MODEL DRIVEN DIAGRAM AUTOMATIC GENERATION

Carlos Mota Pinto
EDP Distribuição – Portugal
carlos.motapinto@edp.pt

António LEITÃO
EDP Distribuição – Portugal
antonio.leitao@edp.pt

Fernando Miguel SOUSA
EDP Distribuição – Portugal
fernandomiguel.sousa@edp.pt

Pedro Manuel SILVA
Efacec – Portugal
pmsilva@efacec.com

ABSTRACT

The paper presents a new solution for SCADA/DMS electrical feeder network schematics and automation synoptics based on a model-driven approach, allowing for an automatic and on the fly diagram generation replacing the traditional manual drawing and manual dynamic behaviour configuration for each diagram instance.

INTRODUCTION

In EDP Distribuição (EDP Group), Portugal, Distribution System Operator (DSO), with more than 6 million customers, over 400 HV/MV Substations, 60 thousand MV/LV Substations, 80 thousand km Network (HV/MV) and a LV Network about 140 thousand km, with the increase of remote controlled circuit breakers in overhead MV lines and MV substations carried out by in the past years, the actualization of the associated representation and data in traditional SCADA diagrams became a problem namely with the inherent increase in human resources, time and cost required to maintain diagrams and recurrent delays between field electric network construction and system configuration update with consequent restrictions in network operations.

There was also the need for a normalized representation of automation information related to all HV/MV substations. Each HV/MV substation, alongside the one-line diagram, has several associated synoptics related with automation functions that are somehow repetitive in nature and the overall number of this type of synoptics exceeds 2000. These considerations were the main drive to an imperative change the way synoptics and feeder schematics were designed and generated.

Considering that the diagrams actualization was one of the most time consumption tasks, and that in the major cases synoptics are repetitive in terms of the information they present, in order to improve the quality, normalization, human errors and also the quickness of the SCADA system actualizations, something different from the traditional way of drawing SCADA diagrams should be introduced.

METHODS

There were some main objectives to accomplish: normalization of data and diagrams layout, ease of access and fast actualization of data representation, in order to achieve an update of the changes in synoptics and schematics layout in nearly real time. The challenge was presented to EFACEC, the SCADA/DMS system manufacturer used by EDP Distribuição.

The solution was based on a model driven diagram automatic generation approach. The legacy and the generational diversity of the SCADA database were additional challenges, since it was necessary to normalize all the data points by the same pattern.

RESULTS

Feeder schematic diagrams

For the MV electrical feeder network schematics, the solution is a linearization of the geo-referenced electric network (Image 1), beginning at the MV circuit breaker of a HV/MV substation, representing the state and other relevant telemetered data of remote controlled points and ending in the sectionalizers that are normally opened (Image 2).

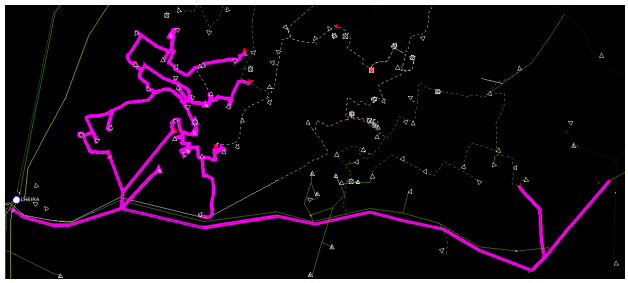

Image 1

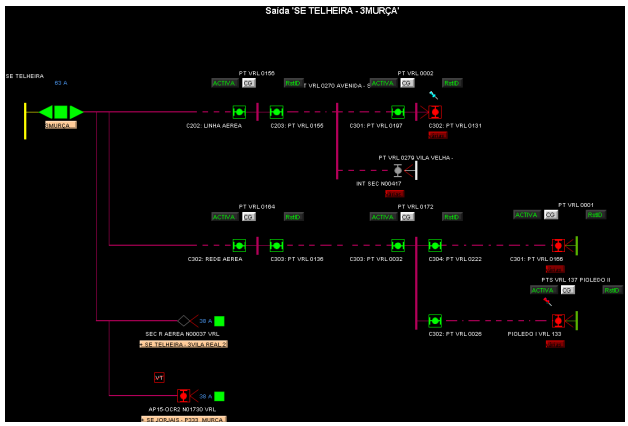

Image 2

There are links to the adjacent feeders facilitating user navigation and the dynamic behaviour is identical to other diagrams such as telemetry data, control actions, colouring, tagging, etc.

These diagrams can be called up/generated from any equipment referred in the system (e.g. other diagrams, lists

of alarms, lists of abnormal conditions, power applications' results, etc) thus being readily available to help operators solving problems. It is assumed that operations that require complete diagrammatic information for the feeder are performed with the assistance of other diagrams, like geographic, schematic or installation synoptic diagrams, which are manually built. The integration between all these specialised diagrams is achieved with tools like selecting an equipment in a diagram which results in automatically navigating in other diagrams in order to centre them around the representations of the selected equipment.

Instead of trying to represent all feeder equipment, the automatically generated feeder schematic diagram focuses on representing only the most important information concerning outage resolution – the rules for generating these schematics can be summarised as following:
- Scope: the equipments represented belong to the feeder's current configuration joined with its normal configuration;
- Filtering: only telecontrolled switches and manually controlled switches that are currently open are represented in the diagram. In addition, dynamic topological changes made to the model, like Shunts and Open Points are also represented. All line segments joining these equipment are represented, which means that colouring and any condition set on these equipment will also be displayed on these diagrams.

These diagrams are generated on the fly and they represent the network taking in account the as built configuration data and the state of the network, and if this data / state changes, the feeder schematics will be generated accordingly reflecting consistency with the model. The on the fly generation, the automatic reconfiguration of the schematics and the integration between the GIS and SCADA information are distinctive aspects from other implementations that intended to build schematics.

The solution

A feeder schematic diagram is generated in two steps:
- Compute the tree structure: an equipment is supplied to the topology processor, which finds the corresponding circuit breaker and then returns a tree structure obtained applying the rules described above.

Note that a feeder is in general described by a graph structure, not by a tree. However our approach was a linearization of this graph into a tree, which means that a joining node is repeated in multiple end-of-branches.
- Generate the diagram: a set of configured presentation rules is used to transform the tree structure into a diagram. These rules include:
  - A generic definition of the SCADA entities that should be represented near each switch type.
  - Graphical symbols and their dynamic behaviour used to represent switches, lines and all scada signalizations.
  - Commands to be offered by each symbol
  - Geometry and other graphical aspect details

The result is a normal diagram that is ready to be drawn by the system's diagram visualiser, thus being completely integrated in the system and offering the normal functionality.

Pattern diagrams

Feeder schematic diagrams are all generated according to the same set of rules, which typically are not expected to change over time. On the opposite side, there are multiple different types of synoptic diagrams showing automation information related to HV/MV substations. For example, there are synoptics focused on voltage regulation, others aiming on displaying loads, etc. These diagrams are also likely to change in time, with new types of diagrams being created or new telemetry signals being available and displayed. Another relevant difference is that the information structure required by automation synoptics is in general simpler than that required by feeder diagrams – instead of being topology based, automation substation diagrams show information that can be obtained querying for scada entities matching a name/tag pattern for a given hierarchy root.

A consequence of the above exposed is that this problem required a solution that could be easily configured and modified – the diagram and symbology editor has been found to be the right place to support this need.
The solution implemented in the editor is based on the following principles:
- A symbol is a set of graphics that normally have a dynamic behaviour depending on attributes of an equipment. These attributes could be telemetry data and status, model data (coming from SCADA or GIS databases), power applications calculations, abnormal conditions (tagging), etc.
- A diagram is composed by placements of symbols, each placement being associated with an "equipment" (an "equipment" could be a network or telecommunications equipment, an installation, a region, a topological island, etc)
- A symbol can be composed by placements of other symbols. In this case, since the symbol definition is generic, each sub-symbol cannot be associated with concrete equipment but instead it must be associated with a function that accepts an anchor equipment and returns another equipment (possibly the same anchor equipment) – function examples: the substation owning a given bay; the feeder breaker of a given bay; the transmission transformer feeding a given distribution transformer.
- A symbol container is a special object that repeats a (compound) symbol for each "equipment" returned by a function. Examples of these functions: all outfeed bays of a given substation; all major substations of a given area.
- A pattern diagram is a diagram containing a single placement of a (compound) symbol, which typically has containers at some levels. This placement is anchored on a

specific "equipment" – a substation, a substation bay, an area, etc

A Pattern Diagram realization is not stored – only the Pattern Symbol is persistent.

The editor allows the user to visually define the symbols, containers and pattern symbols, allowing choosing from pre-defined functions to obtain attributes and associations to other equipment.
Whilst some functions are generic in nature (e.g. getting the parent, some type of child, the description) and thus can be reused, others are very specific and may involve complex computations. Typically, functions are implemented as database procedures possibly joining Scada, GIS and tagging models, but in fact they can be satisfied by any server (e.g. the topology processor).

As a result, any change in the model or in the pattern symbol is immediately reflected in all open diagrams. Note that this works for a switch open/close status change but an equipment insertion and removal is also a model change that will automatically be reflected in diagrams, possibly affecting multiple graphics.

Image 3 shows on the left a symbol pattern defined in the editor – on top there is a line configured to be repeated vertically for all equipments returned by the get Substations() function. On the right we can see the resulting diagram in operation – the symbol was anchored on an Operational Area, and thus this diagram displays a line for each substation of that area.

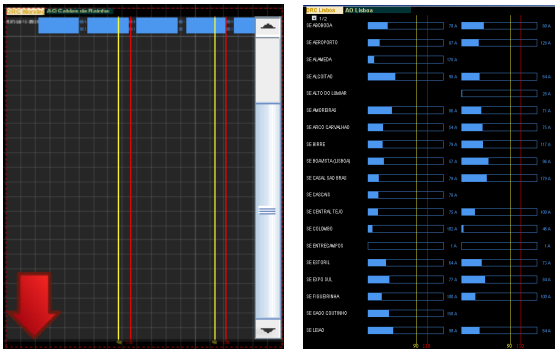

Image 3

Image 4 shows another example of a pattern diagram that has been anchored on a substation. On the bottom right corner we can see substation name and parent areas (obtained from Scada and GIS model data). The rest of the diagram shows automation telemetry data for that substation – all this data was obtained querying for specific scada entities anchored on that substation, that match some tag name patterns.

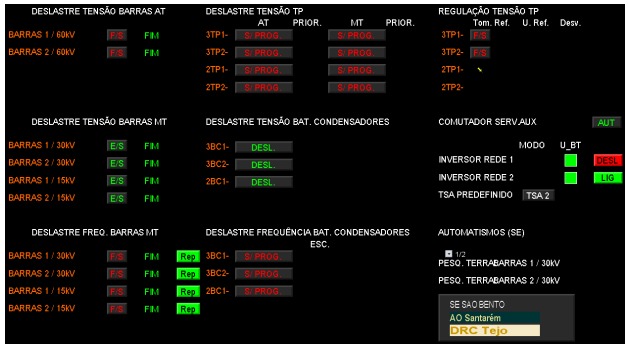

Image 4

Real Time Operation

The ability to generate an automatic and on the fly diagram of a network branch, by just one click, not only has become fundamental to network operation but also enhanced the paperless concept in EDPD Dispatch Centres.
Easily the operator can find the Nodes of a complex Branch/Feeder and act accordingly.
The feature of navigation thru different perspectives of the Pattern Diagram allows the operator to acknowledge the whole integration of that branch in the network.
This new concept has diret influence in the quality indexes of EDPD, namely the Equivalent Interruption Time (TIE).

Image 4

DISCUSSION

The model-driven diagrams automatic generation approach drastically reduces maintenance cost, is less prone to human errors and mitigates delays in the change actualization process. It also assures normalized diagram layouts with great benefits in the operation and control of the network.
The objectives were achieved and offered a better solution than EDPD expected, namely with respect to real time performance response and the level of automatic generation capability.