IEC TS 61970-556

Edition 1.0 2016-09

TECHNICAL SPECIFICATION

colour inside

Energy management system application program interface (EMS-API) – Part 556: CIM based graphic exchange format (CIM/G)

IEC TS 61970-556:2016-09(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing 20 000 terms and definitions in English and French, with equivalent terms in 15 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
65 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

IEC TS 61970-556

Edition 1.0 2016-09

TECHNICAL SPECIFICATION

colour inside

Energy management system application program interface (EMS-API) – Part 556: CIM based graphic exchange format (CIM/G)

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 33.200

ISBN 978-2-8322-3654-3

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

ENERGY MANAGEMENT SYSTEM APPLICATION PROGRAM INTERFACE (EMS-API) –

Part 556: CIM based graphic exchange format (CIM/G)

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.

2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.

3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.

6) All users should ensure that they have the latest edition of this publication.

7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.

8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

The main task of IEC technical committees is to prepare International Standards. In exceptional circumstances, a technical committee may propose the publication of a technical specification when

- the required support cannot be obtained for the publication of an International Standard, despite repeated efforts, or
- the subject is still under technical development or where, for any other reason, there is the future but no immediate possibility of an agreement on an International Standard.

Technical specifications are subject to review within three years of publication to decide whether they can be transformed into International Standards.

IEC TS 61970-556, which is a technical specification, has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

The text of this technical specification is based on the following documents:

Enquiry draft	Report on voting
57/1731/DTS	57/1770/RVC

Full information on the voting for the approval of this technical specification can be found in the report on voting indicated in the above table.

This document has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts in the IEC 61970 series, published under the general title *Energy management system application program interface (EMS-API)*, can be found on the IEC website.

In this technical specification, the following print types are used:

– *attributes for user defined graphic elements: in italic type.*

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- transformed into an International standard,
- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

This technical specification is part of the IEC 61970 series that define an Application Program Interface (API) for an Energy Management System (EMS).

IEC 61970-301 specifies a Common Information Model (CIM): a logical view of the physical aspects of an electric utility operation. The CIM is described using the Unified Modelling Language (UML), a language used to specify, visualize, and document systems in an object oriented manner.

This part of IEC 61970, which is a technical specification, specifies how to exchange CIM based graphic objects using XML, which details how to display an object. This document defines a format to facilitate efficient graphic data transfer, which will meet the real-time requirements for on-line remote diagram browsing and exchanging.

ENERGY MANAGEMENT SYSTEM APPLICATION PROGRAM INTERFACE (EMS-API) –

Part 556: CIM based graphic exchange format (CIM/G)

1 Scope

This part of IEC 61970, which is a technical specification, specifies a CIM-based graphic exchange format (CIM/G). It includes graphic file structure and graphic element definitions.

This document supports a mechanism for off-line exchange of graphic displays and on-line remote browsing of diagrams among distinct SCADA/EMS systems that may be provided by multiple vendors and located in different places.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61970-301, *Energy management system application program interface (EMS-API) – Part 301: Common information model (CIM) base*

IEC 61970-453, *Energy management system application program interface (EMS-API) – Part 453: Diagram layout profile*

IEC TS 61970-555, *Energy management system application program interface (EMS-API) – Part 555: CIM based efficient model exchange format*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 60050 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

3.1
domain object
instance of a class that models a Real-World Object with a unique identity

Note 1 to entry: A domain object inherits from a CIM *Identified Object;* it is normally not a diagram object. The definition of Domain object refers to IEC 61970-453. In this document, it indicates the graphic model of power system equipment.

3.2
diagram
electronic equivalent of a seamless paper plan

Note 1 to entry: The diagram is an identified container for the diagram objects. Examples of diagrams include substation schematics, transportation or distribution network orthogonal schematics, or pseudo-geographical schematics. A diagram has a well-defined coordinate space. A diagram is a set of power system graphic elements, basic graphic elements, analog and static text. This definition refers to IEC 61970-453. The diagram in this document indicates a single picture file, such as substation one-line diagram file, power grid diagram file.

3.3
diagram object
representation of domain objects or static background in a diagram

Note 1 to entry: An example for domain objects includes breakers. An example for static background object includes lakes. This definition refers to IEC 61970-453. Diagram object is defined as Graphic Element in this document. Graphic Element includes Basic Shape Graphic Element and Power System Graphic Element.

3.4
diagram object style
definition of how to render diagram objects possibly based on the state of domain objects

Note 1 to entry: Typically, the diagram object style is resolved in a very specific way for each system. This definition refers to IEC 61970-453. It indicates rendering style of Basic Graphic Elements and Power System Graphic Elements in this document.

3.5
layer
set of graphics displayed as varying granularity within certain zooming levels

Note 1 to entry: Each layer within the same magnification of graphics can be configured separately as visible or invisible, so as to meet the different description requirements of grid models in different application environments.

3.6
basic shape graphic element
SVG compatible basic shape graphic element, such as line, rectangle, circle, ellipse, polygon, etc.

3.7
power system graphic element
graphic element of power system equipment and topology

Note 1 to entry: This is a representation of domain objects and instantiation of diagram objects in IEC 61979-453.

4 Relationship of this technical specification with other standards

This part of IEC 61970 (CIM/G) is a part of the IEC 61970 series. It is based on the IEC 61970-300 and 61970-400 series. It has a close relationship with IEC 61970-453, as shown in Figure 1. This document inherits the roles shaped in IEC 61970-301; at the same time it complements and expands IEC 61970-453 on diagram presentation and rendering. The diagram layout profiles and linkage with power equipment model are defined in IEC 61970-453; while the detailed diagram presentation and exchange format are defined in this document, not only in off-line cases, but also for on-line applications.

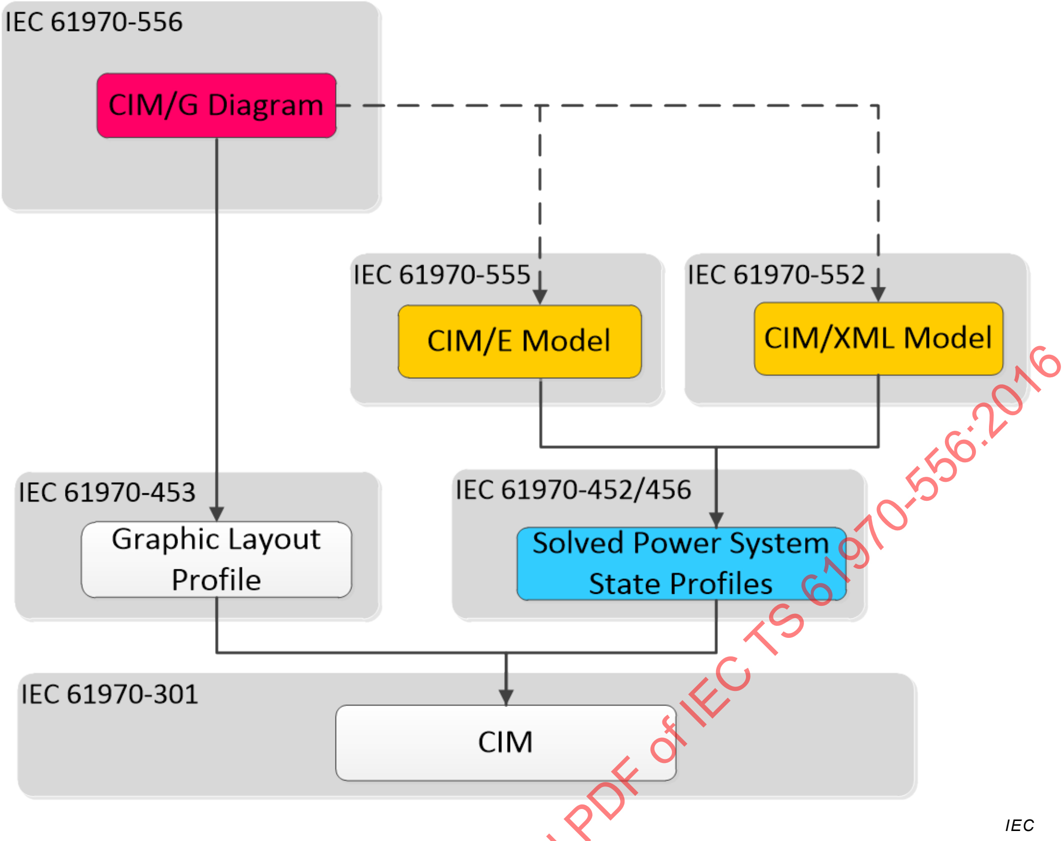


Figure 1 – Relationship with other standards

5 Use cases

With this document, the following scenarios are envisaged:

1) Diagram layout profile and the exchange rules of diagram objects are defined in IEC 61970-453, and the exchange entity is specified in this document, which specifies how to exchange CIM based diagram objects using XML and details how to display an object. This document can realize the off-line exchange of graphic displays among distinct SCADA/EMS systems that may be provided by multiple vendors and located in different places, i.e. between control center A and control center B, between a control center and a substation or power plant, etc. Mutual conversion between graphics and models can also be achieved efficiently by it. Figure 2 shows a typical scenario of off-line application.

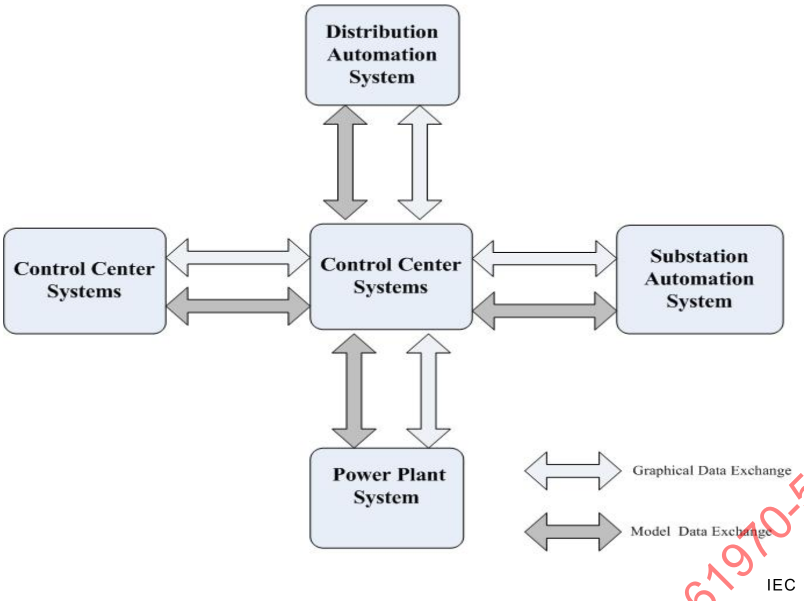


Figure 2 – Scenario of off-line application

2) On-line remote browsing of diagrams among distinct SCADA/EMS systems that may be provided by multiple vendors and located in different places, i.e. between control center A and control center B, between a control center and a substation or power plant, etc. Figure 3 shows a typical scenario of common MMI and online browsing.

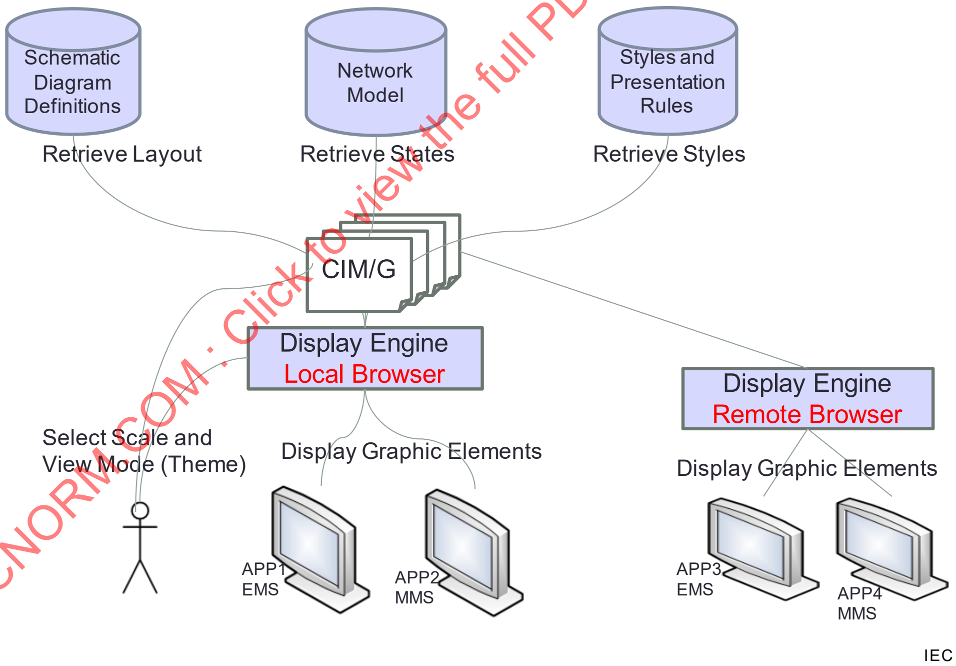


Figure 3 – Scenario of common MMI and online browsing

The online diagram browsing mechanism based on CIM/G is just like the web browsing based on HTML. The information browsing on Internet cannot be achieved without specific HTML description. The CIM/G specification works similarly to HTML specification, which describes the power equipment for on-line diagram browsing. Upon that substantial project costs could be saved, especially for those substations or power plants whose diagrams are seldom called by the control center but should be browsed quickly when required.

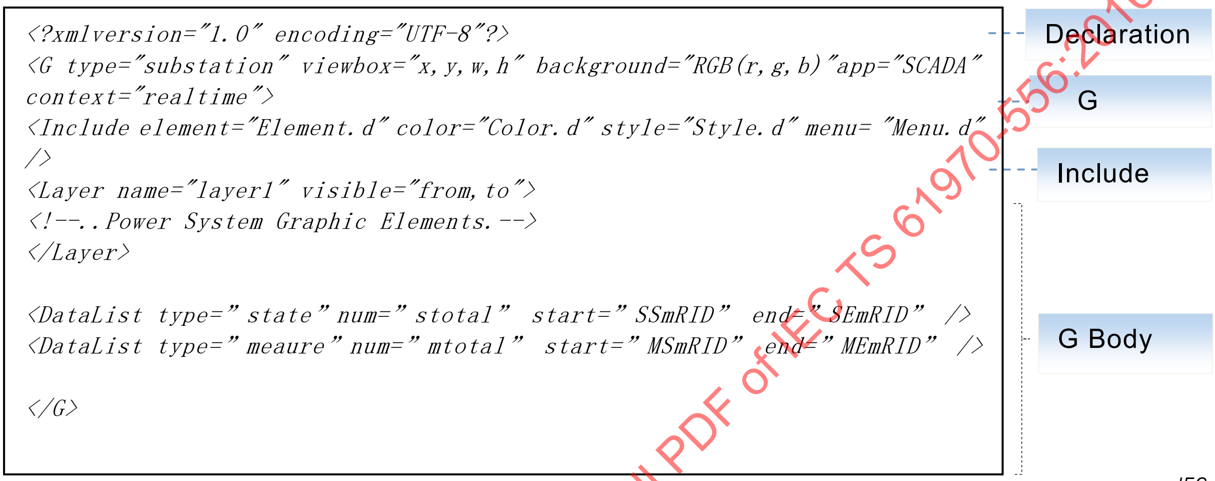
6 Structure of diagram file and definition file

6.1 General

CIM/G files are plain text files based on XML. There are two kinds of files in CIM/G; one is the visible diagram file, and the other is the definition file.

6.2 Diagram file structure

The diagram file includes Declaration tag, Include tag, and G block. ‘G’ block includes ‘Layer’ tag and graphic element tag. Figure 4 shows a general CIM/G file.



```
<?xmlversion="1.0" encoding="UTF-8"?>
<G type="substation" viewbox="x,y,w,h" background="RGB(r,g,b)"app="SCADA"
context="realtime">
<Include element="Element.d" color="Color.d" style="Style.d" menu= "Menu.d"
/>
<Layer name="layer1" visible="from,to">
<!--..Power System Graphic Elements.-->
</Layer>

<DataList type=" state " num=" stotal "  start=" SSmRID "  end=" SEmRID "  />
<DataList type=" meaure " num=" mtotal "  start=" MSmRID "  end=" MEmRID "  />

</G>
```

IEC

Figure 4 – Structure of diagram file

Declaration: *Declaration* shows XML version and encoding information of CIM/G.

Include tag: *Include* tag is used to import the predefined common used graphic information, the definition file structure refer to 8.2.The *Include* tag has three attributes: *element, color* and *style*.

- The *element=‘Element.d’* list the predefined power system graphic elements, refer to Annex A.
- The *color=‘Color.d’* list the predefined colour’s code, name, related RGB value and voltage level, all of them have the same meaning, it could be extended or redefined by the user (vendor), refer to Annex B.
- The *style=‘Style.d’* list the default values of diagram style, refer to Annex C.
- The *menu=‘Menu.d*’ list the predefined menu, it could be extended or redefined by the user (vendor), refer to Annex E.

G tag: G is the root tag in a G file. All of the graphic objects in G tag should be displayed concretely. The entire graphic data block starts and ends with G tag. *In the G block, all graphic elements are sensitive points which could interact with the user, just like ‘Anchor’ points in Web page.* It has five attributes: ‘*type’, ‘viewbox’, ‘background’ ,‘app’, ‘context’*.

- The ‘*type’* attribute indicates the diagram types such as *grid, substation, powerplant, GIS*(geographic diagram), etc.
- The ‘*viewbox’* attribute indicates the G diagram visible rectangle area (x,y,w,h), which defined by the top left point coordinate with the width and height of the diagram.
- The ‘*background’* attribute indicates the background colour by colour name or *RGB(r,g,b)* value, or a picture by *image file*. Other attributes could be defined by the user.

- The *'app'* attribute indicates the diagram could be used for the *'application',* such as *'SCADA'.*
- The *'Context'* attribute indicates the diagram could be used in the *'context',* such as *'realtime'.*

Layer tag: Layer tag is used to describe different layers graphic elements in a diagram. Each layer can appear at specific zooming level or different topics.

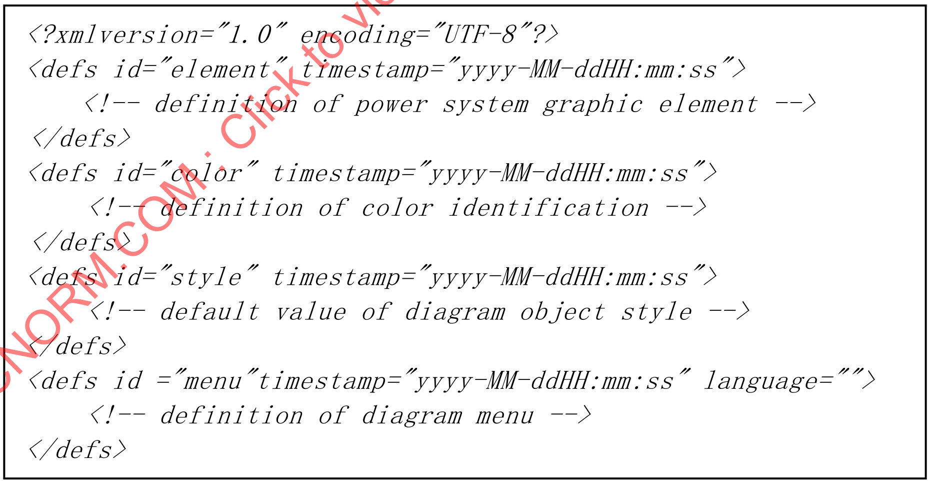
- The *name* attributes indicate the name of the layer.
- The *visible* attributes indicate the visible zoom level by two values of '*from*, *to'*. Other attributes could be defined by the user.

DataList tag: The *DataList* tag is used to describe the dynamic data set associated with power system graphic elements in the diagram. It could be used to speed up the preparation of the dynamic data list for the diagram data refreshing. It has 4 attributes: '*type', 'num', 'start'* , *'end'*.

- The *'type'* attribute indicates the type of dynamic data,it includes state, measurements and control.
- The *'num'* attribute indicates numbers of the dynamic data; the dynamic data is numbered in order.
- The *'start'* attribute indicates the number of the first dynamic data.
- The *'end'* attribute indicates the number of the last dynamic data.

6.3 Definition files structure

The definition files are named Element.d, Color.d , Style.d and Menu.d, referring to Annexes A, B, C and E. Here the 'defs' tag is used to define the power system elements, voltage and colours, as well as common styles. All of the definitions are not directly visible; they could be referenced from the diagram file (G file) − this is a big difference with the G file. The structure of these kinds of files is as shown in Figure 5.



```
<?xmlversion=″1.0″ encoding=″UTF-8″?>
<defs id=″element″ timestamp=″yyyy-MM-ddHH:mm:ss″>
    <!-- definition of power system graphic element -->
</defs>
<defs id=″color″ timestamp=″yyyy-MM-ddHH:mm:ss″>
    <!-- definition of color identification -->
</defs>
<defs id=″style″ timestamp=″yyyy-MM-ddHH:mm:ss″>
    <!-- default value of diagram object style -->
</defs>
<defs id =″menu″timestamp=″yyyy-MM-ddHH:mm:ss″ language=″″>
    <!-- definition of diagram menu -->
</defs>
```

Figure 5 – Structure of definition file

Here, the declaration tag and defs tag are used. The definitions of elements, colours, styles and menu could be put into one file as in this example, and could also be put into three different files as in Annexes A, B, C and E.

Declaration: *Declaration* shows XML version and encoding information of CIM/G, as in the diagram file.

defs: *defs* tag is also a standard tag of SVG; it indicates the definitions following it. It has two basic attributes: ‘id’ and ‘timestamp’. Besides the above two, other attributes could also be defined by the user.

- *Id*: indicates the type of the definition, such as element, colour, style and menu, it could be extended based on user needs.
- *Timestamp:* indicates the time stamp of the definition file be modified, *"yyyy-MM-ddHH:mm:ss"* means year, month, day, hour, minute, second. It is adopted for the version comparison between common style package and local user-defined style package, refer to Clause 9.
- *Language:* indicates the language used by menu, such as *English, French, German, Russian, Chinese,* etc. It is adopted for all menus.

7 Basic shape graphic element

The Basic Shape Graphic Element uses the same grammar rules of SVG (refer to Table 1). In order to improve the efficiency, some attribute names of basic elements have been abstracted from their original long name (refer to Table 1). In order to shape the graphic elements for power system, variables of basic graphic elements in the SVG are appropriately simplified in CIM/G, as shown in Table 1:

Table 1 – Common elements and attributes of SVG

Element name	Main attributes
rect	x, y, width, height, rx, ry
circle	cx, cy, r
ellipse	cx, cy, rx, ry
line	x1,y1,x2,y2
polyline	Points
polygon	Points
path	D
text	x, y, font-family, font-size
image	x, y, width, height, xlink:href
animate	from, to

As above, rect, circle, ellipse, line, polyline and polygon represent basic standard shape, text represents a single string, path represents the geometry of the outline of an object, defined in terms of‘M’ (moveto),‘L’ (lineto),‘C’ (curveto), ‘a’(arc) and‘z’ (closepath) commands. In path attributes, ’uppercase’ command indicates the absolute coordinates. In the attributes of image and ‘*xlink:href*’ indicates the external file references. In the attributes of animate, ‘*from’* and ‘*to’* show the moving way of the path.

CIM/G supports all of the basic shape graphic elements and allows users to define any other graphic elements. See Clause 8, *Power system graphic elements,* for the detail.

Any user-defined graphic element could be referenced directly in the same way as basic shape graphic elements. This is different from the indirect reference pattern in SVG, where the *‘xlink:href*’ attribute in ‘*use*’ tag is adopted when the user-defined graphic element is referenced. At the same time, the indirect reference pattern of user-defined graphic element is still allowed in CIM/G.

A reference example in CIM/G is shown as follows:

<UserDefinedElement attributes = ”……” / >

A reference example in SVG is shown as follows:

<use xlink:href=UserDefinedElement Attributes =”……” / >

To improve efficiency, CIM/G abbreviates the SVG attribute names. Common drawing attributes of basic painting environment are shown in Table 2:

Table 2 – Common drawing attributes

Description	Attribute name of CIM/G	Attribute name of SVG
width	w	Width
height	h	Height
line colour	lc	Stroke
line width	lw	stroke-width
line style	ls	stroke-dash-array
fill colour	fc	fill
fill mode	fm	fill-rule
transform	tf	Transform
font size	fs	font-size
font family	ff	font-family

The default value of common drawing attributes and styles are detailed in Annex C.

8 Power system graphic element

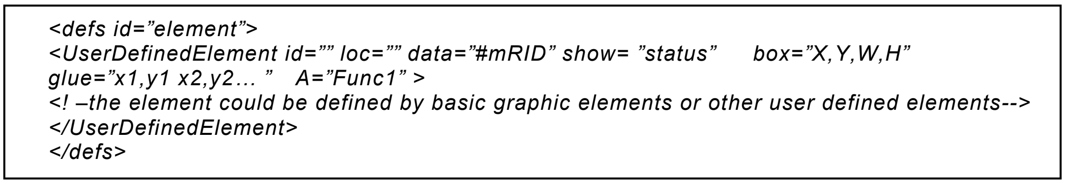
8.1 General

Power system graphic elements include all kinds of power system equipment (such as Breaker, Disconnector, Power transformer, Bus-bar, etc.) and equipment containers (such as bay, voltage level, substation, power-plant, etc.) (refer to IEC 61970-301 for detail) and the corresponding model exchange format refers to IEC TS 61970-555 for detail. All graphic elements in a diagram could be sensitive points or sensitive areas, which could interact with users through use of mouse or keyboard devices, just like ‘Anchor’ points on a Web page.

Power system equipment graphic elements are defined in ’element.d’ file (refer to Annex A). These elements can be referenced in any diagram.

8.2 User-defined power system graphic element

Power system graphic elements should correspond to the equipment model defined in IEC 61970-301. A power system graphic element is defined by ‘*defs*’ tag. The definition of elements is as shown in Figure 6.



```
<defs id=”element”>
<UserDefinedElement id=”” loc=”” data=”#mRID” show= ”status”     box=”X,Y,W,H”
glue=”x1,y1 x2,y2… ”   A=”Func1” >
<! –the element could be defined by basic graphic elements or other user defined elements-->
</UserDefinedElement>
</defs>
```

IEC

Figure 6 – Definition of power system graphic element

Here, *UserDefinedElement* indicates the name of the power system equipment, such as *Breaker, Disconnector, PowerTransformer, BusbarSection,ACLineSegment, DCLineSegment,* etc. which was defined in IEC 61970-301.The substation equipment defined in IEC 61850 could also be defined as a graphic element in the same way as above. Annex A list the common used power system graphic elements; it could be redefined or extended by the users or vendors as needed.

There are six common attributes for each user defined graphic element. The *'id', 'loc', 'data'* and *'show'* attributes are the formal attributes of the graphic element (like the formal parameters in the program calls).The actual parameters are passed by referencing graphic element. The *'box'* and '*glue*' is the ontology property of the graphic element. Meanings of the attributes are as follows:

id: indicates the unique identifier of this graphic element in current diagram. The *id* could be empty in definition file, but must be passed by referencing file.

loc: indicates the location area of a graphic element in the canvas: x and y are the upper left corner coordinates of the graphic element, w and h its width and height in the canvas. The *loc* could be empty in the definition file, but must be passed by the referencing file. Here x, y, w and h should be in low case.

data: indicates the graphic element associated CIM model data or object ID. There are two types of attribute data, one is the actual value, and the other is the indirect data ID or index with a '#' prefix. There may be several data in the attribute, separated by commas. The *data* could be empty in definition file, but must be passed by the referencing file.

show: indicates display status and style of the graphic element. The format is *show=”Q,T,F,S”.* Here, each of Q,T,F and S is an integer number less than 255 (maximum one byte value). The *data* could be empty in the definition file, but must be passed by the referencing file. The detail refers to 8.3.

box: indicates the rectangle of the graphics element: 'X','Y' are the left-top coordinates of the element box, 'W','H' the width and height of the box. X, Y, W and H for definition should be in upper case. They should be defined in the definition file. Here, X, Y, W and H should be in upper case.

glue: indicates the connectivity points of the graphic element, where x0,y0till to xn, yn meaning the graphic coordinates of terminals. The number of terminals is determined by the associated power equipment, such as two terminals for a circuit breaker, three terminals for a three-winding power transformer, while the terminal number of a bus is determined by the number of lines connected to this bus and the number may be n. The glue is numbered according to the order of the coordinates, e.g. point (x0,y0) corresponds to glue0, point (x1,y1) to glue1 and till point (xn,yn) to gluen. This attribute should be defined in definition file.

A: indicates the Anchor related interactive functions or diagrams to be called when this element being pointed by mouse device.

8.3 Reference of user-defined graphic element

The user-defined graphic element may be referenced directly as shown in Figure 7.

```
<G>

<UserDefinedElement  id=”myid”  loc=”x,y w,h”  data=”#mRID”  show= ”status”  A=”F1”/>

  </G>
```

IEC

Figure 7 – Reference of user-defined graphic element

The *‘id’, ‘loc’, ‘data’, ‘show*’ and *‘A’* attributes may be adopted via directly referencing the user-defined graphic element.

id: indicates the unique identifier of this graphic element in current diagram.

loc: indicates the location area of a graphic element in the canvas: x and y are the upper left corner coordinates of the graphic element, w and h its width and height on the graphic element. x, y, w, h for referencing should be in low case. When a graphic element is drawn in a diagram, its real left-top coordinates are calculated by:

x_real = XX_definition+x_reference;

y_real = YY_definition+y_reference;

and when ‘w’ and ‘h’ of is Inconsistent with ‘W’ and ‘H’ in the definition of graphic element, ‘W’ and ‘H’ should be transformed to ‘w’ and ‘h’.

data: indicates the graphic element associated CIM model data or object ID. There are two types of attribute data, one is the actual value, and the other is the indirectly data ID or index with a ‘#’ prefix. There may be several data in the attribute, separated by commas.

show: indicates display status and style of the graphic element. The format is *show=”Q,T,F,S”.* Here, each of Q,T,F and S is an integer number less than 255(maximum one byte value). The show attribute must be passed from referencing file. The *data* could be empty in definition file, but must be passed from referencing file.

‘Q’ indicates Quality of data, the definition refers to IEC 61970-301 and IEC 61850.

‘T’ indicates Topological status (Energized, Un-energized and Grounded); the related colour refers to Annex B.

‘F’ indicates Flashing flag, F=0 means no flashing; F=1 means flashing every 1 second, other values could be defined by users.

‘S’ indicates Shape change marker; S=0 means no shape change, other values could be defined by users.

For example, show=”0,3,0,0” means the data quality is valid, the topological colour is 500kV voltage colour code, and there is no flashing or shape changing. For convenience in some cases, “Q,T,F,S” could be simplified as “status”.

A: indicates the Anchor related interactive functions or diagrams to be called when this element is being pointed by a mouse device.

The five attributes(*id, loc, data, show, A*) should be passed by referencing diagram, that indicates the actual equipment id, location in the diagram, dynamic data showing styles and interactive function. Based on all attributes, the user defined element could be rendered on the diagram canvas.

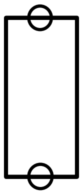
9 Examples of some graphic elements and diagram

9.1 Breaker graphic element

The Breaker graphic element is a graphic presentation of a circuit breaker in the power system. It is commonly presented by a rectangle. Colour of the device is set according to its voltage level (refer to Annex A), or the name of the colour, the RGB value. An unfilled rectangle indicates status of the breaker is open, while a filled rectangle shows it is close. A Breaker element may have other common attributes which are described in the basic graphic element.

1) Definition of Breaker in ‘element.d’

A Breaker element definition is shown in Figure 8. Refer to Annex A for the whole graphic element set.

Definition	Shape
<defs> *<Breaker id="breaker0" loc="" data="" show="" box="0,042,18" glue="18,3 18,15" A=”F1”>* *<rect x="4" y="3" w="12" h="28" fm="0"/>* *</Breaker>* *<defs>*	

IEC

Figure 8 – Definition of Breaker element

2) Reference of Breaker in a diagram

A Breaker element is referenced as shown in Figure 9.

```
<Breaker id="CB5013"   loc ="50,56 16,32"   data="#765013"   show= "0,C,0,0"   A=”F1”/>
```

IEC

Figure 9 – Reference of Breaker element

Here show=”0,C,0,0” means the data quality is valid, the topological colour is normal voltage colour code, and there are no flashing or shape changing.

9.2 Dynamic Text graphic element

A Dynamic Text（DText）graphic element is adopted to describe the electrical measurement. It is not a power system equipment, however it could be used in all power system graphic element. Its definition structure is shown in Figure 10.

```
<defs>
    <DText   id=”dtext1”   loc=”” data=”#mRID” show=”status” box=”X,Y W,H” A=”F1”/>
</defs>
```

IEC

Figure 10 – Definition of Dynamic Text graphic element

A reference example of DText graphic element is illustrated as shown in Figure 11.

```
<G>
            <DText id="id1" loc="x,yw,h" data="#mRID" show="Q,T,F,S" A="F1"/>
</G>
```

IEC

Figure 11 – Reference of Dynamic Text graphic element

The *'id', 'loc', 'data'* and *'show'* attributes may be adopted via directly referencing the tag. Meanings of the attributes refer to 8.2 and 8.3.

9.3 Link graphic element

The Link graphic element is a special graphic element. It is not a normal equipment, just a short line to link two or more equipment, and without any measurement and impedance. It presents the idea of 'ConnectivityNode' defined in IEC 61970-301. The well designed power system graphic element could be connected directly by its glue point, in most cases it is not necessary to use the link element. The Link element may have three attributes: *points, connect and show*.

An example of Link graphic element is illustrated as shown in Figure 12.

```
<G>

<Link points="x1,y1 x2,y2" connect="glue0,equip0;glue1,equip1" show="Q,T,F,S" />

</G>
```

IEC

Figure 12 – Reference of Link graphic element

points: This indicates the linked points.

connect: This represents relationship of connection between equipment graphic elements. The format is defined with connect = "glue0, equip0;glue1, equip1", which means that the glue point 0 of equipment0 and glue point 1 of equipment1 are connected by the Link element.

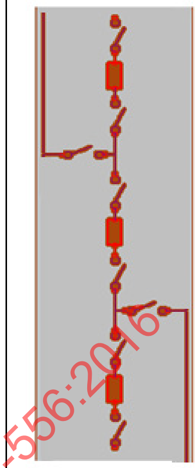
9.4 Bay graphic element

It is the graphic presentation of 'Bay' model defined in IEC 61970-301, which is used to describe the configuration of breakers, disconnectors and ground disconnectors in the bay area. Since most bays can be generalized as bay graphic elements, the substation diagram can be briefly described by several bay graphic elements.

The definition of Bay graphic element is described in 8.3. How to reference the Bay graphic element is addressed in 8.4.

An example of Bay graphic element is illustrated as shown in Figure 13.

Definition	Shape

<defs>
  <Bay id="bay1-3/2" Volt=”500KV” loc=”” data=”” show=”” box="X,Y,W,H" glue=”x1,y1” A=””>
    <Disconnector id=”d1”loc=”54,9620,20”data=”#mRID” show=”status1”/>
    <Breaker id=”b1”loc=”50,56 20,20” data=”#mRID” show=”status2”/>
    <Disconnector id=”d2”loc=”54,181 20,20” data=”#mRID” show=”status3”/>
    <Disconnectorid=”d3”loc=”54,272 20,20” data=”#mRID” show=”status4”/>
    <Breakerid=”b2”loc=”50,232 20,20” data=”#mRID” show=”status5”/>
    <Disconnectorid=”d4”loc=”54,357 20,20” data=”#mRID” show=”status6”/>
    <Disconnector id=”d5”loc=”54,448 20,20” data=”#mRID” show=”status7”/>
    <Breaker id=”b3”loc=”50,408 20,20” data=”#mRID” show=”status8”/>
    <Disconnector id=”d6”loc=”50,5 20,20” data=”#mRID”show=”status9”/>
    <Disconnector id=”d7”loc=”5,151 20,20” data=”#mRID” show=”status10”/>
    <Disconnector id=”d8”loc=”65,327 20,20” data=”#mRID” show=”status11”/>
    <Link points="59,440 59,452 "connect="1,d7"show=”status12”/>
    <Linkpoints="59,264 59,276" connect="1,d8" show=”status13”/>
  </Bay>
</defs>
	

IEC

Figure 13 – An example of bay element definition

A Bay element is referenced as shown in Figure 14.

```
<G>

<Bay id="bay1-3/2" Volt=”500KV”loc="x,y,w,h" data= "#mRID"show="Q,T,F,S" A=”F1”/>

</G>
```

IEC

Figure 14 – Reference of a bay element

9.5 Voltage level diagram

A Voltage level diagram is a container, and may be adopted directly without pre-defined. There may be multiple buses and bays within a Voltage level diagram. An example is illustrated in Figure 15, which includes two buses and two bays.

```
<G>
  <Voltagelevel id="220kV" station=”S1”loc=”x,y,w,h” data="#mRID" show=”Q,T,F,S” >
       <Bus id="bus1"   loc="x1,y1 w1,h1"   data="#mRID1"   show="status1" />
       <Bay id="bay1"   loc="x2,y2 w2,h2"   data="#mRID2"   show="status2" />
       <Bay id="bay2"   loc="x3,y3 w3,h3"   data="#mRID3"   show="status3" />
       <Bus id="bus2"   loc="x4,y4 w4,h4"   data="#mRID4"   show="status4" />
  </Voltagelevel>
</G>
```

IEC

Figure 15 – Example of voltage level diagram

9.6 Substation or power plant diagram

Substation or Power Plant diagram is also a container, and may be adopted directly without pre-defined. There may be multiple equipment within a Substation or Power Plant diagram and different voltage levels may be included in it. An example of Substation or Power Plant diagram is illustrated in Figure 16, which includes three voltage levels. The dynamic data list for a diagram could be useful to speed up the dynamic data refreshment, especially for substation diagram, power plant diagram, power grid diagram and GIS diagram. Please refer to 6.2, *Diagram file structure*.

```
<G>
  <Substation id="name1" loc="x1,y1 w1,h1" data="#mRID" show="Q,T,F,S"   grid=”G1”>
      <Voltagelevel id="500kV" loc="x2,y2 w2,h2"   data="#mRID2"   show=”status1” />
      <Voltagelevel id="220kV" loc="x3,y3 w3,h3"   data="#mRID3"   show=”status2” />
      <Voltagelevel id="35kV"   loc="x4,y4 w4,h4"   data="#mRID4"   show=”status3” />
       <DataList type=”state” num=”stotal” start=”SSmRID” end=”SEmRID” />
      <DataList type=”meaure” num=”mtotal” start=”MSmRID” end=”MEmRID” />
  </Substation>
</G>
```

IEC

Figure 16 – Example of substation diagram with voltage level

In some cases, bay or equipment may be described without adopting the voltage graphic element, for example as shown in Figure 17:

```
<G>
<Substationid=”name1”loc=”x1,y1 w1,h1” data=”#mRID” show=”Q,T,F,S”volt=”500kV” grid=”G1”>
        <Bus id=”bus1” loc=”x1,y1 w1,h1” data=”#mRID1” show=”status1” />
        <Bay id=”bay1” loc=”x2,y2 w2,h2” data=”#mRID2” show=”status2”/>
        <Bay id=”bay2” loc=”x3,y3 w3,h3” data=”#mRID3” show=”status3”/>
        <Bus id=”bus2” loc=”x4,y4 w4,h4” data=”#mRID4” show=”status4” />
        <DataList   type=”state”    num=”stotal”   start=”SSmRID” end=”SEmRID” />
        <DataList   type=”meaure” num=”mtotal”   start=”MSmRID” end=”MEmRID” />
   </Substation>
</G>
```

IEC

Figure 17 – Example of substation diagram without voltage level

In this example, the voltage attribute is added to Substation element, due to the fact that there is no Voltage container element. If individual bay has different voltage, the voltage attribute should be added to each Bay element.

9.7 Power grid graphic diagram

The Power grid diagram, as shown in Figure 18, is shaped as topology of connecting the transmission lines and substations or power plants. Usually a substation or power plant is simplified to a point or a specific icon. There are two types of power grid graphic, one is the ordinary power grid diagram, and the other is the geographic power grid diagram based on GIS.

In an ordinary power grid diagram, a substation or power plant is shown as a Substation/Power Plant graphic element when full display, and is enlarged to detailed one-line diagram when the graphics zoomed in enough, and vice versa when zooming out.

In the geographic Power grid diagram, a substation or power plant is drawn according to the actual geographic coordinates, and is also marked as a Substation/Power Plant graphic element. When the graphic is zoomed in or out, different geographic layers may be displayed along with the corresponding scaling.

The Power grid graphic element is the topmost container element, and may be adopted directly without pre-defined. An example of Power grid graphic element is illustrated in Figure 18, which includes four transmission line and four substations connected as a grid.

```
<G>
<PowerGrid id=”grid1” box=”x,y,w,h” type=”normal” volt=″500kV″>
<ACLineSegment  id=″L1″ loc=″x1,y1,w1,h1″ data=″#mRID1″ show=″status1″ A=”F1”  />
<ACLineSegment  id=″L2″ loc=″x2,y2,w2,h2″ data=″#mRID2″ show=″status2″ A=”F2”  />
<ACLineSegment  id=″L3″ loc=″x3,y3,w3,h3″ data=″#mRID3″ show=″status3″ A=”F3” />
<ACLineSegment  id=″L4″ loc=″x4,y4,w4,h4″ data=″#mRID4″ show=″status4″ A=”F4” />

<Substation  id=″S1″ loc=″x5,y5,w5,h5″ data=″#mRID5″ show=″status5″  A=”F5” />
<Substation  id=″S2″ loc=″x6,y6,w6,h6″ data=″#mRID6″ show=″status6″  A=”F6”/>
<Substation  id=″S3″ loc=″x7,y7,w7,h7″ data=″#mRID7″ show=″status7″  A=”F7”/>
<Substation  id=″S4″ loc=″x8,y8,w8,h8″ data=″#mRID8″ show=″status8″ A=”F8”  />

<DataList type=”state”num=”stotal”  start=”SSmRID”  end=”SEmRID”   />
<DataList type=”meaure”num=”mtotal”  start=”MSmRID”  end=”MEmRID”  />

</PowerGrid>
</G>
```

IEC

Figure 18 – Example of power grid graphic diagram

The Power grid diagram usually has four attributes:

id: indicates the name of power grid.

box: indicates the box area of power grid.

type: indicates the type of power grid, ‘normal’ indicates the ordinary power grid diagram, and its value could be *“OneLine” or “GIS”*. ‘GIS’ means the geographic power grid diagram.

volt: indicates the voltage level of the power grid.

10 Mechanism of runtime display

10.1 Mechanism of runtime local display

The mechanism of on-line remote diagram browsing between different control centers is as follows (see Figure 19).

1) The graphic browser sends a request to the graphic server for diagram A.
2) The graphic server responds and sends the ‘A.g’ file to the graphic browser.
3) The graphic browser renders a static graph, based on the received ‘A.g’ file.
4) The graphic server sends real-time data continuously to the browser.
5) The graphic browser refreshes the dynamic data display.
6) When the graphic browser sends a stop/quit request, the graphic server stops sending the data.

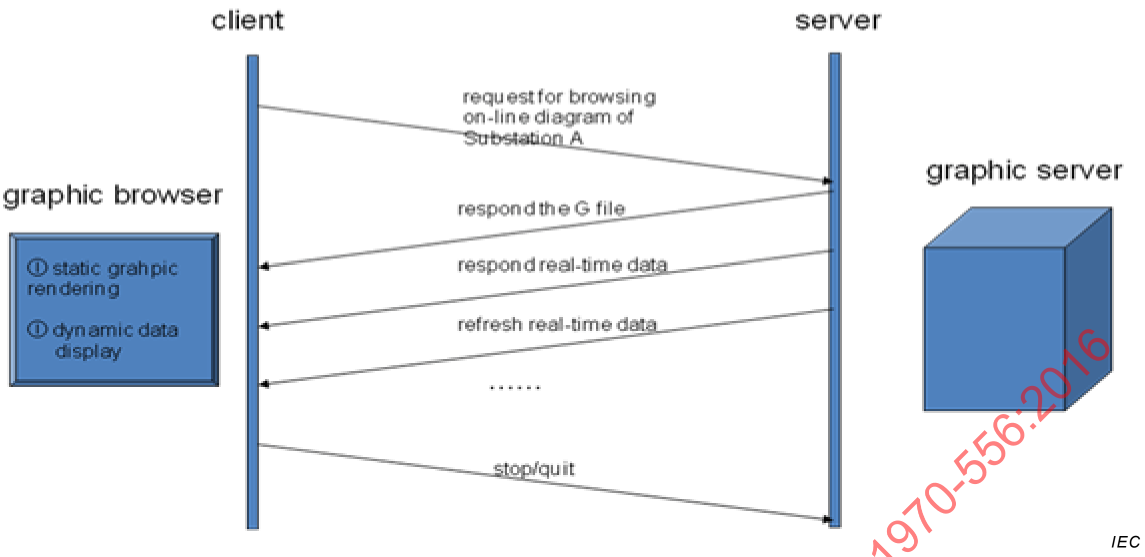


Figure 19 – Mechanism of runtime local display

10.2 Mechanism of runtime remote browsing

The common graphical style package, such as *‘element.d’, ‘color.d’’style.d’, and ‘memu.d’*are all defined at the server, individual graphical style package may be defined by the users at the client. When the remote graphics is required to be displayed locally, either individual or common graphical style package may be adopted. When the graphic is displayed by the common graphical style package, a comparison should be run between it and the locally individual graphical style package. If they are consistent, the locally individual package may be adopted to render the graphic display style. On the contrary, if they are discrepant, the locally individual package should be updated and be consistent with the common package at the server before rendering the graphic display style. But the runtime remote browsing, as well as the required handshaking between client and server for display call up, is not part of the scope of this document. This technical specification only covers the display format (see Figure 20).

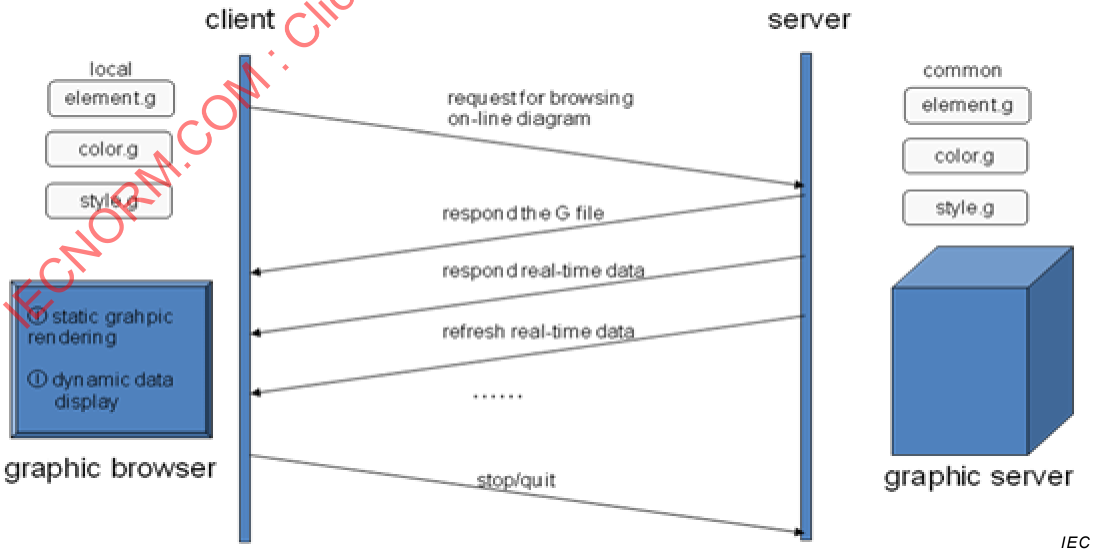


Figure 20 – Mechanism of runtime remote browsing

10.3 Menu description

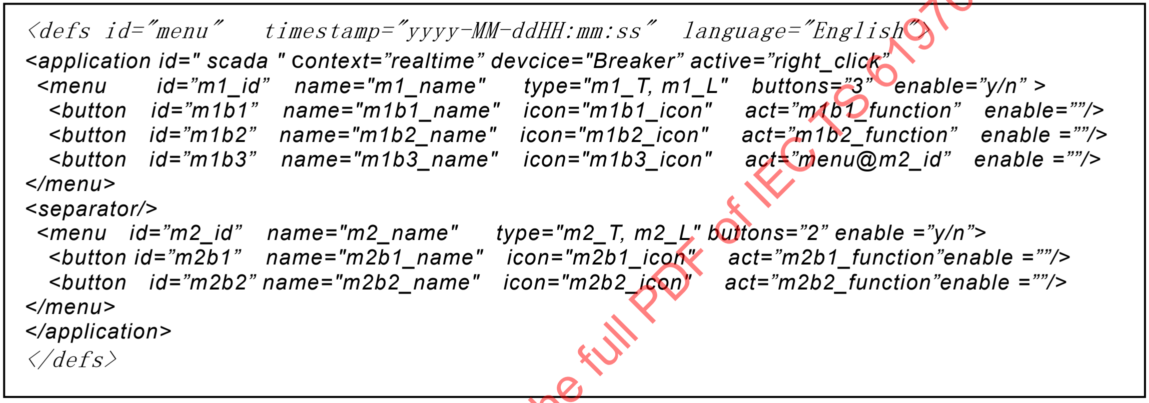
To describe a menu, the *‘application’, ‘menu’ and ‘button’* elements are used. The ‘application’ element is used to describe the application environment that a menu belongs to. The application element may include multiple ‘menu’ elements, while each menu element includes several ‘button’ elements.

The menu can include many layers of nested elements to form a hierarchical structure, with root menu on the top layer and the leaf menu on the bottom layer.

Each ‘button’ has an action expressed by ‘act’ attribute, which could be a specific action (i.e., to call a function, program or another diagram), or the menu of next level (with a prefix of menu@ to distinguish from the direct invocation).

The action attribute ‘act’ of all the ‘button’ elements for the leaf menu shall have a value of a specific action.

The *‘separator’* element can be used to separate the ‘button’ elements into different groups. One example of menu definition is as shown in Figure 21.



```
<defs id="menu"   timestamp="yyyy-MM-ddHH:mm:ss"   language="English">
<application id=" scada " context="realtime" devcice="Breaker" active="right_click"
 <menu      id="m1_id"    name="m1_name"     type="m1_T, m1_L"   buttons="3"  enable="y/n" >
  <button   id="m1b1"    name="m1b1_name"   icon="m1b1_icon"    act="m1b1_function"   enable=""/>
  <button   id="m1b2"    name="m1b2_name"   icon="m1b2_icon"    act="m1b2_function"   enable =""/>
  <button   id="m1b3"    name="m1b3_name"   icon="m1b3_icon"    act="menu@m2_id"   enable =""/>
</menu>
<separator/>
 <menu   id="m2_id"    name="m2_name"     type="m2_T, m2_L" buttons="2" enable ="y/n">
  <button id="m2b1"    name="m2b1_name"    icon="m2b1_icon"    act="m2b1_function"enable =""/>
  <button   id="m2b2" name="m2b2_name"    icon="m2b2_icon"    act="m2b2_function"enable =""/>
</menu>
</application>
</defs>
```

IEC

Figure 21 – Menu description

1) The ‘defs’ element is used to define all the menus of the system, which includes at least three basic attributes:
    - ‘*id*’ attribute, whose value must be menu to indicate this tag is to define a menu;
    - ‘*timestamp*’ attribute, to indicate the latest modification time of this menu definition with precision of second;
    - ‘*language*’ attribute, to indicate the language used for the names of the menu and button, such as *English, Chinese,* etc. The scope includes all the menus and buttons packaged within the defs element. It is usually set while initializing the system.
2) The application element is used to define the application environment that the menu belongs to, which includes at least four basic attributes:
    - ‘*id*’ attribute, the identifier of the application. If its value is ‘window’, that means it represents the menu of the window. If its value is the application name, which means it represents the menu of the devices within the diagram. Its value could be string or integer.
    - ‘*context*’ attribute, the context that the application belongs to. Its value could be string or integer, such as: Real time(the value is 0),Study(the values is 1),Test(the value is 2).
    - ‘*device*’ attribute, to describe the device type. Different menus can be defined for different devices. Its value is usually the name of device type.
    - ‘*active*’ attribute, to indicate the activation method of the menu, e.g., the value of right-click means to activate the menu by mouse right-clicking.
3) The menu element is used to define a single menu, which includes at least five basic attributes:

- ‘*id*’ attribute, the identifier of the menu. Its value could be string or integer.
- ‘*name*’ attribute, the name of the menu. It could be the string in the language specified by the ‘*language*’ attribute.
- ‘*buttons*’ attribute, the number of buttons included in the menu. Its value is an integer.
- ‘*enable*’ attribute, to indicate if the menu is enabled or disabled. It is expressed as ‘y’ or ‘n’, where ‘y’ means the menu is enabled, while ‘n’ means disabled.
- ‘*type*’ attribute, the type of the menu. It is expressed in a pair (T, L), where T represents the menu type, and L represents the location of menu on window. For example, a fixed menu on the top of the window, the ‘*type*’ should be *(M_FIXED, L_TOP) or (0, 0).*

The frequently used menu types include:

M_FIXED:	fixed menu, value is 0;
M_PULLDOWN:	pull down menu, value is 1;
M_POPUP:	popup menu, value is 2;
M_BOOKMARK:	bookmark menu, value is 3;
M_LIST:	list menu, value is 4;
M_FLOATING:	floating menu, value is 5.

The frequently used menu locations include:

L_TOP:	on the top of the window, value is 0;
L_BOTTOM:	on the bottom of the window, value is 1;
L_LEFT:	on the left of the window, value is 2;
L_RIGHT:	on the right of the window, value is 3.

4) The button element is used to define a single button, which includes at least five basic attributes:
   - ‘*id*’ attribute, the identifier of the button. Its value could be string or integer.
   - ‘*name*’ attribute, the name of the button. It could be the string in the language specified by the ‘*language*’ attribute.
   - ‘*icon*’ attribute, the icon of the button.
   - ‘*act*’ attribute, the action of the button. While expressing a direct action, its value corresponds to the name of called function or program (including the path). While expressing the next level menu, it will put prefix menu@ before the identifier of the next level menu in order to distinguish from the direct action.
   - ‘*enable*’ attribute, to indicate if the button is enabled or disabled. The representation method is same with that of the ‘*enable*’ attribute of the menu. However, this attribute is valid only when the value of the ‘*enable*’ attribute of the menu is *‘y’*.

Refer to Annex E for an example of menu definition.

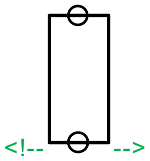
Annex A
(informative)

Definition of power system graphic element

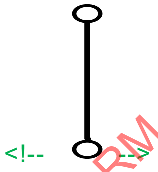
This annex describes a set of common definitions for graphic elements of power system. As the default setting, the graphic elements of the power system in diagram will primarily be instantiated by referring to the following definition. The content of the definition can be expanded, which is named as ‘element.d’.

The following is an example. It will not be imposed by this technical spcification and can be redefined or extended according to user’s usage.

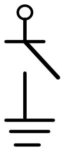
```
<? xml version="1.0" encoding="UTF-8"?>
<defs  id="element" timestamp="yyyy-MM-ddHH:mm:ss">

<Breaker id="" loc="" data="" show="" box="0,0 36,18" glue="18,3 18,15" >
<rect  x="4" y="3" w="28"h="12" fm="0" />
<text  x="40" y="8" w="10" h="15" value="*id"  A=”” />
</Breaker>
```



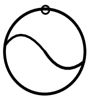
```
<Disconnector id="" loc="" data="" show="" box="0,0 48,16" glue="11,4 11,42" >
<circle cx="11" cy="7" r="3" />
<line x1="11" y1="10" x2="11" y2="36"/>
<circle cx="11" cy="39" r="3" />
<textx="20" y="16" w="10" h="15" value="*id"  A=”” />
</Disconnector>
```



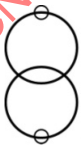
```
<GroundDisconnector id=""loc="" data="" show="" box="0,0 42,18" glue="4,12" >
<circle cx="6" cy="12" r="2" />
<line x1="6" y1="12" x2="14" y2="12" />
<line x1="14" y1="9" x2="14" y2="15" />
<line x1="14" y1="12" x2="22" y2="5" />
<line x1="22" y1="12" x2="30" y2="12" />
<line x1="30" y1="7" x2="30" y2="17"/>
<line x1="32" y1="8" x2="32" y2="16"/>
<line x1="34" y1="9"x2="34" y2="15"/>
<text x="40" y="10" w="10" h="15" value="*id"  A=”” />
</GroundDisconnector>
```

<!--  -->

```
<Generator id="" loc="" data="" show="" box="0,0  38,40" glue="4,12" >
<circle cx="19" cy="20" r="14" />
<path  d="M5,20 a7,7 1 0 14, 0"/>
<path  d="M33,20 a7,7 1 0 -14, 0"/>
<text  x="40" y="20" w="10" h="15" value="*id"  A=”” />
<text  x="40" y="40" w="20" h="15" value="P:"  />
<DText  id=”*id”loc=”x,y,w,h” data=”#Mrid1”  />
<text  x="40" y="60" w="20" h="15" value="Q:" />
<DText  id=”*id”loc=”x,y,w,h” data=”#Mrid2”   />

</Generator>
```

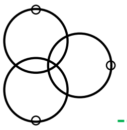
<!--  -->

```
<PowerTransformer2  id=""loc="" data="" show="" box="0,0  62,40" glue="20,4
20,56">
<circle cx="20" cy="20" r="16" />
<circle cx="20" cy="40"r="16"/>
<text x="40" y="40" w="10" h="15" value="*id"  A=”” />
<text  x="40" y="0" w="20" h="15" value="P:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID1” show=”status” />
<text  x="40" y="20" w="20" h="15" value="Q:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID2” show=”status” />
<text  x="40" y="60" w="20" h="15" value="P:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID3” show=”status” />
<text  x="40" y="80" w="20" h="15" value="Q:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID4” show=”status” />
</PowerTransformer2>
```

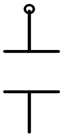
<!--  -->

```
<PowerTransformer3 id=""loc="" data="" show="" box="0,0 62,40" glue="40,4 14,29
66,29">
<circle cx="40" cy="20"r="16" />
<circle cx="25" cy="40"r="16" />
<circle cx="55" cy="40" r="16" />
<text x="0" y="60" w="10" h="15" value="*id"  A=”” />
<text  x="40" y="0" w="20" h="15" value="P:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID1” show=”status” />
```

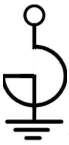
```
<text  x="40" y="20" w="20" h="15" value="Q:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID2” show=”status” />
<text  x="40" y="60" w="20" h="15" value="P:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID3” show=”status” />
<text  x="40" y="80" w="20" h="15" value="Q:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID4” show=”status” />
<text  x="40" y="100" w="20" h="15" value="P:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID5” show=”status” />
<text  x="40" y="120" w="20" h="15" value="Q:"  />
<DText id=”*id” loc=”x,y,w,h” data=”#mRID6” show=”status” />

</PowerTransformer3>
```

<!--  -->

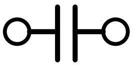
```
<ShuntCapacitor id="" loc="" data="" show="" box="0,0 26,40" glue="13,8" >
<line x1="13" y1="8" x2="13" y2="16" />
<line x1="4" y1="16" x2="22" y2="16" />
<line x1="4" y1="22" x2="22" y2="22" />
<line x1="13" y1="30" x2="13" y2="38"/>
<text x="30" y="12" w="10" h="15" value="*id" A=””  />
</ShuntCapacitor>
```

<!--  -->

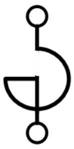
```
<ShuntReactor id="" loc="" data="" show="" box="0,0 26,40" glue="13,3" >
<path  d=”M13,10 a8,8 0 1 0 -5 8”/>
<line  x1="13" y1="3" x2="13" y2="10" />
<line  x1="5" y1="18" x2="13" y2="18" />
<line  x1="13" y1="18" x2="13" y2="37" />
<line  x1="4" y1="37" x2="22" y2="37" />
<line  x1="6" y1="39" x2="20" y2="39"/>
<line  x1="8" y1="41" x2="18" y2="41" />
<text x="30" y="16" w="10" h="15" value="*id"  A=”” />
</ShuntReactor>
```

<!--  -->

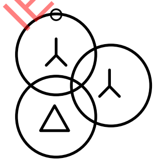
```
<SeriesCapacitor id=""loc="" data="" show="" box="0,0 40,26" glue="6,11" >
   <line x1="7" y1="11" x2="15" y2=”11” />
   <line x1="16" y1=”4”x2="16" y2="19"/>
   <line x1="23" y1=”4”x2="23" y2="19"/>
   <linex1="23" y1=”11” x2="32" y2="11"/>
   <text x="38" y="10" w="10" h="15" value="*id"  A=”” />
</SeriesCapacitor>
```

<!--  -->

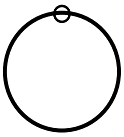
```
<SeriesReactor id="" loc="" data="" show="" box="0,0 26,40" glue="13,5 13,30" >
<line x1=“13” y1=”5” x2=”13” y2=”10” />
<line x1="8" y1="18" x2="13" y2="18" />
<line x1="13" y1="18" x2="13" y2="30" />
<path d=”M13,10 a8,8 0 1 0 -5 8” />
<text x="38" y="10" w="10" h="15" value="*id"  A=”” />
</SeriesReactor>
```

<!--  -->

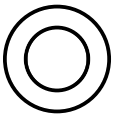
```
<PT id=""loc="" data="" show="" box="0,0 40,44" glue="21,5" >
   <circle cx="20" cy="15" r="10"/>
   <circle cx="28" cy="29" r="10"/>
   <circle cx="13" cy="29" r="10" />
   <line x1="20" x2="20" y1="14" y2="19" />
   <line x1="24" x2="20" y1="10" y2="13" />
   <line x1="20" x2="20" y1="15" y2="15" />
   <line x1="15" x2="19" y1="10" y2="13" />
   <line x1="29" x2="29" y1="31" y2="31" />
   <line x1="9" x2="7" y1="27" y2="31" />
   <line x1="8" x2="14" y1="32" y2="32" />
   <line x1="14" x2="14" y1="34" y2="34"/>
   <line x1="14" x2="12" y1="31" y2="27" />
   <text x="38" y="10" w="10" h="15" value="*id" A=”” />
</PT>
```

<!--  -->

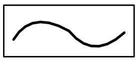
```
<CT id=""loc="" data="" show="" box="0,0 50,50" glue="23,7" >
   <circle cx="23" cy="21" r="14" />
   <text x="40" y="10" w="10" h="15" value="*id" A=”” />
</CT>
```

<!--  -->

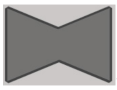
```
<SubstationMark id="" loc="" data="" show="" box="0,0 25,25" glue="" >
    < circle cx="12" cy="12" r=”10” />
    <text x="30" y="4" w="10" h="15" value="*id" />
  <DText id="*id"loc="x,y,w,h"data="#mRID" show="status"  A=””/>
</SubStationMark>
```

<!--  -->

```
<PowerPlantMark id="" loc="" data="" show="" box="0,0 60,40" glue="">
<rect  x="2" y="2" w="56"  h=”36” />
<path  d="M5,20 a7,7 1 0 14, 0"/>
<path  d="M33,20 a7,7 1 0 -14, 0"/>
<text  x="60" y="4" w="10" h="15" value="*id" />
<DText  id=”*id”loc=”x,y,w,h” data=”#mRID” show=”status” A=”” />
</ PowerPlantMark>
```

<!--  -->

```
<Turbine id="" loc="" data="" show="" box="0,0 80,70" glue="" >
<polygon points=”0,0 6,3 12,0 12,12 6,9 0,12” />
<text  x="85" y="4" w="10" h="15" value="*id"  A=””/>
</Turbine>
```

<!--  -->

```
<BusBarSection  id="" loc="" data="" show="" box="0,0 w,h" glue="">
<path  d="M0,0 w,0"/>
<text  x="60" y="4" w="10" h="15" value="*id" A=”” />
<text  x="60" y="20" w="10" h="15" value="V:" />
<DText  id=”*id” loc=”x,y,w,h” data=”#mRID” show=”status” />
</ BusBarSection>

<ACLineSegment  id="" loc="" data="" show="" box="0,0 w,h" glue="">
<path  d="M0,0 w,h "/>
<text  x="60" y="4" w="10" h="15" value="*id"  A=””  />
<text  x="60" y="4"  w="10" h="15" value="P:"  />
 <DText  id=”*id” loc=”x,y,w,h” data=”#mRID1” show=”status” />
<text  x="60" y="20" w="10" h="15" value="Q:"  />
```

```
<DText  id=”*id” loc=”x,y,w,h” data=”#mRID2” show=”status” />
</ ACLineSegment>

<DCLineSegment  id="" loc="" data="" show="" box="0,0 w,h" glue="">
<path  d="M0,0 w,h "/>
<text  x="60" y="4" w="10" h="15" value="*id"  A=””    />
<text  x="60" y="20"  w="10" h="15" value="P:"  />
<DText  id=”*id” loc=”x,y,w,h” data=”#mRID1” show=”status” />
<text  x="60" y="30"  w="10" h="15" value="I:"     />
<DText  id=”*id” loc=”x,y,w,h” data=”#mRID2” show=”status” />
</ DCLineSegment>

</defs>
```