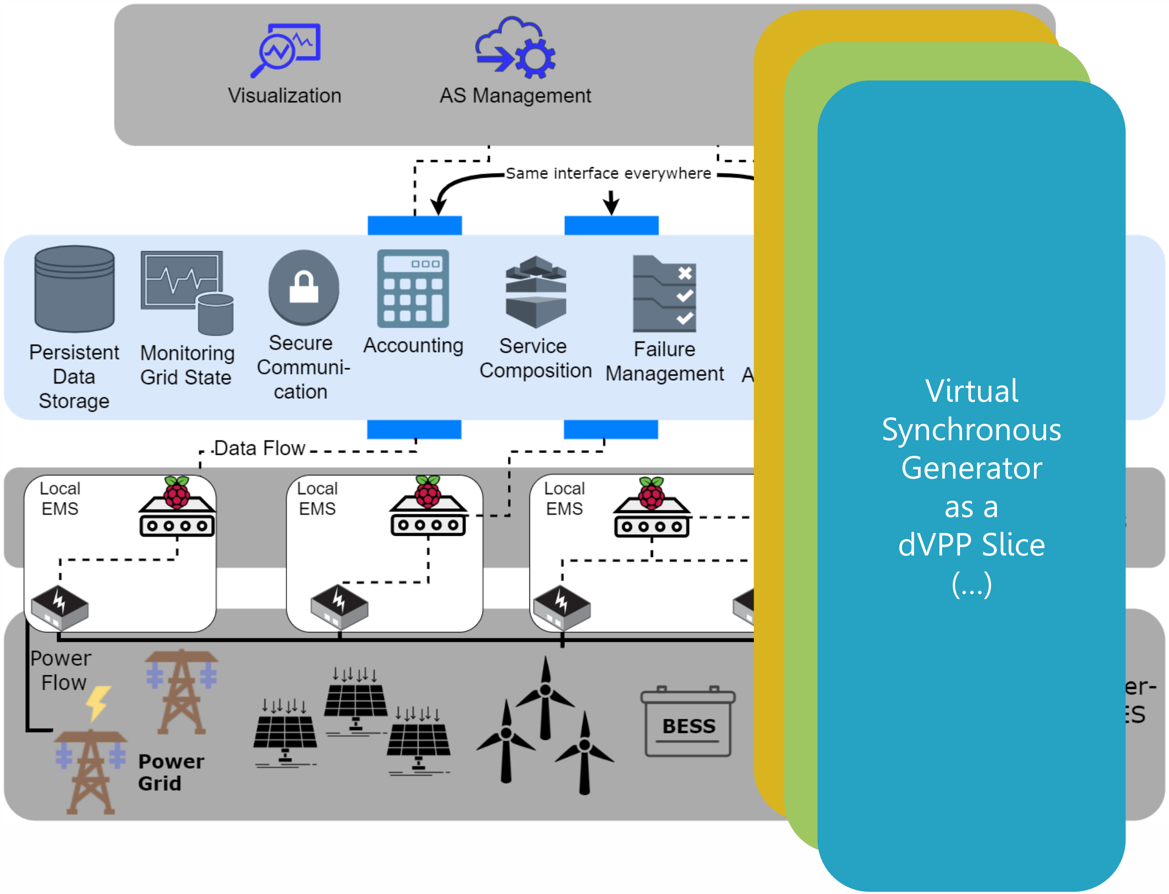
EASY-RES ICT Architecture

Functional View

- Abstraction levels:
  - High level applications at the top
  - Enabled by middleware:
    - Monitoring, Discovery
    - VERs used for (Dis-) Aggregation
  - Selectively hidden details of component layer
- Virtual Synchronous Generator as a service:
  - Slice of the dVPP
  - Using resources on all layers
  - Multiple AS in parallel

  → Separation of concerns: Different view for different AS
- Middleware and applications form central control

Dynamic Virtual Power Plant



[Stocker 2022]

EASY-RES ICT Architecture

Conclusion

- ICT middleware for active distribution networks automation for system services
  - Achieving high level of transparency
  - Openness with respect to:
    - CI-DRES from multiple vendors
    - Different renewable sources
  - Scalability in all dimensions
  - Utilizing both ICT virtualization and power virtualization
  - Novel key concepts:
    - Virtual Energy Resources
    - Energy Classes
- Offering Virtual Synchronous Generator as a service:
  - Slice of the whole Dynamic Virtual Power Plant:
    - Tailored set of ancillary services
    - Provided by an optimized set of resources (CI – DRES & ICT units)

Q&A

- Which of these components present the highest level of abstraction?

A. CI - DRES

B. MV - VER

C. Atomic VER

D. LV – VER

E. They all present the same level of abstraction

EASY-RES ICT Architecture

Q&A

- What concept is used to achieve the Openness Design Goal?

A. Virtual Energy Resources

B. Energy Class Interface

C. Software Defined Networking

D. None of the above

EASY-RES ICT Architecture

Q&A

- Which transparency is **NOT** achieved using Virtual Energy Resources?

A. Location transparency

B. Access transparency

C. Migration Transparency

D. Replication transparency

E. Concurrency transparency

F. None of the above

References and Further Readings

[vanSteen 2017] Maarten van Steen, Andrew S. Tanenbaum,"Distributed Systems, Principles and Paradigms", Pearson Education, Inc. Edition: 3rd (3. Jan. 2017), ISBN-10: 1543057381, ISBN-13: 978-1543057386+

[Haberfellner 2019] Haberfellner, R., de Weck, O., Fricke, E., & Vössner, S. (2019). Systems Engineering–Grundlagen und Anwendung. 12. *Auflage. Zürich: Orell Füssli*.

[Strickland 2011] Strickland, J. (2011). *Systems Engineering Processes and Practice*. Lulu. com.

[Boardman 2006] Boardman, J., DiMario, M., Sauser, B., & Verma, D. (2006, July). System of systems characteristics and interoperability in joint command and control. In *2nd Annual System of Systems Engineering Conference, Defense Acquisition University, Ft. Belvoir, Virginia*.

[Kurose 2016] Computer Networking: A Top Down Approach, 7th edition, Jim Kurose, Keith Ross. Addison-Wesley, April 2016

[Stocker 2022] Stocker et al., An ICT Architecture for Enabling Ancillary Services in Distributed Renewable Energy Sources Based on the SGAM Framework [to be submitted]

The Consortium

ARISTOTLE UNIVERSITY OF THESSALONIKI

UNIVERSIDAD DE SEVILLA

UNIVERSITÄT PASSAU

Lancaster University

TU Delft

elektro gorenjska

Stadtwerke LANDAU a.d.ISAR

stadtwerk haßfurt

ΑΔΜΗΕ
ΑΝΕΞΑΡΤΗΤΟΣ ΔΙΑΧΕΙΡΙΣΤΗΣ ΜΕΤΑΦΟΡΑΣ ΗΛΕΚΤΡΙΚΗΣ ΕΝΕΡΓΕΙΑΣ

fenecon ENERGY ENGINEERING

ZD.B ZENTRUM DIGITALISIERUNG. BAYERN

This project has received funding from the European Union´s Horizon 2020 Programme for research and innovation under Grant Agreement no 764090.

Thank you!

Hermann de Meer

Armin Stocker

Affiliation:	Chair of Computer Networks and Computer Communications University of Passau, Germany
Phone:	+49 851 509 3051
E-Mail:	hermann.demeer@uni-passau.de
EASY-RES website:	http://www.easyres-project.eu/