EMPOWERING THE SYSTEMS ENGINEER - IEC 61850 AND VISUAL LANGUAGES

Rogério PAULO, Efacec – Portugal, rdpaulo@efacec.com
Mário LEMOS, EDP – Portugal, mario.lemos@edp.pt
António CARRAPATOSO, Efacec – Portugal, amc@efacec.com
Carlos PEÇAS LOPES, EDP – Portugal, carlos.pecaslopes@edp.pt
Rui BERNARDO, EDP – Portugal, rui.bernardo@edp.pt

ABSTRACT

In this paper we propose a contribution to the realization of a model-driven systems engineering vision: an initial analysis towards the design of a unified modelling language built upon IEC 61850.
To efficiently profit from the advantages and manage the flexibility and complexity of full IEC 61850 systems, engineers need not only to exchange models of the system, but also to straightforwardly understand them during the whole system life-cycle. For this purpose a standard exchange format is required (ex: IEC 61850-6 SCL), but a standards-based visual language would constitute a unique industry enabler by increasing model understandability for the user from design, through implementation and testing, up to operation and maintenance.
Throughout the paper authors present proposals of diagrammatical notations and corresponding use cases as well as introduce an on-going substation application.

INTRODUCTION

As IEC 61850 establishes its first maturity level and the first major update regarding engineering (IEC 61850-6 Ed. 2[1]) is released and reaching wider industry application, the issue of engineering methods and tools remains pointed out as one of the yet to be fully tackled domains of power systems automation.
As we steadily move from a wired world to a software and communications world we will be able, in a near future, to expand the application of IEDs and IEC 61850 communications from the station bus to the process bus, power generation and wide-area applications from transmission to DA and DER, effectively establishing a common real-time automation infrastructure across the power system based on cooperative and interoperable agents. Many required standards to operationally support this seamless smart grid real-time infrastructure are already available. However, authors agree [10] that designing, implementing, operating and maintaining such a wide and complex virtual infrastructure with less people and less time will lead to significant engineering costs if adequate support for systems engineers is not provided.
On this topic users have raised concerns regarding tool quality and availability, use of the single system tool and engineering-level interoperability. This touches only the tip of the iceberg and new approaches are required to better support the activities of the different actors and roles [2]. Future systems engineering includes specification, design, implementation, functional testing as well as configuration, condition, performance and maintenance management in an increasingly complex technological environment that combines real-time with non-operational applications, permanent and dynamic associations, static and dynamic engineering, as well as protection, automation and control (PAC) with management applications.

People, Processes Tools, Models and Systems

At the core of every mature engineering discipline, from civil engineering to software, lie common processes and methodologies that, together with common languages, enable practitioners to collaborate and leverage knowledge. Power systems automation is not different and engineers need to unambiguously define and communicate models that can be handled/ transformed by tools (fig. 1).

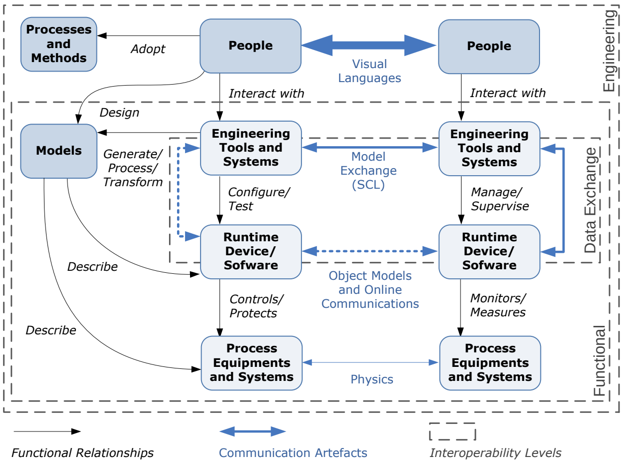

Fig. 1. Users, tools, models, systems and their relationships.

In this paper we focus on proposals towards the definition of a unified automation systems modelling language. This domain-specific visual language (DSL/DSVL) would allow engineers to adopt model-driven approaches and handle complex distributed automation system models with different levels of granularity (detail) and completeness, suitable for describing specifications, executable functional models or system documentation. Complexity is handled through the use of multiple coherent models of different artefacts such as (i) functions, devices, distributed applications, system architectures, and (ii) reusable sub-system designs, libraries or templates, etc.

Advantages of Model-Driven Engineering (MDE)

In a model-driven approach a repository of the whole consistent (formal) system model is built iteratively during the project and system life-cycle. Models are visualized/ edited/ animated through partial diagrammatic views, instead of unstructured and ambiguous sets of documents.
MDE, compared to document-driven approaches, improves quality and productivity, this in turn leading to enhanced system reliability and maintainability.

With enough semantic expressiveness, models and diagrams can also be directly used for (semi-)automated: (i) generation of PAC schemes and settings (ex: topology-based schemes like interlocking, self-healing, shedding, etc.), (ii) generation of communication parameters (ex: datasets, multicast configuration), (iii) simulation for testing, (iv) validation (static, functional and performance), (v) plug-and-play online monitoring for testing, management or commissioning, (vi) device configuration and deployment (portable across different vendor's device runtimes through specific model transformations), (vii) database generation (HMI, alarm, DMS/EMS, CMD/AMS) as well as (viii) documentation generation. Through MDE, systems engineering, performed mostly manually today, can evolve into a highly automated activity.

UNIFIED AUTOMATION LANGUAGE

The proposed approach to define a unified automation systems language (UAL) would build on "reverse modelling" and extension of the current IEC 61850 standard series to establish a single meta-model (the language abstract syntax or M2 model in MOF[2] terms) for the power systems engineering DSL. Language-dependent artefacts, such as editors or serialization formats (SCL or its future revisions), can be automatically generated from the meta-model. An actual system would be described as a UAL-conformant model (M1-level model).

The meta-modelling elements of the abstract syntax are interrelated and correspond to different layers, including, among others: (i) the process/ plant, (ii) the physical system including PAC, communication devices and links, (iii) communication data flows, (iv) distributed function allocation, (v) signal interaction between functions and (vi) behaviour definition.

The second fundamental component of UAL is a common diagrammatical notation or visual syntax. Diagram symbols are organized in purpose-oriented standard diagram types, but mixed diagrams are allowed (ex: associating communication networks with plant structure). Examples of such notations are loosely defined and employed in several IEC 61850-related papers, publications and toolsets. There are also a number of existing industry work, reports and standards that can provide relevant contributions: CIGRE B5.39, IEC 61131, IEC 61499[4], IEC 60617, SysML[5], IEC 61850-90-4[6] and IEC 61850-90-11[7], IEC 61850 UML Modelling, to name a few.

It should be noted that establishing UAL would require enhancements to IEC 61850 together with the specification of a visual notation, the missing link between technology and people. A similar approach led to UML[8].

Advantages of the Language Definition Approach

As opposed to adopting a general purpose language such as UML, or even SysML, a DSL approach (defined either as a UML profile/extension or MOF-based meta-model) would allow solutions to be expressed in the common terms of power systems automation engineering.

From a standardization standpoint this would also foster formal, explicit and non-fragmented definition of the currently implicit IEC 61850 meta-model.

Visual Languages and Standardization

It must be understood that a notation's primary goal is to describe model artefacts for user reasoning in an intelligible way. Contrary to the current assumption of IEC 61850, visual notations should not be under the scope of vendor tools but under the scope of international standardization as the multiple notation and semantics of proprietary languages will negatively impact "user-level interoperability". Such a language would nevertheless be implemented in different tools, serving different purposes, from a specification and design tool dealing with static engineering for VHV/HV substations, a commissioning tool for online system verification or a management tool dealing with dynamic engineering in MV/LV smart grid plug-and-play applications.

LANGUAGE OVERVIEW

Common Elements

The adoption of a common basic value type system (primitive types, structures, enumerations, arrays, etc.), including syntactic rules for defining both derived types and expressing data type literals, used throughout the language and its models, is fundamental to define data points, model attributes and properties, logics, etc.

The definition of common object naming, identification and meta-association rules is most important to enable uniform model navigation and traversal mechanisms.

Common hierarchical modelling constructs, including dependencies, enable consistent organization of different model objects. Diagrammatically these may be represented in Package Diagrams (PD), equivalent to the UML counterpart, or by object containment regions (boundaries) in other structural diagrams.

Another fundamental construct, which is missing in IEC 61850, is the type-instance construct. Such concept is most useful for modelling functional units such as logical nodes, but is also applicable to other elements, such as connectivity-enabled primary process equipment objects.

Structural Diagrams

Primary Process Diagrams (PPD)

Establishing the primary or controlled process structure (fig. 2) is fundamental for specification purposes and all subsequent engineering steps. SCL includes a substation section definition that is not generalizable to all power system components (also considering non-electrical processes). Its meta-model should be revised to include flexible object types with a generic connectivity model that can support libraries of standard equipment types, their subcomponents and characteristic attributes as well as applicable notation. This would allow straightforward extension of the equipment classes much like the standard logical nodes classes. Since the architecture of the secondary system is intimately tied to the primary process, defining relationships (containment, control, monitoring, virtualization, etc.) between primary equipment, hierarchy and topology to devices, networks and functions is also required.

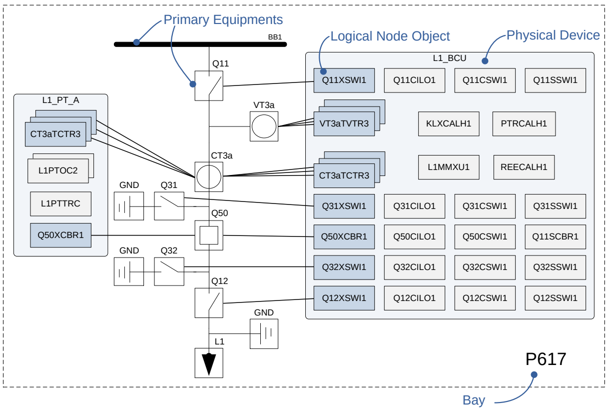

Fig. 2. Conventional PPD example portraying single line diagram and logical object associations.

Physical System Diagrams (PSD)
PSD (fig. 3) are employed to describe the system physical architecture, comprehending devices and physical associations, either communication networks, or conventional wiring. Through symbols and decorators PSD notation allows the description of different L1/L2 network topologies, including redundancy (PRP/HSR/RSTP), as well as different media, device types (end nodes, switches, routers, clocks, etc.) and addressing. IEC 61850-90-4 [6] proposes a notation for Ethernet LAN, but enhancements (and eventually other diagram types) would need to be considered for portraying other networking technologies (including WAN technologies employed in smart grid applications) and logical communication structures such as VLANs, multicast domains, priority handling or L3 architectures (IP, VPN, etc.).

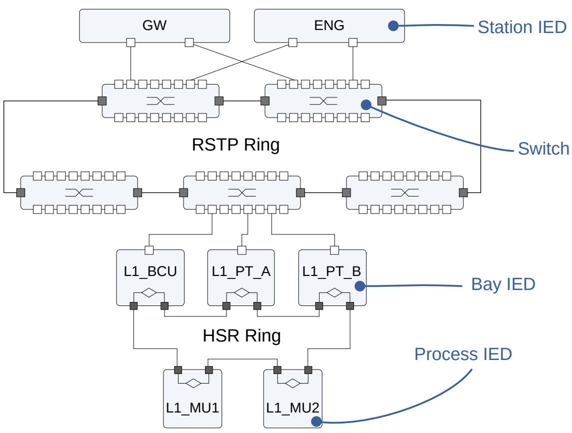

Fig. 3. PSD example (notation adapted from [6])

Data Flow Diagrams (DFD)
The physical architecture is not sufficient to describe the logical interactions between IEDs. The relevant data flows (client/server, GOOSE, SV, FTP or other) between devices (or logical devices) comprehend a higher level logical structure that must be understood and may be represented by the DFD diagram type (fig. 4). In [9] an alternative notation for IEC 61850 service associations is proposed.

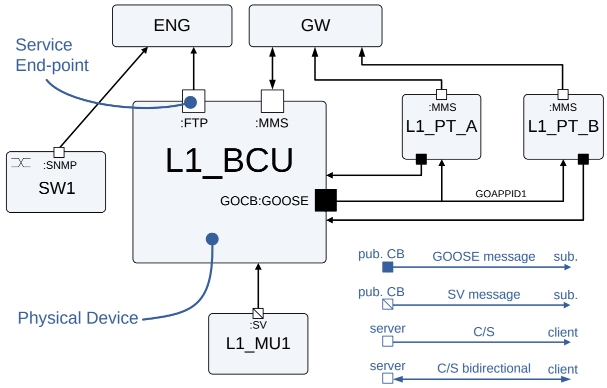

Fig. 4. Communication dataflow diagram notation.

Logical Node Functional Diagrams (LNFD)
IEC 61850 mainly describes interface instances that conform to a set of non-instanciable logical node classes with standardized semantics (LN Domain Classes). While logical node instances (LN Objects) are suitable for information modelling in a communication-only perspective, they are not suitable for distributed systems modelling. LN should evolve to (fig. 5): (i) include type-instance constructs (LN Class-LN Object) that would enable reuse of specification and implementation through instanciable class libraries, (ii) include behaviour and data encapsulation within LN Classes, and, (iii) incorporate function block concepts (similar to SysML or IEC 61499 blocks), extending data objects to typed input/output ports (representing either data values or streams of samples).

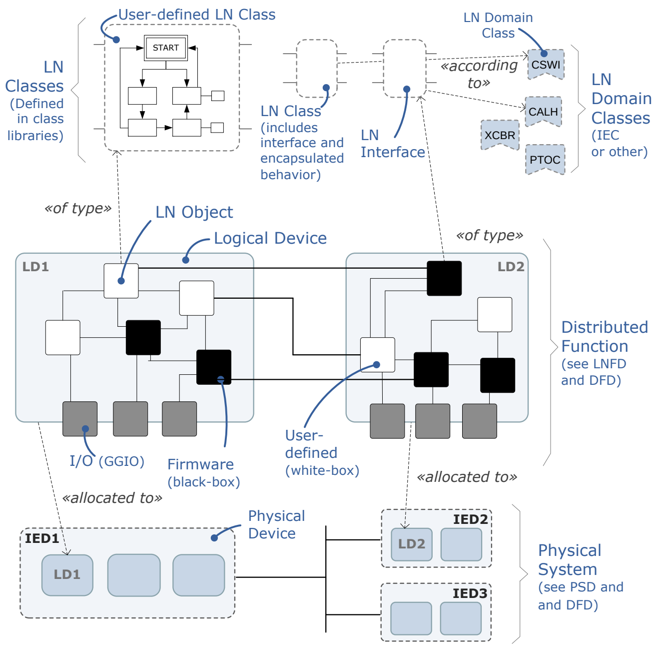

Fig. 5. Extending LN to typed distributed function blocks.

Logical node functional diagrams (fig. 6) comprehend networks of LN Objects interconnected via I/O links (mapped to GOOSE, SV, C/S, hardwired or internal device data flows, depending on LN Object to Device allocation). Functional diagrams serve the purpose of representing the actual signal interactions between LN Objects of a given distributed application/ function.

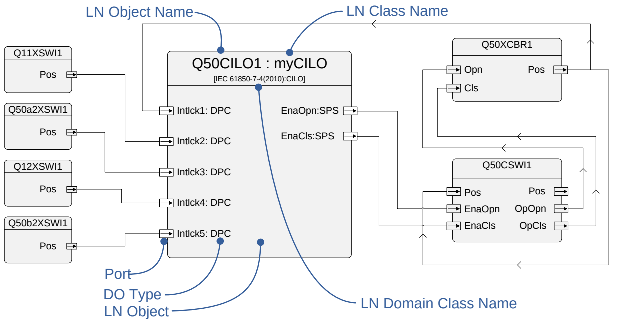

Fig. 6. LNFD portraying object allocation and signal flow for a given distributed application (ex: interlocking).

Incorporating other higher order functional modelling constructs is not straightforward but can also be envisioned, namely (i) composite LN Class definitions (from IEC 61499), (ii) LN Class composition or reference associations (from SysML) and/or (iii) applications and sub-applications (from IEC 61499). This would enable the explicit definition of given distributed functions (interlocking, Volt/VAr control, shedding and restoration, and switching schemes are examples of distributed functions that coexist on a given system through cooperating shared LN Objects) and solve the granularity issue of current LN Domain Classes (in today's systems a given function is frequently implemented with multiple logical node instances conforming to given domain classes that are technologically related and indivisible).

Dynamic Diagrams

For the purpose of defining behaviour (executable semantics) dynamic diagrams are employed, describing the logic encapsulated by LN Classes. IEC 61131-3 state charts (SFC), block diagrams (FBD) or structured text (ST) as well as IEC 61499 execution control charts (ECC) are examples of existing languages that can be integrated in UAL for this purpose. Current work in progress in IEC TC 57 is targeting the description of logics in IEC 61850[7]. In case of firmware-specific LN Classes, a typical case being modelling of protection functions, LN Classes may be provided only as interface classes (black-box).
Once the concept of LN Class is fully established inheritance and polymorphism of object-orientation theory may also be considered.

AN APPLICATION USE CASE

Current Substation Automation Systems (SAS) solutions used in EDP Distribuição are based on distributed architectures, supported by IEC 61850, providing a large number of benefits in addition to traditional PAC features. However, as complexity increases and given that systems documentation is still not assured in a consistent and integral way, comprehending and managing the SAS during the entire life-cycle is not an easy task, requiring a significant amount of effort and expertise [11].
Based on EDP's SAS requirements, leveraging IEC 61850 and exploring new generation of Efacec's engineering toolset, the authors will explore the application of a model-driven approach supported in diagrammatical notations, to establish SAS specification, particularly the advanced control and automation functions.
The authors expect to achieve higher understandability, reduced ambiguity and engineering interface simplicity, hence increasing engineering performance, as well as to learn and anticipate the potential benefits and constraints of this approach to the whole SAS life-cycle.

CONCLUSIONS AND FURTHER WORK

To support engineering throughout the system lifecycle, from specification to asset management, other elements are required but not discussed in this paper. These would include process and user interface, secondary equipment component models but also formal description of functional requirements (including reliability and performance). A modelling language must also be proven suitable for use in the scope of different organization models and processes including both top-down or bottom-up methodologies.
In conclusion, establishing a unified power systems automation modelling language would bring significant benefits to the industry and is viable through the use of the existing systems engineering body of knowledge, but for it to be successful it must be the result of a meeting point of stakeholders, methodologies and technology - it should be industry-driven and not vendor-driven. The authors hence submit this contribution as an input to the future development of an internationally standardized language.

REFERENCES

[1] IEC 61850-6, 2009 "Configuration description language for communication in electrical substations related to IEDs"
[2] R. Paulo, 2011, "IEC 61850 Engineering Tools and Processes: Increasing the Maturity Level", CIGRE SC B5 Colloquium
[3] OMG, 2011 "Meta Object Facility (MOF)", v2.4.1
[4] IEC 61499 all parts, 2012, "Function blocks"
[5] OMG, 2012, "Systems Modeling Language (SysML)", v1.3
[6] IEC 61850-90-4 DTR, 2012 "Network engineering guidelines for substations"
[7] IEC 61850-90-11, WIP TR, "Methodologies for modelling logics for IEC 61850-based applications"
[8] OMG, 2011, "Unified Modeling Language (UML)", v2.4.1
[9] F. Visser et. al., 2011, "Graphical Specification for IEC 61850 Based Substation Automation Systems", CIRED *2011*
[10] L. Hossenlop et. al., 2011 "Toward an Auto-Configuration Process Leveraging the IEC 61850 Standard", CIRED 2011
[11] M. Lemos et. al., 2011 "Substation Automation Systems current challenges and future requirements – the InPACT project perspective", CIRED 2011