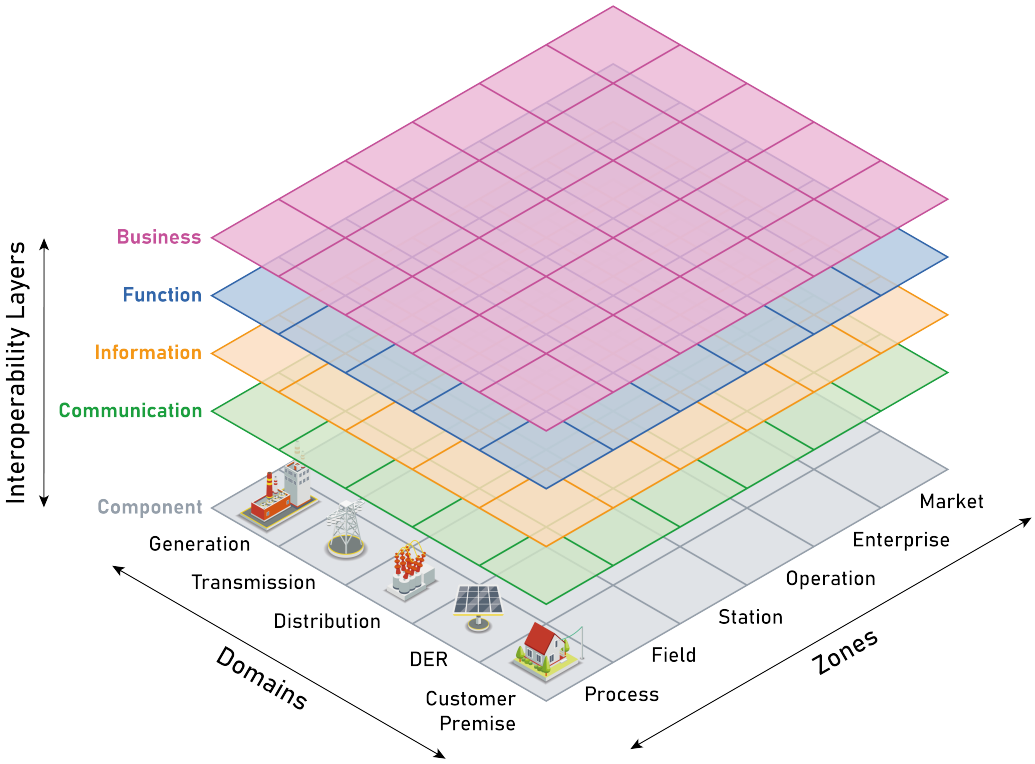
Application of the Smart Grid Reference Architecture Model with the SGAM Toolbox



Center for Dependable Systems Engineering

Salzburg University of Applied Sciences

Authors
Vereno D., Neureiter C., Eschlberger S., Millaku M.

August 2025

Contents

1 Introduction

The document at hand introduces the **SGAM-Toolbox**—a purpose-built modeling environment grounded in the *Smart Grid Architecture Model* (SGAM), developed under the 2012 Smart Grid Coordination Group (SGCG) initiative. Built as an add-in for Sparx Systems' *Enterprise Architect*, the toolbox implements a domain-specific modeling language aligned with model-based systems engineering approaches to capture the *domains*, *zones*, and *interoperability layers* of smart grid architectures.

At its core, the SGAM-Toolbox empowers interdisciplinary teams to collaboratively design, visualize, and validate high-level system architectures through a standardized framework. By anchoring models in a UML-based metamodel conformant to the *Meta Object Facility (MOF)* and integrated into Enterprise Architect via MDG Technology and a C# add-in, the toolbox simplifies architecture modeling while preserving semantic rigor.

This toolbox is not intended as a substitute for foundational methods—such as UML modeling, systems engineering, or use-case management—but rather serves as a *practical bridge* between those established practices and the SGAM framework. It supports development by providing structured, reusable modeling constructs grounded in recognized smart grid standards.

Documented in detail in our *IEEE Open Access* paper (Access 13, pp. 119243–119261, 2025; DOI: 10.1109/ACCESS.2025.3586722), the SGAM-Toolbox underwent a comprehensive revision to version 3.0. This updated version strengthens its formal underpinnings, sharpens the linkage between requirements engineering and architecture modeling, refines semantic consistency across viewpoints, and clearly delineates logical and technical layers in the architecture model.

This documentation is structured to provide a clear and concise overview of the SGAM framework as implemented in the SGAM-Toolbox. It introduces the **SGAM planes and layers**, explains the **domain-specific language (DSL) elements** provided by the toolbox, and includes a **modeling guide** that demonstrates how to apply these concepts in practice. While the focus is on practical usage within the modeling environment, a deeper explanation of the underlying concepts, design decisions, and technical implementation can be found in the accompanying IEEE publication.

2 Installation of the SGAM Toolbox

The *SGAMToolbox.Setup.0.0.msi* file will be downloaded to the specified location. After the download has finished, open the downloaded file. This will open the *SGAM Toolbox Setup Wizard*, as depicted in Figure 1.

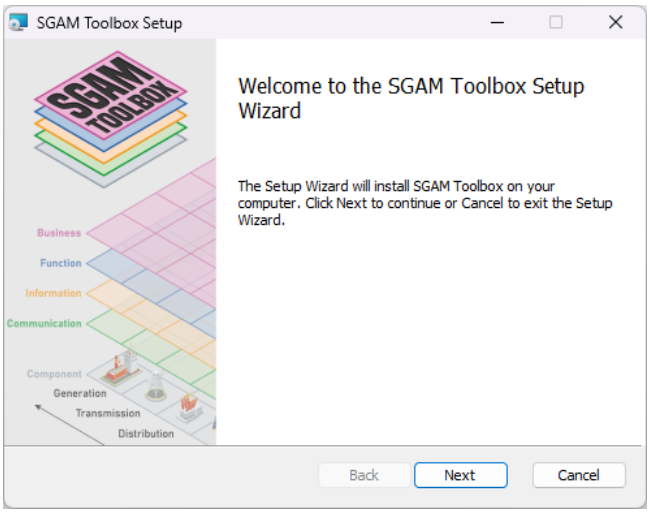


Figure 1: Installation wizard

Click *Next* and accept the license agreement, then click *Next* again. Specify the installation location for the SGAM Toolbox or leave the default location ("*C:\ProgramFiles(x86)\SGAM Toolbox*"). Click *Next* and *Install*. Finally, after the installation is completed, click *Finish* (Figure 2).

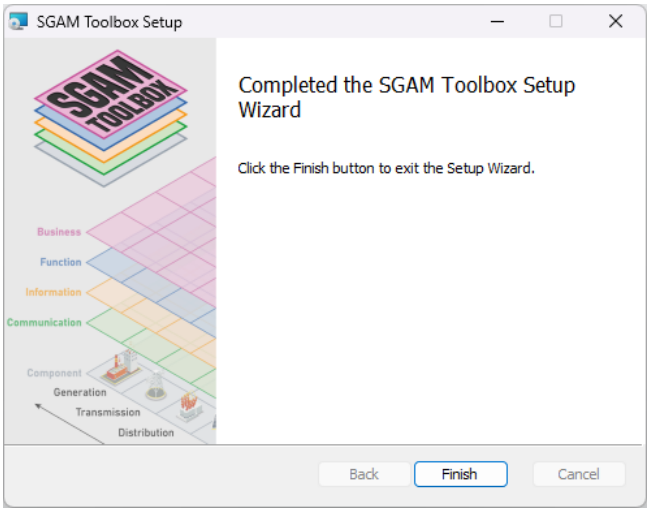


Figure 2: Finished installation wizard

Now, you can open Enterprise Architect. If you go to *Specialize* you will find the SGAM Toolbox under *Add-Ins* (Figure 3).

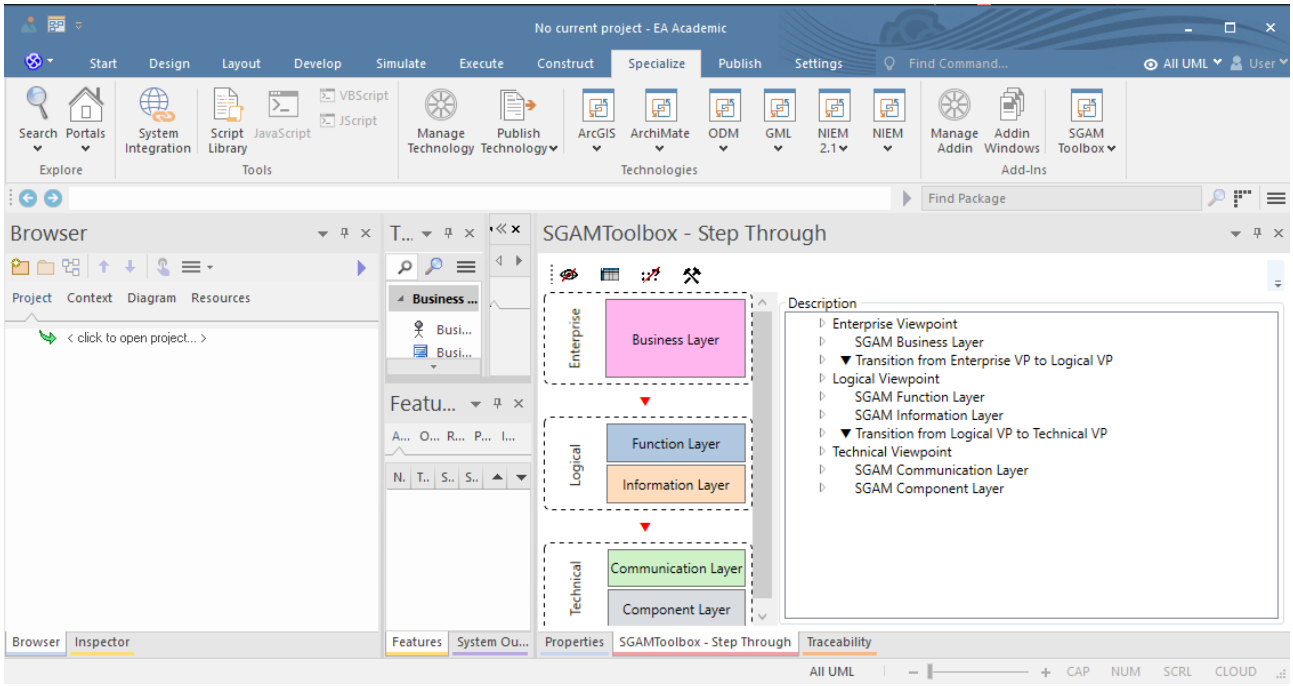


Figure 3: Where to find the SGAM Toolbox in Enterprise Architect

3 The Smart Grid Reference Architecture Model

The *Smart Grid Architecture Model (SGAM)* provides a structured, multidimensional framework for designing and analyzing Smart Grid systems. Originally introduced by the Smart Grid Coordination Group (SGCG) in 2012 as part of the European Mandate M/490, SGAM supports harmonized system development by visualizing architecture elements along three key dimensions: **interoperability layers**, **zones of operation/automation**, and **domains** of the energy system.

The SGAM framework helps stakeholders from various disciplines to align their viewpoints by offering a common modeling language and reference structure. It enables system architects to describe functions, components, and communication paths across business processes, IT systems, and physical infrastructure in a consistent and traceable way.

The SGAM-Toolbox builds directly on this model by providing predefined modeling elements and a visual grid for positioning system components according to their function, location, and level of abstraction. Understanding the SGAM dimensions is a prerequisite for using the toolbox effectively in smart grid projects.

Figure 4 illustrates the SGAM reference architecture model with its five interoperability layers (Business, Function, Information, Communication, Component), mapped over the zones (Process to Market) and domains (Generation to Customer Premises).

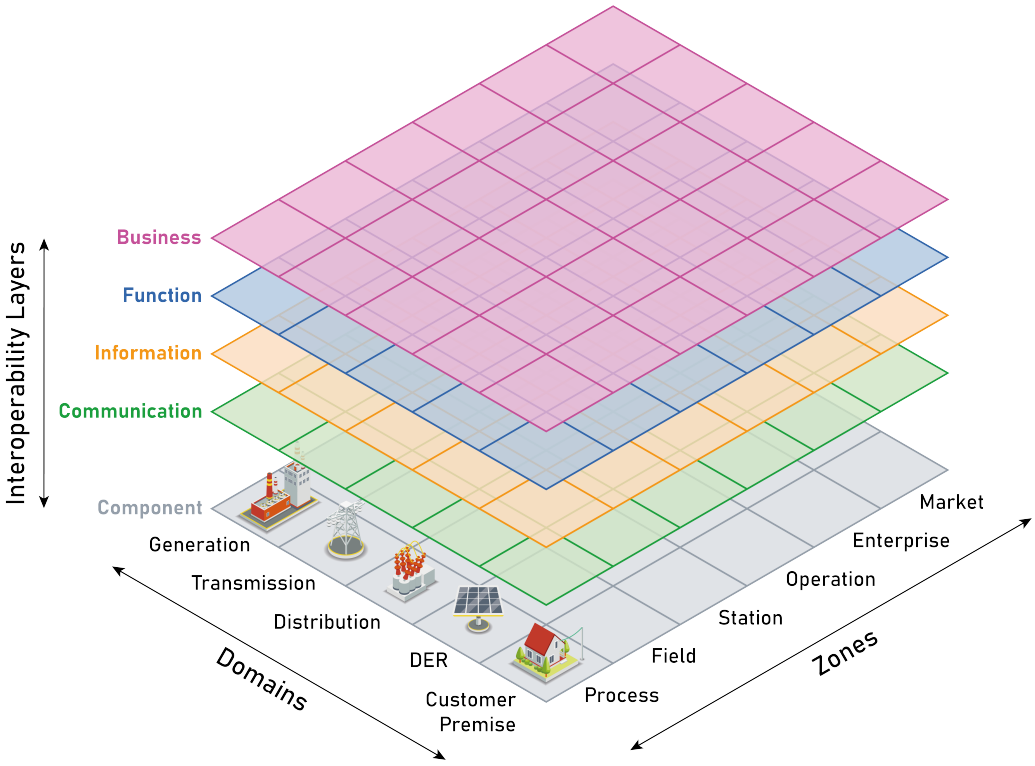


Figure 4: Smart Grid Reference Architecture Model (SGAM) framework

3.1 SGAM Plane

SGAM is a framework for allocating elements of a smart-grid architecture to a three-dimensional cuboid, which is depicted in Figure 5. Its structure is comprehensively described in the Smart Grid Reference Architecture by the Smart Grid Coordination Group: Two of its three axes define the so-called SGAM plane, which enables the structured placement of system elements based on their functional role and position within the energy system. This plane forms the foundation for describing system architectures in a clear and consistent way, facilitating interdisciplinary communication and system-level traceability.

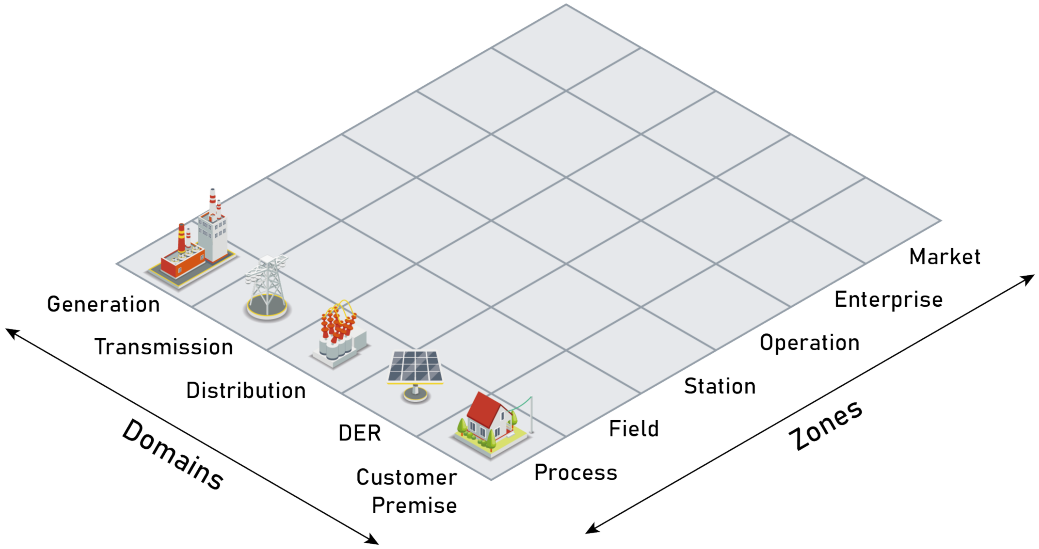


Figure 5: SGAM plane

The *domain* axis represents the energy-conversion chain whereas the orthogonal zones axis assigns elements along the automation pyramid. The domain axis is partitioned into five distinct categories, as described in Table 1.

Domain	Description
Bulk Generation	represents the bulk generation of energy, mainly by power plants
Transmission	represents the long-distance transportation of electrical energy
Distribution	represents the local distribution of electricity to consumers
DER	(distributed energy resource) represents small-scale power generation directly connected to the distribution grid
Customer Premises	represents the electricity consumption by households or industrial applications and the generating resources directly connected to consumer

Table 1: SGAM domains

The vertical axis of the SGAM plane defines the *zones*, which represents the hierarchical levels of power system management. This structure reflects the traditional automation pyramid and is rooted in principles of *spatial and data aggregation* as well as *functional separation*.

The six SGAM zones—Process, Field, Station, Operation, Enterprise, and Market—enable the classification of functions according to their control level and proximity to the physical process (see Table 2).

Zone	Description
Process	represents processes or equipment directly involved in the generation of electrical energy
Field	represents components for the protection, control, and monitoring of power systems
Station	represents various data and areal aggregation processes from the zone below which includes SCADA systems
Operation	represents various control systems, like a charging station management system (CSMS)
Enterprise	represents organizational aspects, like services and infrastructure for enterprises
Market	represents activities of the energy market—retail and wholesale

Table 2: SGAM zones

3.2 SGAM Interoperability Layers

Achieving sufficient interoperability between elements in large heterogeneous systems of systems is challenging. To address this, the GWAC interoperability stack was applied to the smart-grid domain by adding an orthogonal axis to the SGAM plane and creating five parallel projections of the plane, each corresponding to an interoperability layer. Interoperability must be assured on different levels—from a business perspective down to physical connections.

Layer	Description
Business Layer	describes business and regulatory aspects, including business models and projects as well as various stakeholders
Function Layer	is centered around use cases. At this level, use case functionality is modeled independently of its physical implementation. To model functions and services at a sufficient level of detail, sub-use-cases may be defined
Information Layer	is used for modeling the information exchange between actors, functions, services, and components. It is primarily focused on information objects: elements that represent pieces of information. An information object may be described by a data model if a suitable one is available. The information layer is vital for enabling interoperable information exchange within a smart-grid system.
Communication Layer	describes the protocols and other mechanisms for information exchange
Component Layer	contains the technical components, which include network and electricity infrastructure

Table 3: SGAM layers

4 SGAM Domain-Specific Language

The main feature of the SGAM toolbox is the domain-specific language (DSL) that is delivered with it. This DSL comprises modeling elements and relationships that aid the developers of a smart grid system in expressing it using a model-based approach. Each interoperability layer of the framework uses different elements and relationships to express different views on the system. In the following, each element of the SGAM DSL is listed and explained.

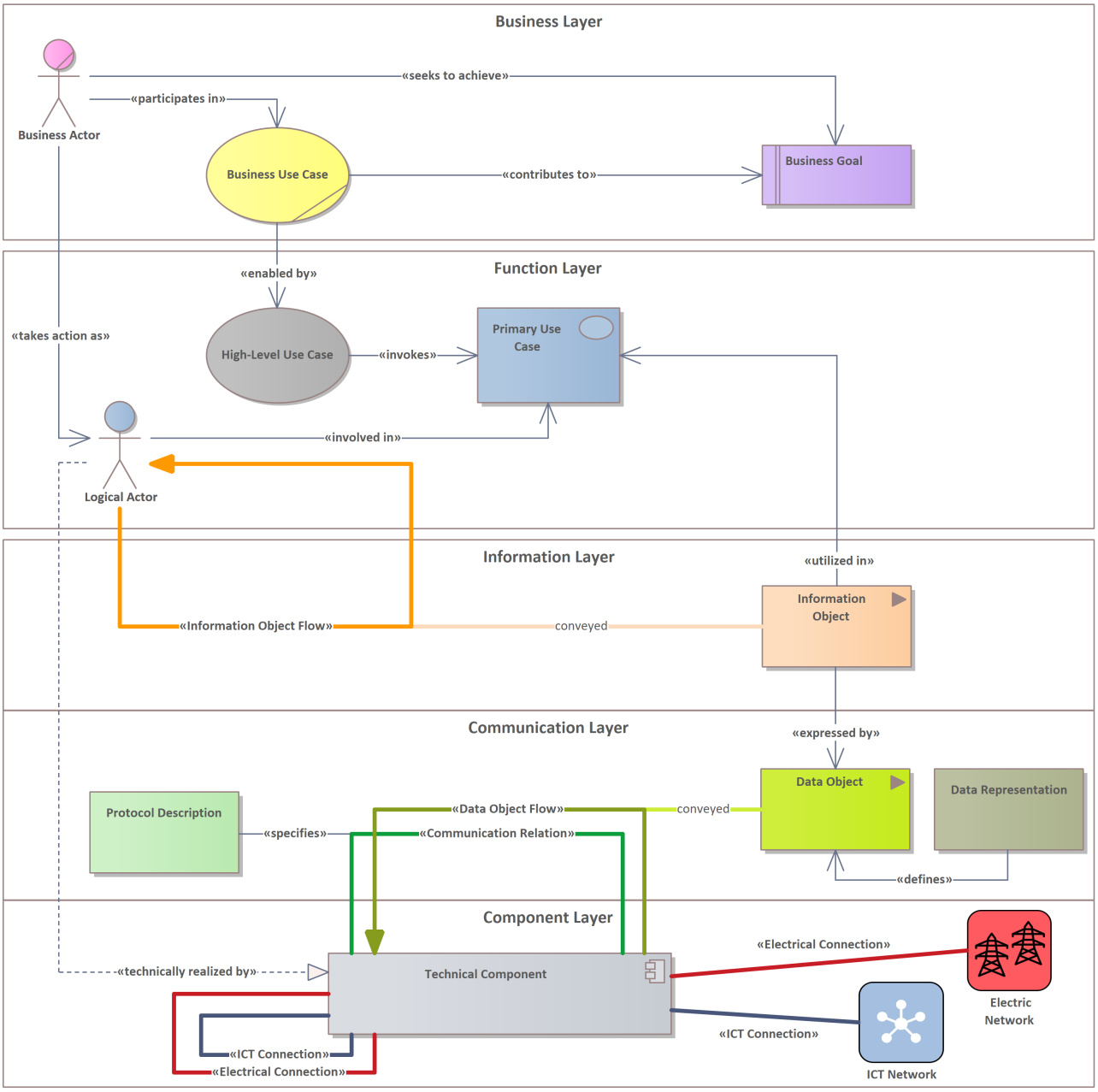


Figure 6: Overview of the SGAM DSL

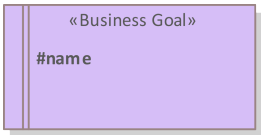
Business Layer Elements		
Business Actor	A *Business Actor* represents an entity fulfilling one role and being intrinsically linked to that role. It Seeks To Achieve its Business Goals by participating in one or more Business Use Cases. Relating to requirements engineering, a stakeholder can hold one or more roles relevant to the Business Use Cases of interest and is therefore represented by one or more Business Actors. A Business Actor and its role including tasks, duties, privileges and interests are often homonymous.	
Business Goal	A *Business Goal* is a desired outcome that a Business Actor Seeks To Achieve, expressed in a way that is measurable and achievable, and that contributes to the overall success of the organization or fullfillment of needs of an individual role of a Business Actor.	
Business Use Case	A *Business Use Case* is a description of a sequence of actions that one or more Business Actors Seek to Achieve their Business Goals, expressed in a way that is independent of any specific system or technology. Also, a Business Use Case represents the central concept being analyzed by a Business Case and can serve as a foundation for executing a Business Case Analysis.	

Table 4: SGAM business layer elements

Business Layer Relationships		
Contributes To	A *Contributes To* relationship is modeled between a Business Use Case and a Business Goal. It expresses that the Business Use Case is in accordance with a Business Goal to foster its fulfillment.	
Participates In	A *Participates In* relationship indicates that a Business Actor is actively involved in carrying out a Business Use Case to fulfill their Business Goal. Hence, it is a relationship between Business Actors and Business Use Cases.	
Seeks To Achieve	A *Seeks To Achieve* relationship is a relationship between a Business Actor (BA) and a Business Goal (BG). It expresses, that a BA has an intention and actively attempts to accomplish the respective BG.	

Table 5: SGAM business layer relationships

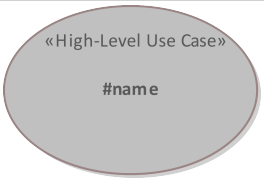
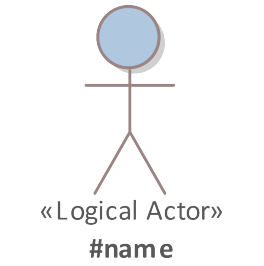
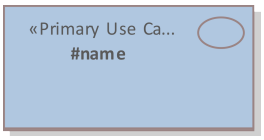
Function Layer Elements		
High-Level Use Case	A *High-Level Use Case* is a use case, that describes a general system capability, idea or concept independently from a specific technical realization or architectural solution.	
Logical Actor	The *Logical Actor (LA)* element represents logical entities involved in the realization of one or more Primary Use Cases. A Logical Actor is obtained on basis of a model transformation from a certain Business Actor (BA). Thus, by following the given Takes Action As relation, each LA is uniquely associated with one specific BA.	
Primary Use Case	The *Primary Use Case (PUC)* element is used to refine High-Level Use Cases (HLUC) into more specific use cases that the HLUC invokes with the ultimate goal to achieve a system functionality.	

Table 6: SGAM function layer elements

Function Layer Relationships		
Enabled By	An *Enabled By* relationship is modeled between a Business Use Case and a High-Level Use Case (HLUC). This relationship expresses that a HLUC represents the system-level realization of the business utility prescribed by a Business Use Case.	
Invokes	The *Invokes* relation between a High-Level Use Case (HLUC) and a Primary Use Case (PUC) indicates that the PUC encapsulates a specific sub-process that contributes to fulfilling the requirements of the preceding HLUC. This relationship implies that the PUC provides a more detailed and granular representation of a particular step within the broader process outlined by the HLUC.	
Involved In	The *Involved In* relationship between a Logical Actor (LA) and a Primary Use Case (PUC) expresses that the LA actively participates in the interactions or information exchanges occurring within the PUC. The LA can initiate actions as outlined in the PUC's process description or respond to queries or requests from other LAs involved in the same PUC. A LA can be engaged in multiple PUCs, and every PUC must involve at least one LA.	
Takes Action As	The *Takes Action As* relation between a Business Actor (BA) and one or more logical actors (LAs) indicates that the Business Actor's needs and responsibilities are represented by the logical actors in the Function Layer. This ensures that the BA's intended actions and interactions are captured and reflected in the Function Layer.	

Table 7: SGAM function layer relationships

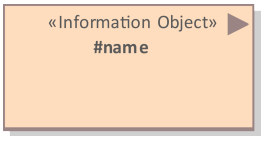
Information Layer Elements		
Information Object	Elements of the type *Information Object* are used to specifiy particular information to be exchanged. The definition of these Information Objects is obtained on basis of the use case description of concerning Primary Use Cases.	

Table 8: SGAM information layer elements

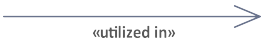
Information Layer Relationships		
Information Object Flow	The *Information Object Flow* element represents a relation between two Logical Actors. It associates particular Information Objects as specified on level of the Function Layer.	
Utilized In	The *Utilized In* relation between an Information Object and a Primary Use Case indicates that the Information Object is employed within the Primary Use Case to convey the meaning of specific information that is exchanged via messages between entities during the Primary Use Case's execution. The same Information Object can be Utilized In multiple Primary Use Cases, signifying that the same informational content is transferred between Logical Actors. This relationship does not mandate the specification of the information format or transmission mechanism, as these elements are yet to be defined later and not at the Function Layer. An Information Object does not necessarily need to be associated with an Information Object Flow if it merely represents information generated or stored within an entity without being transmitted to another entity at that point in time.	

Table 9: SGAM information layer relationships

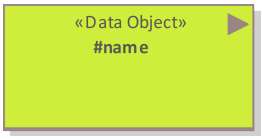
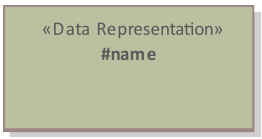
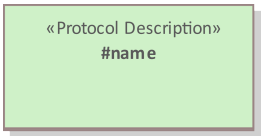
Communication Layer Elements		
Data Object	A *Data Object* is an instantiatable entity based on a Data Representation, embodying the structure and format defined by the representation. It encapsulates a collection of data elements organized according to the specifications of the representation. Data objects serve as the building blocks for exchanging and processing the encapsulated Information Objects within and between systems. They can be stored in various formats, such as XML documents, JSON files, or relational databases.	
Data Representation	A *Data Representation* is a formal specification that outlines the structure and format of Data Objects. It serves as a blueprint for organizing and composing data, ensuring consistency and facilitating data exchange between systems. Data Representations can be based on official standards like XML schemata or OWL ontologies, or they can be custom-defined to meet requirements.	
Protocol Description	A *Protocol Description* can be used to serve as a guidance for communication interaction, outlining the methods and technologies employed to exchange data. It encapsulates the high-level details of connection parameters, including the choice of protocols like HTTPS/REST or specialized ones like RTP, ensuring seamless data transmission between technical components. A Protocol Description can itself be composed of multiple ones by aggregations to model a protocol stack as it is done in the ISO/OSI model or TCP/IP model.	

Table 10: SGAM communication layer elements

Communication Layer Relationships		
Communication Relation	The *Communication Relation* element is used to express a relation between two Technical Components. This relation is used on the SGAM Communication Layer and covers information on the used protocol and technology but not its physical connection.	
Data Object Flow	The relation *Data Object Flow* refers to a transmission channel between Technical Components that enables the transfer of Data Objects. It represents the movement of data between these components, enabling the exchange of the encapsulated information. A Data Object Flow can convey one or more Data Objects.	
Defines	The *Defines* relation associates a Data Representation to one or many Data Objects. All Data Objects adhering to the Data Representation's structure and format are Defined by the Data Representation. This relation implies that the Data Representation serves as a blueprint for the organization and composition of the Data Objects. In other words, the Data Objects conform to the specifications outlined by the Data Representation. The Defines relation is not limited to Data Representations based on official standards; it can encompass custom or proprietary ones as well.	
Expressed By	The *Expressed By* relation between an Information Object and a Data Object signifies that the Information Object can be concretized and conveyed through a Data Object. This Data Object serves as a representation of the Information Object's abstract meaning. For instance, an HTTP message adhering to a well-defined format like JSON Schema or XML Schema (a Data Representation) can effectively express the content or idea of an abstract Information Object. The Data Object's structure and format provide a standardized mechanism for encapsulating and communicating the Information Object's essence.	
Specifies	The *Specifies* relation connects a Protocol Description to one or more Communication Relations. The purpose is to assign the connection parameters defined in the Protocol Description to a Communication Relation.	

Table 11: SGAM communication layer relationships

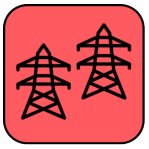
Component Layer Elements		
Electric Network	An *Electrical Network* is a interconnected system designed for transporting electrical energy. It encompasses both localized distribution grids and geographically expansive transmission grids. Technical Components like intelligent electronic devices (IEDs), transformers, and power plants can be connected to it. These networks are also enabled for the interconnection of multiple electrical networks.	
ICT Network	An *ICT Network* offers services for data transmission across varying spatial scales, ranging from localized home area networks (HAN) to geographically expansive wide area networks (WAN), comprising network devices such as 5G base stations, switches, and routers. If functioning as a black box, it links Technical Components, enabling the exchange of data between them. In contrast, a white box model needs a linked composite structure diagram or external resource that reveals the network's internal structure, showcasing the network topology of devices and their interconnections.	
Technical Component	The *Technical Component* represents a type of physical or virtual system part or a subsystem. It may be able to process electrical energy and/or data transferred to it by Electrical Connections or ICT Connections.	

Table 12: SGAM component layer elements

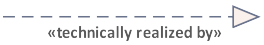
Component Layer Relationships		
Electrical Connection	*Electrical Connections* are relations, introduced to describe the electrical architecture on the abstract level of the SGAM Component Layer. Typically such relations are used as point-to-point connection between two technical components or, as connection between one Technical Component and one element of type Electric Network. If it makes no sense to specify an Electric Network between (i.e. it is a device-internal connection), then also two Technical Components can by directly connected.	
ICT Connection	*ICT Connections* are relations, introduced to describe the ICT architecture on the level of the SGAM Component Layer. Typically such relations are used as point-to-point connection between two technical components or, as connection between one component and one element of type ICT Network. If ICT Connections are device-internal, the ICT Network modeling element may be omitted and the two Technical Components directly connected.	
Technically Realized By	The *Technically Realized By* relationship denotes a many-to-many mapping between logical actors (LAs) and Technical Components (TCs). This implies that multiple LAs can be associated with various TCs, and vice versa.	

Table 13: SGAM component layer relationships

5 Modeling Guide

As already outlined in the installation section you need to have the modeling software Enterprise Architect and the SGAM Toolbox installed to follow this guide. This section provides a practical guide to using the SGAM Toolbox for modeling smart grid architectures, illustrating how to create an architecture model across the enterprise, logical, and technical viewpoints.
The scenario presented here is the same as the one you find in the scientific paper for the SGAM Toolbox'es revision [1]: A simplified example of a car rental company utilizing its electric vehicles (EVs) to participate in electricity markets, specifically through a virtual power plant (VPP) for providing balancing services. Figure 7 illustrates the scenario, its actors and their relations. If you want to follow the example closely, you can use the online-browseable demonstrative model [2].

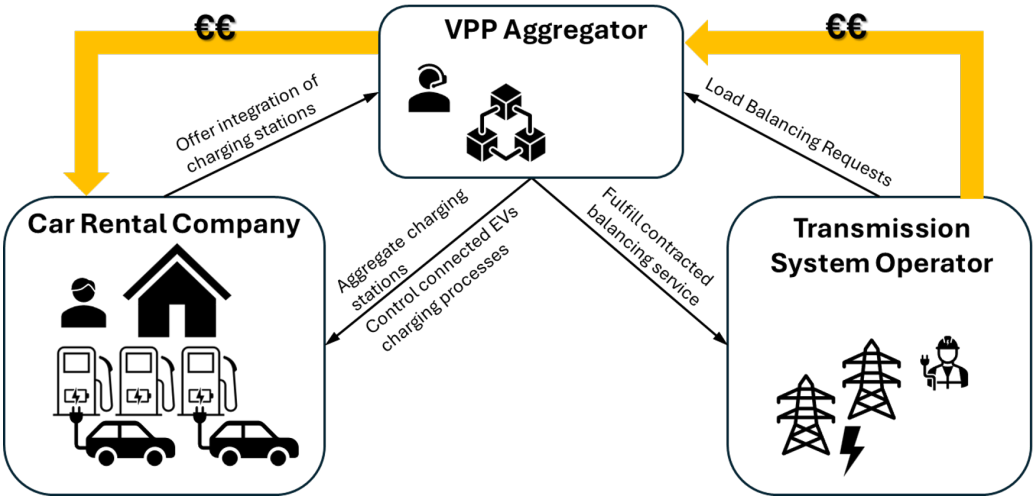
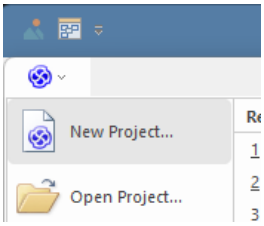


Figure 7: Scenario description: Car rental company utilizing EVs for VPP participation

5.1 Creating the modeling project

Open Enterprise Architect and create a new project by clicking on the EA icon in the left upper corner and select `New Project...`.



After selecting an appropriate place to save the project, you should see an empty project containing only a single root package in the model browser called `Model`. Figure 8 shows a good start of which windows we consider most useful for beginners to work with the SGAM Toolbox. If you do not see some of them, you can press the given keyboard shortcuts in the Figure 8 to unhide a window.

[1] https://doi.org/10.1109/ACCESS.2025.3586722
[2] https://dsse.at/sgam/downloads/v3.0.0.0/html-export-demo-parking-center-vpp/

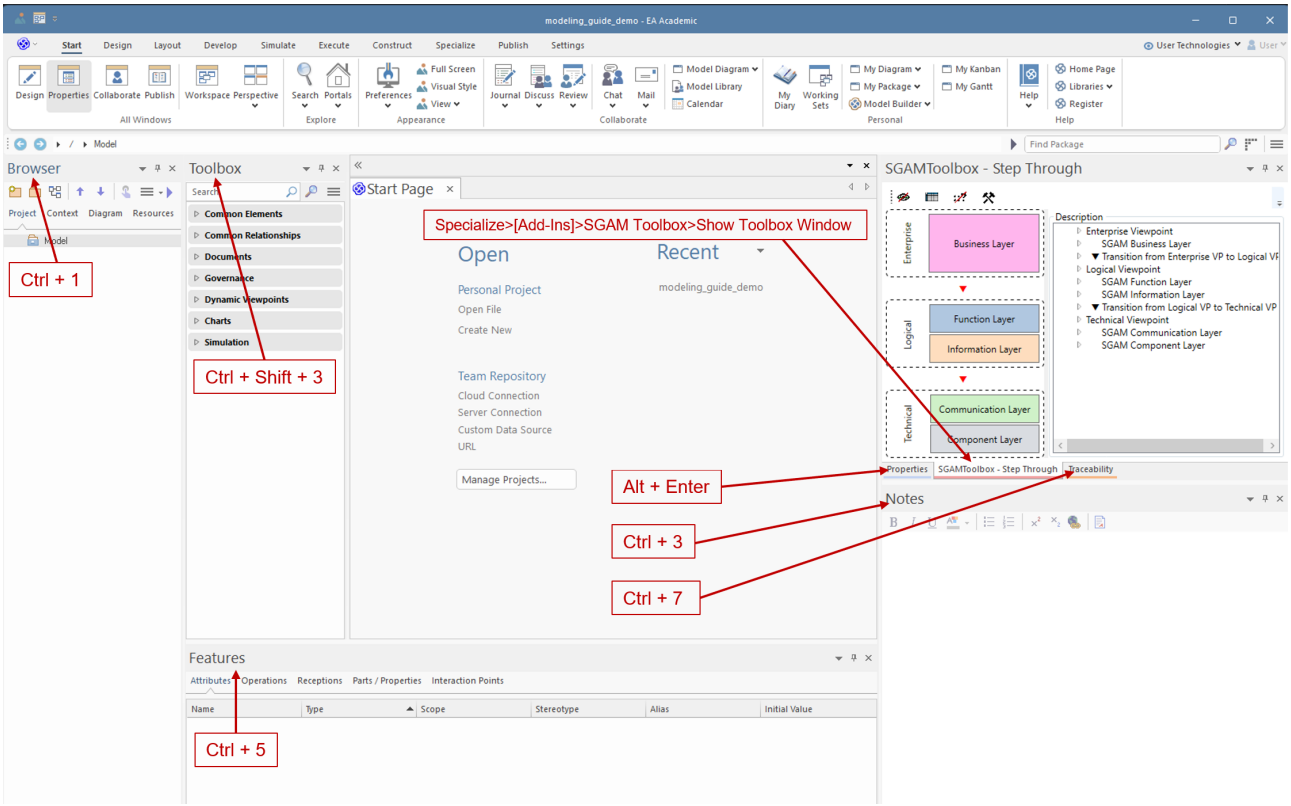
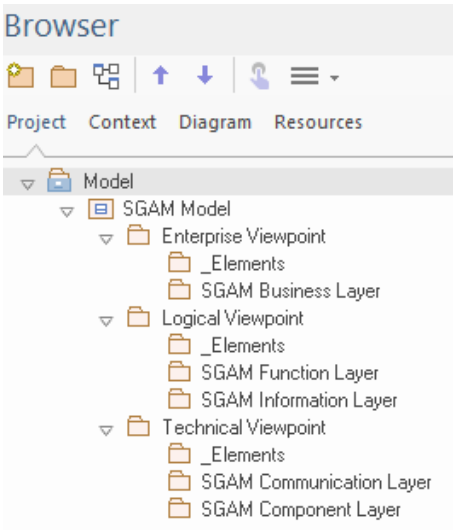


Figure 8: Empty project and useful windows in EA for SGAM Toolbox modeling

If you do not see the add-in window of the SGAM Toolbox, you can toggle its visibility via the menu `Specialize>[Add-Ins]>SGAM Toolbox>Show Toolbox Window`.

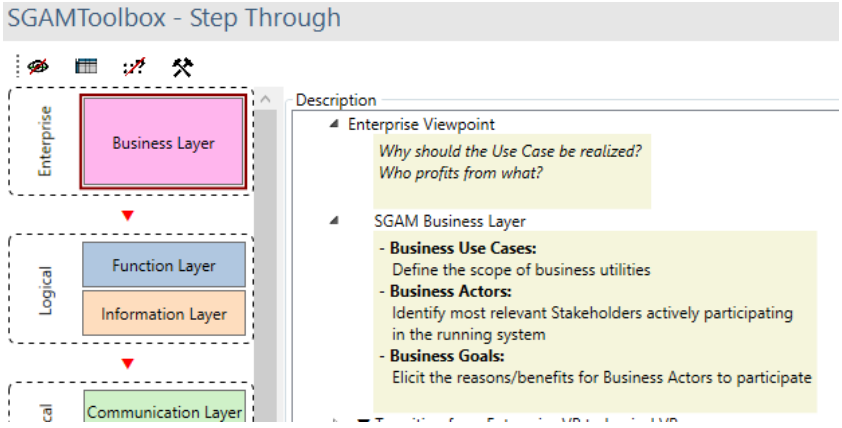
5.2 Base model structure

The package structure of a SGAM model is not a fixed hierarchy but rather depends on the modelers needs, model size, and complexity. In this guide we showcase a structure that we identified as a good start for small-sized projects. The SGAM Toolbox provides templates for multiple use cases. They are available by clicking on the package symbol with the star in the `Browser Window`. Because of the small size of the demonstrative project, we will create the model structure now manually.
In the `Browser Window` RIGHT-Click on the `Model` package and select `Add View`. Name it `SGAM Model` and click `OK`. This package will contain all the model elements of the SGAM model.
Then repeat the process by three times RIGHT-Clicking on the `SGAM Model` package and selecting `Add View` to create the packages `Enterprise Viewpoint`, `Logical Viewpoint`, and `Technical Viewpoint`. In each of the three viewpoints packages, create a package `_Elements` to hold the model elements of the respective viewpoint. Only in the enterprise and technical viewpoints create also a package called `_Model Transformation` to hold the interlinking diagrams between viewpoints. Directly beneath each viewpoint create packages for the SGAM layers. The final structure should resemble the following:



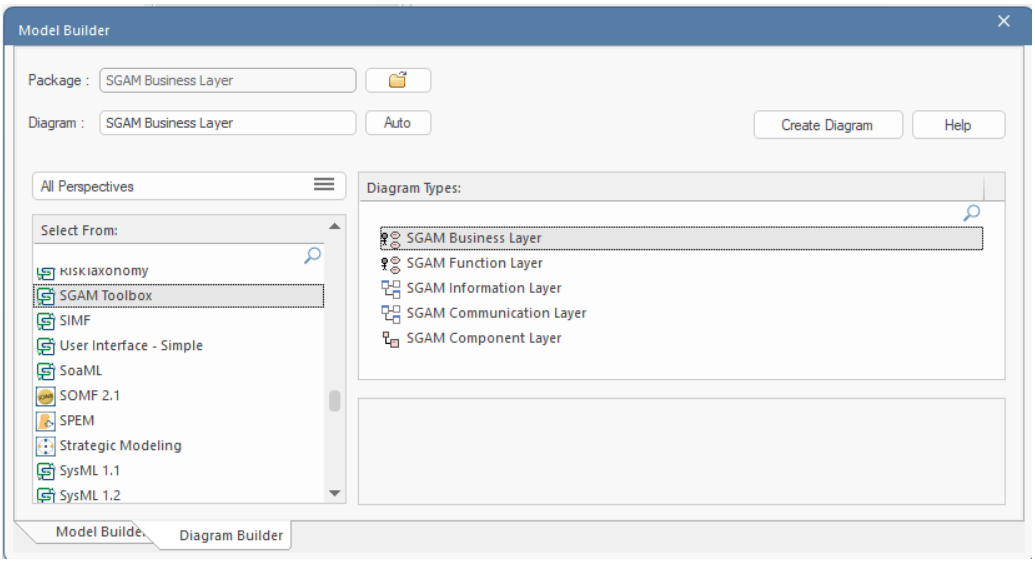
5.3 Enterprise Viewpoint – Modeling the business value

In the Enterprise Viewpoint the focus is on capturing and structuring business-level concerns. The `SGAM Business Layer` is identifying `Business Actors` – stakeholders who actively participate in business processes – and defining their measurable, achievable `Business Goals` they have in related `Business Use Cases`. Additional concerns such as business processes, organizational structure, policies, strategic alignment, value streams, stakeholder concerns, external influences, and business-relevant information can be allocated here but are not included in the SGAM Toolbox's diagrams.
In any case if want a little guidance of what to do in which viewpoint or layer, the SGAM Toolbox `Add-In Window` provides a short help for you: The contextual help text changes with the selected diagram type. The main content of the diagram, viewpoint, or the interlinking diagrams between viewpoints (so-called Model Transformations) is briefly described without going into too much detail.



5.3.1 SGAM Business Layer

Create a new diagram by RIGHT-Clicking on the `SGAM Business Layer` package and selecting `New Diagram`. Select from `SGAM Toolbox` in the list on the leftward side, choose the `SGAM Business Layer` diagram type and name it `Business Case Analysis`.



The diagram should automatically open and in the `Toolbox Window` you should see the elements available for the SGAM Business Layer. If not, you can select the appropriate toolbox with the needed modeling elements from the drop-down menu on the `Activate toolbox page...` button in the quick actions of the `SGAM Toolbox Add-In Window` in Figure 9.

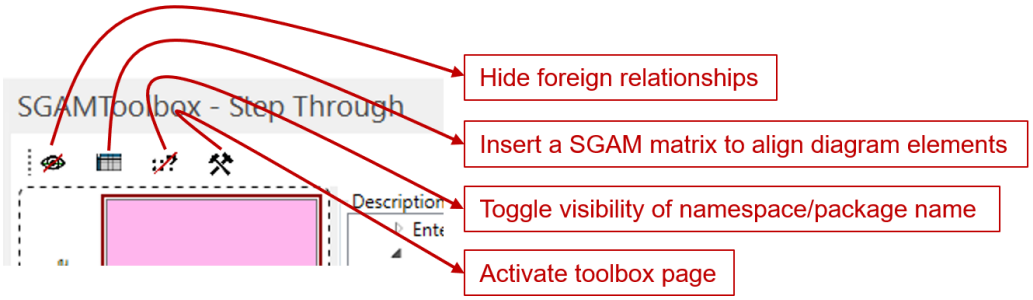
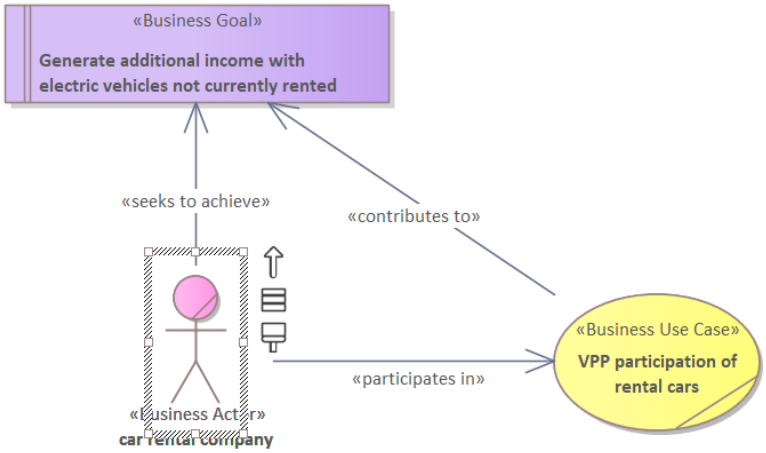


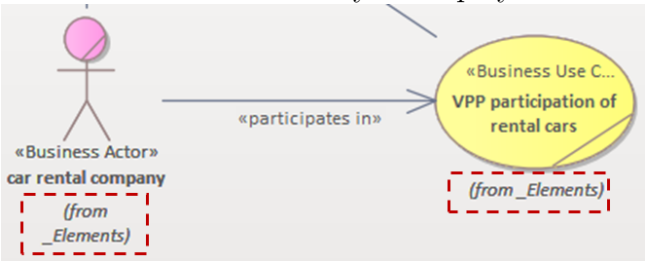
Figure 9: SGAM Toolbox quick actions

By LEFT-Clicking on the desired element in the toolbox, you can add it to the diagram by a second LEFT-Click. Create a `Business Actor` (Car rental company), a `Business Use Case` (VPP participation of rental cars), and a `Business Goal` (Generate additional income with EVs not currently rented) by placing them on the diagram and modifying their names by pressing F2 on the selected element.
To create the appropriate relations between the elements, you can select the desired relationship from the `Toolbox Window`. Create a `participates in` connector by clicking on it in the `Toolbox Window`, then click and hold on the `Business Actor` and drag the mouse to the `Business Use Case` element to finally release the mouse button. Pay attention to draw relation from the `Business Actor` as source to the `Business Use Case` as target as this is the only allowed direction in the metamodel.
A more convenient method to connect elements is to use the `Quick Link` feature: Click on an element that needs to be connected as source or target, then click and hold the white up-pointing arrow on the right upper side of the element and drag it to the second element. Then select the appropriate relationship from the context menu that appears. The correct direction of the relationship is automatically deduced and the relation is created.

By connecting all three elements with SGAM Business Layer relationships your resulting diagram should look like this:



If you want to add more information like a detailed description to a modeling element you can select it and write something in the `Notes Window` and is directly attached to the respective element. To tidy up our model, we recommend to place all modeling elements except the diagrams into the previously created `_Elements` package of the `SGAM Business Layer` package by drag-and-dropping them in the model browser. A helpful feature to locate elements displayed in a diagram in the model browser is to select it on the diagram and press Alt + G.
This moving of elements into remote packages introduces a note about the new package where the element resides now and is often not necessary to display on the diagram.



To quickly change this behavior we implemented a toggle button in the `SGAM Toolbox Add-In Window` to toggle the visibility of these in the quick actions of the SGAM Toolbox Add-In Window (see Figure 9).
If you want to model the full example of the Business Layer, please consult the online-browseable demonstrative model [3].

5.4 Logical Viewpoint – Modeling the high-level functionality and information exchange

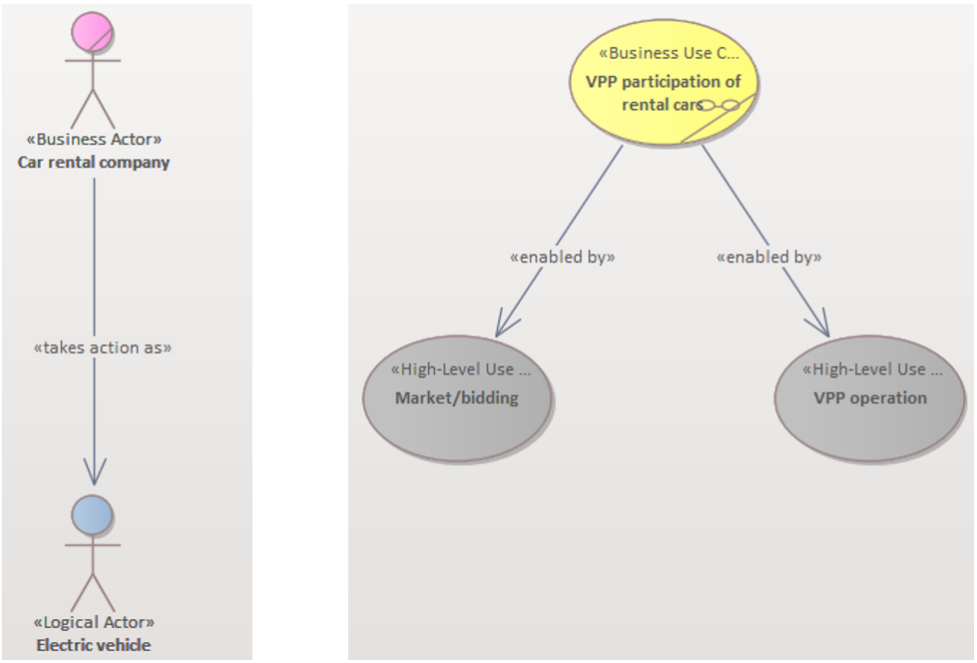
The Logical Viewpoint focuses on deriving high-level system functionality from business concepts without specifying implementation details. The `SGAM Function Layer` is used to model the high-level functionality of the system, which is derived from the business use cases. The `SGAM Information Layer` is used to model the information exchanged within defined actions between the logical actors in the system.

[3] https://dsse.at/sgam/downloads/v3.0.0.0/html-export-demo-parking-center-vpp/

5.4.1 SGAM Function Layer

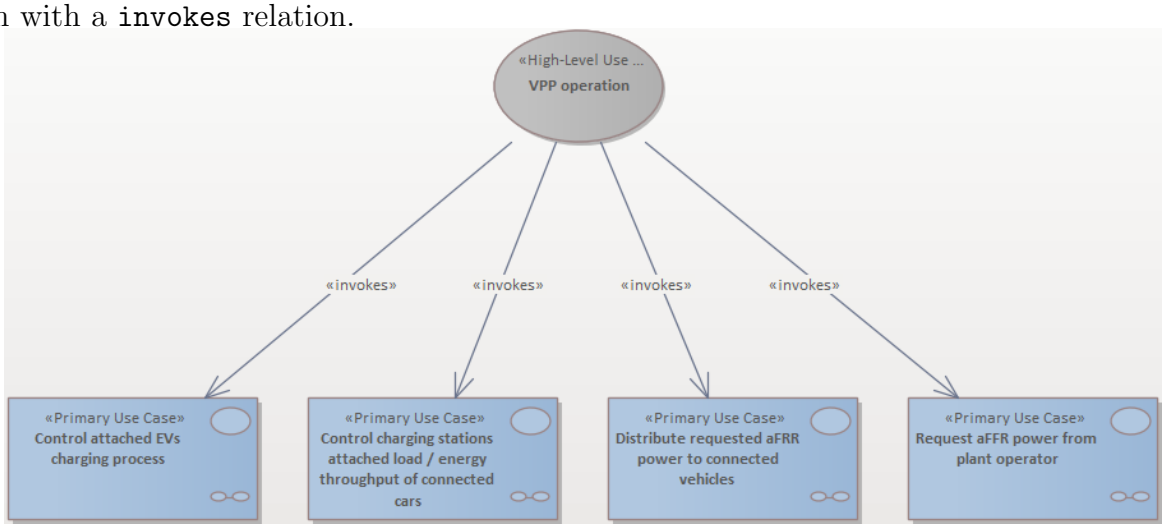
In the `SGAM Function Layer` we model the high-level functionality as `High-Level Use Cases` (HLUCs) of the system, which is derived from the business use cases. These HLUCs are further decomposed into `Primary Use Cases` (PUCs) that represent the main actions or processes within the system. `Logical Actors` process information, take actions, and interact with each other which is described in the PUCs.
At first, we need to make the model transformations from the `SGAM Business Layer` to the `SGAM Function Layer`.

Business Actors to Logical Actors *Business Actors to Logical Actors*



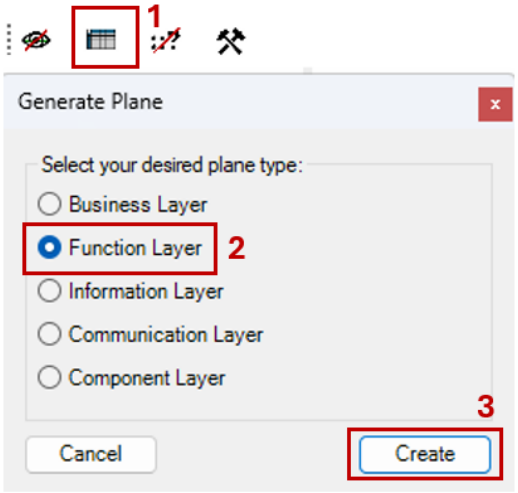
Create a new diagram of type `SGAM Function Layer` in the `_Model Transformation` package in the Logical Viewpoint and name it `Business Actors to Logical Actors`. Drag-and-drop all previously created `Business Actors` from the `SGAM Business Layer` package onto the diagram pane. Looking at our Business Actor for the car rental company, we want to depict that its owned and operated EVs are entities belonging to the car rental company interacting with other things in the system. Therefore, we create a `Logical Actor` called *Electric Vehicle* and connect it via the quick link feature to the `Business Actor` *Car rental company* with a `takes action as` relation. The other aspect, we need to transform to the Logical Viewpoint is the `Business Use Case` *VPP participation of rental cars*. Create another SGAM Function Layer diagram in the `_Model Transformation` package and name it `Business Use Cases to High-Level Use Cases`. Create two HLUCs: *Market Bidding* and *VPP operation* and connect them to the `Business Use Case` with a `enabled by` relation.
Not all elements are needed to be traced via such model transformations. But you should have good reasons to not do so: Many un-traced elements across viewpoints are an indicator that you may have missed some important entities. For instance, if you have many `Logical Actors` in the Logical Viewpoint but no related`Business Actors` considered, your project might be imposed to a risk of failure because you forget to include important stakeholders.
To refine the HLUCs to PUCs we create a new diagram of type `SGAM Function Layer` in the `SGAM Function Layer` and name it `High-Level Use Cases to Primary Use Cases`. As

depicted below we create the `Primary Use Cases` for the HLUC *VPP operation* and connect them with a `invokes` relation.

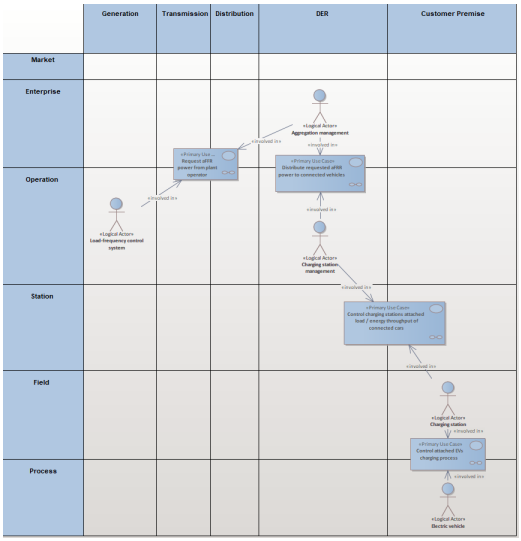


Because these PUCs are the entry points for creating more detailed descriptions of system functionality we can use UML activity, sequence diagrams or other appropriate means for your project to describe the detailed behavior of the PUCs. For this purpose RIGHT-Click on the `Primary Use Case` and select `New Child Diagram` and choose the appropriate diagram type. Examples are available in the demo-model online.

As a major feature of the SGAM framework to locate elements within a smart grid context we can use the SGAM Plane Matrix to allocate the PUCs to the appropriate zones and domains. To do so, create a new diagram of type `SGAM Function Layer` in the `SGAM Function Layer` package and name it `SGAM Function Layer Matrix`. In the `Toolbox Window` select the `SGAM Plane Matrix` and place it on the diagram pane. Then click on the table-like icon in the `Toolbox Window` generate the plane for the SGAM Function Layer on the currently opened diagram.

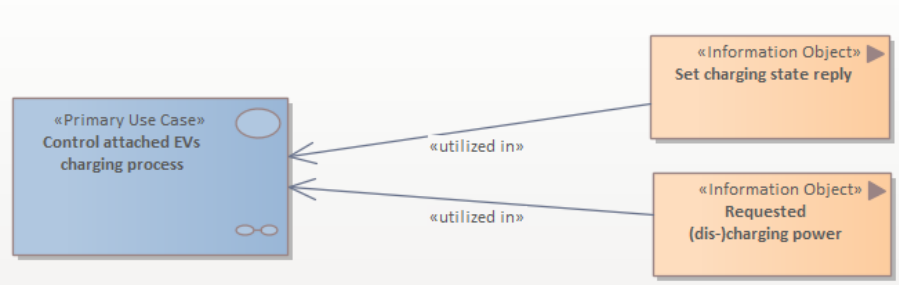


Position the `Primary Use Cases` and `Logical Actors` in the appropriate cells of the matrix. To add new LAs and relate them to the PUCs they are `involved in`, you can use the quick link feature. The resulting diagram should structurally look like this (view demo-model online to see more):



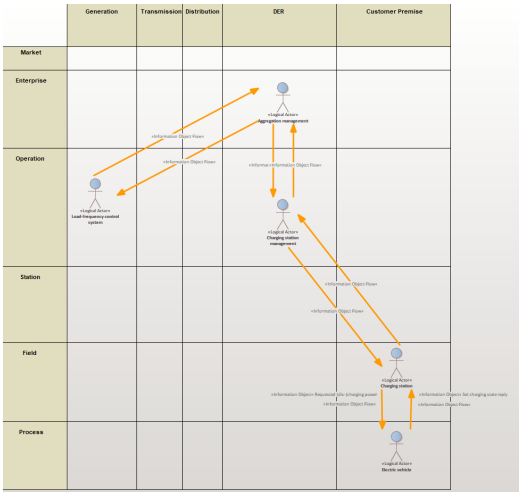
5.4.2 SGAM Information Layer

The `SGAM Information Layer` is used to model the information exchanged between the logical actors in the system. `Logical Actors` exchange `Information Objects` through `Information Object Flows` in the PUCs.
This information does not represent the actual data or messages sent within the running system but rather the semantics of the information exchanged. Imagine the difference by an illustrative example: You want to send gratulation to a remote-living friend on their birthday. The `Information Object` is therefore a *birthday gratulation* which you *somehow* want to convey to your friend using the `Information Object Flow`. Now you would have different options for realizing this. Following options would be depicted in the `Communication Layer` – *not* here in the `Information Layer`: You could send the text message *Happy Birthday!* (`Data Object`) via SMS `Data Object Flow`. You could also make a video call and sing a happy song to your friend. The informational content stays the same and therefore the way *how* you convey the information and how to represent it as data or message is not relevant in the `Information Layer` as it is technology-agnostic.
To model the information exchanged in the PUCs we create a new diagram of type `SGAM Information Layer` in the `SGAM Information Layer` package and name it `Information Objects utilized in Primary Use Cases`. Depict which `Information Objects` are needed by which `Primary Use Cases` by creating the `Information Object` elements and connecting them with a `utilized in` relation.



Then, create another `SGAM Information Layer` diagram in the `SGAM Information Layer` package and name it `SGAM Information Layer Matrix` and add a Information Layer Matrix to the diagram by using the toolbox function. Place all `Logical Actors` on the diagram

in their respective cells of the matrix. Then create `Information Object Flows` between the `Logical Actors`. By RIGHT-Clicking on a flow and selecting the option `Advanced... > Information Items Conveyed` you can specify the `Information Objects` they are exchanged by each flow. The resulting diagram should structurally look like this (view demo-model online to see more):

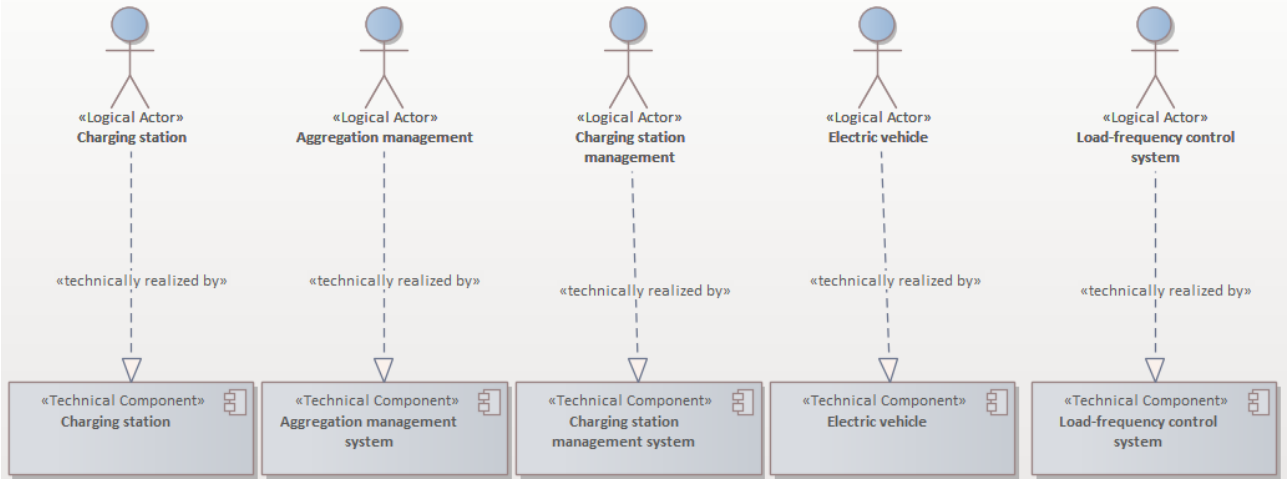


5.5 Technical Viewpoint – Modeling the physical or virtual realization of system components

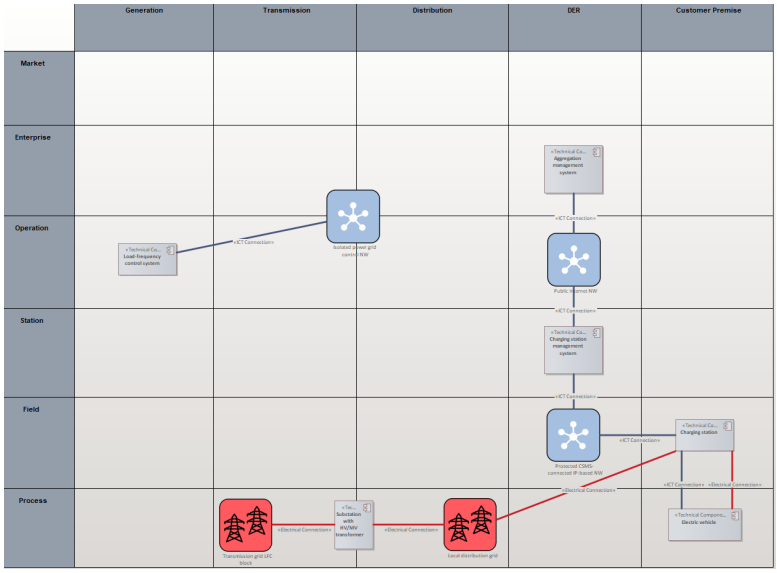
The Technical Viewpoint describes technology-specific system architecture that supports the implementation of the high-level functionality defined in the Logical Viewpoint. However, the focus is not on creating executable code or prototypes with diagram types of the SGAM Toolbox, but rather on defining the system architecture and its components. Of course, these ones *can* be linked to implementation artifacts such as source code or device specifications. In any case, the idea of the SGAM Component Layer is to focus on different types of entities and their physical interrelations – different instances or a bunch of deployed objects of them are typically *not* depicted here if it does not contribute to the understanding of the system architecture design.

5.5.1 SGAM Component Layer

In the `SGAM Component Layer` we model the physical or virtual entities responsible for executing system functionality and their physical interrelations. interrelations in the SGAM Component Layer mean we either need a connection to transport electrical energy (`Electric Connection`) or data signals (`ICT Connection`).
Similar to the Model Transformation from the Enterprise Viewpoint to the Logical Viewpoint, we need to create a Model Transformation from the Logical Viewpoint to the Technical Viewpoint. Create the respective diagram of the type `SGAM Component Layer` in the `_Model Transformation` package of the Technical Viewpoint and name it `Logical Actors to Technical Components`. `Logical Actors` are `technically realized by` `Technical Components` (TCs). An LA can be realized by multiple TCs, a TC can realize multiple LAs.

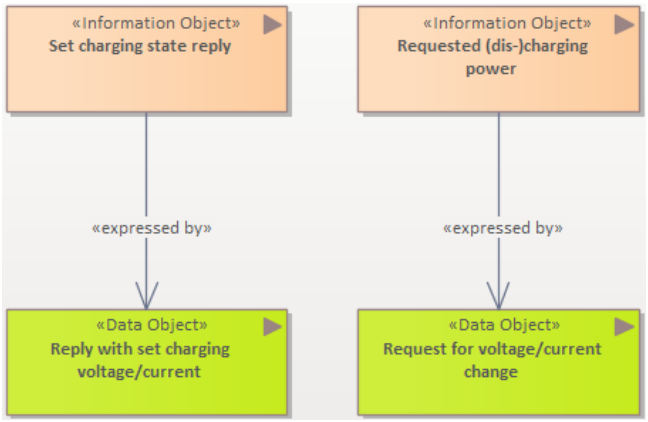


To depict the architectural design related to types of components and their connections within the system, we create a new diagram of type `SGAM Component Layer` in the `SGAM Component Layer` package and name it `Component Layer Matrix`. To transport electrical energy within a rather *similar* area, we take the `Electrical Network` element from the toolbox and place it on the diagram. Indicators for *similarity* are for example that the components are located in the same room, building, or even on the same device. Other criteria are their position in the SGAM plane like a transmission grid in the Transmission Domain or a distribution grid in the Distribution Domain. In any case, within such a network, the frequency and voltage levels are the same, except possible local fluctuations. Connect the network-element with the respective `Technical Components` with an `Electric Connection`.
The same process is done for the `ICT connections` that interconnect `Technical Components` over `ICT Networks` to allow for data exchange. The internal structure of (like physical cabling, router or switch devices) within an `ICT Network` is not modeled with the provided elements of the SGAM Toolbox. If it is necessary for your project to model this, you can link the respective artifact or diagram to the `ICT Network` element by RIGHT-Clicking on the element and selecting `New Child Diagram` like the same process for detailing out PUCs in the Function Layer.
Both mentioned network modeling elements are optional but recommended to be used. A criteria for neglection is that the system is so small that it does not make sense to model the networks, e.g. an internal representation of a single device or an electrical connection between an EV and a charging station via a charging cable.
The resulting diagram should structurally look like this (view demo-model online to see more):



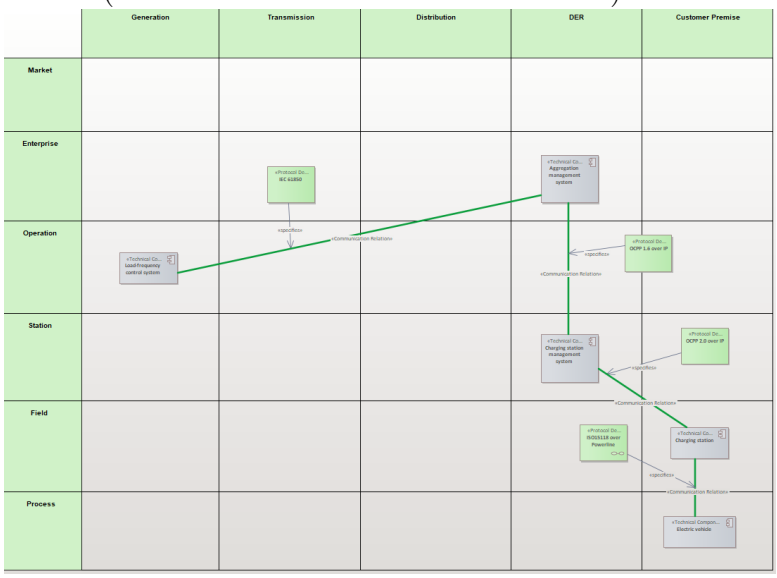
5.5.2 SGAM Communication Layer

In the `SGAM Communication Layer` we model the communication paths and protocols between the components defined in the `SGAM Component Layer`. Additionally, we can define the data structures and protocols used for communicating previously defined information using primarily ICT (Information and Communication Technology) infrastructure.
Make the Model Transformation from the Logical Viewpoint to the Technical Viewpoint for the concern of expressing `Information Objects` as `Data Objects` by creating a `SGAM Communication Layer` diagram in the `_Model Transformation` package of the Technical Viewpoint and name it `Information Objects to Data Objects`. Trace the `Information Objects` from the `SGAM Information Layer` to the `Data Objects` in the `SGAM Communication Layer` by creating `expressed by` relations.

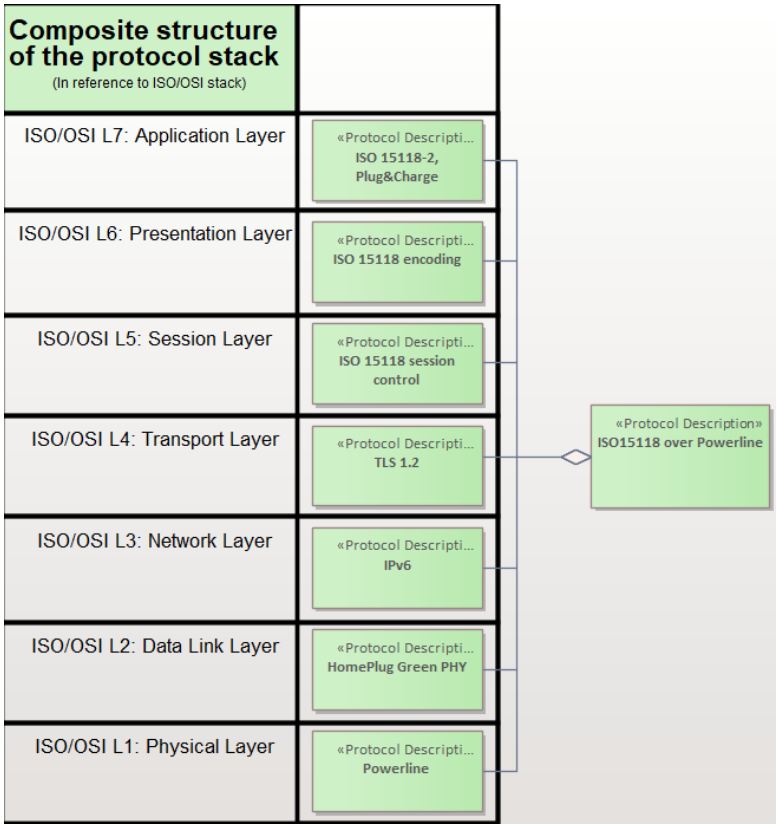


These `Data Objects` could be of proprietary nature or prototypical data structures but in most times they are defined by standards. To document these conformant schemes we create another `SGAM Communication Layer` diagram beneath the `SGAM Communication Layer` package and name it `Data Representations define Data Objects`. Insert the `Data Representation` element from the toolbox, name it according to the applied standard data scheme and connect it with the `Data Object` with a `defines` relation as depicted below:

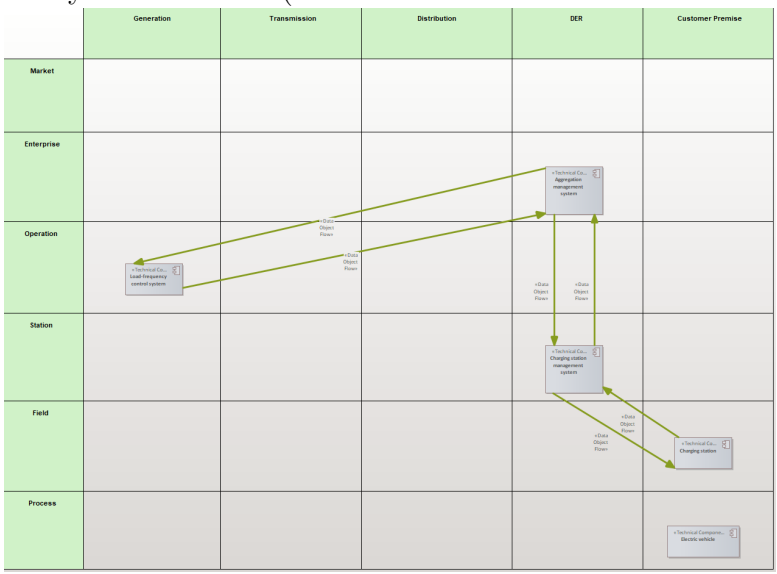
We already defined the `ICT Connections` in the `SGAM Component Layer` diagram that relate to physical interconnections. The actual underlying communication between these components need more than a connection; a stack of protocols is required to be able to exchange data that both sides understand. Therefore, we create a new diagram of type `SGAM Communication Layer` in the `SGAM Communication Layer` package and name it `Communication Layer Matrix: Communication Relations`. Place the `Technical Components` that take part on exchanging data in the respective cells of the matrix. Then create `Communication Relations` between them. Of course, a `Communication Relation` can only exist if there is an uninterrupted path of `ICT Connections` and intermediary components between them. The resulting diagram should structurally look like this (view demo-model online to see more):



To detail the communication relation, a `Protocol Description` can be created and connected to the `Communication Relation` with a `specifies` relation. As outlined in the paper there is a common source of confusion of *what is THE protocol used?* – is it ethernet, TCP/IP, HTTP, MQTT, or something else? The `Protocol Description` can be further detailed by creating a new child diagram of type `Composite Structure`. In this diagram, the overall *protocol stack* can further be decomposed by e.g. in reference to the ISO/OSI model. Aggregating multiple `Protocol Descriptions` to the one specifying the communication relation helps to clarify the confusion about what the protocol means or consists of.



Last point in the `SGAM Communication Layer` is to define the `Data Object Flows` that are used to exchange the `Data Objects` between the `Technical Components`. Similar to the creation of the `Information Object Flows` in the `SGAM Information Layer`, we create a new diagram of type `SGAM Communication Layer` in the `SGAM Communication Layer` package and name it `SGAM Communication Layer Matrix: Data Object Flows`. Put the `Technical Components` in the respective cells of the matrix and create `Data Object Flows` between them. The `Data Object Flows` are linked to the `Data Objects` they convey by RIGHT-Clicking on the flow and selecting `Advanced... > Information Items Conveyed`. The resulting diagram should structurally look like this (view demo-model online to see more):



5.6 SGAM Toolbox – Tips and Tricks

This section lists few tips and tricks to make your modeling experience with the SGAM Toolbox more convenient.

- **Model Transformations:** Use the Model Transformation diagrams to trace elements across viewpoints and layers. This helps to maintain consistency and traceability in your model while keeping a clutter-free workspace.
- **Different Packages for Diagrams and Modeling Elements:** Keep your diagrams and modeling elements in separate packages. This helps to keep your model organized and makes it easier to navigate.
- **No Modeling Without Purpose:** Always have a clear purpose for each modeling element you create. Avoid creating elements that do not contribute to the understanding of the system architecture or its components.
- **Delete Orphan Elements:** Delete elements that are not connected to any other elements in the model or connect them to the appropriate elements if links are missing. You can search for orphan elements the Find in Project Window (Ctrl + F), select the option Diagram Searches in the middle drop-down menu, and then select the option `Find Orphans` on the right drop-down menu.
- **Quick Linker:** Use the Quick Link feature to connect elements quickly and easily. This saves time and helps to maintain the correct direction of relationships.
- **Notes Window:** Use the Notes Window to add additional information to your modeling elements. This helps to keep your model organized and provides context for each element.
- **Traceability Window:** Use the Traceability Window from Enterprise Architect to view the relationships between elements in your model. This helps to understand the dependencies and connections between different elements and is essential to maintaining a coherent model. Also, sometimes it can be helpful to create additional diagrams that focus solely on the illustration of the traceability of particular elements.
- **Namespace Visibility:** Use the toggle button in the SGAM Toolbox Add-In Window to control the visibility of namespaces in your diagrams. This helps to keep your diagrams clean and focused on the relevant elements.
- **Hiding Foreign Relationships:** Use the toggle button in the SGAM Toolbox Add-In Window to hide foreign relationships in your diagrams. This is useful when you display the same modeling element but need to have different relations in focus. With this function you can for example hide all `Information Object Flows` between `Logical Actors` in the `SGAM Function Layer` diagram with one simple click.
- **Using Reference Models:** You can use reference models like the ENTSO-E Harmonized Role Model for electricity by importing the freely available XML file into your project. By creating a generalization relation between your `Business Actor` and the reference model's `Role` you can inherit the properties of the reference model by the SGAM Toolbox function *Generalize SGAM Element Options*. Other reference models are also available like the NIST Logical Reference Model for Smart Grid that is worth to mention here.

- **Syntax Checker:** To avoid producing explicitly by the metamodel forbidden relationships, keep the option `Strict Connector Syntax` enabled in the modeling environment Enterprise Architect. However, if you know what you do and want to freely model, you can disable this option in the menu bar `Start>[Appearance]>Preferences>Links>Strict Connector Syntax`.

6 Conclusion

By providing this Documentation we hope to help you to get a rough idea of the Smart Grid Architecture Framework and the SGAM Toolbox to get you started to model your smart grid architectures in a structured and coherent way. Of course, even in this longer guide than the original paper of the revision of the SGAM Toolbox to version 3 we could not cover all aspects of the framework and the toolbox.
As mentioned many times before, the primary supplementary materials to this documentation are:

- **Scientific Background:** The open-access scientific paper for SGAM Toolbox v3
- **Demonstrative Model:** The online-browseable demonstrative model
- **Language Specification:** The online-browseable MOF-based metamodel

If you have any questions, suggestions, or feedback, please do not hesitate to contact us via the contact form on our website!