Massimo Ancona, Walter Cazzola, Sara Drago, and Francesco Guido. An XML-Based Description of Structured Networks. In Proceedings of International Conference Communications 2004, *pages 401–406, Bucharest, Romania, June 2004. IEEE Press.*

An XML-Based Description of Structured Networks

Massimo Ancona* Walter Cazzola† Sara Drago* Francesco Guido‡

* DISI University of Genova, e-mail: {ancona,drago}@disi.unige.it

† DICo University of Milano, e-mail: cazzola@dico.unimi.it

‡ Marconi Selenia Communication, e-mail: francesco.guido@marconiselenia.com

Abstract

In this paper we present an XML-based formalism to describe hierarchically organized communication networks. After a short overview of existing graph description languages, we discuss the advantages and disadvantages of their application in network optimization. We conclude by extending one of these formalisms with features supporting the description of the relationship between the optimized logical layout of a network and its physical counterpart. Elements for describing traffic parameters are also given.

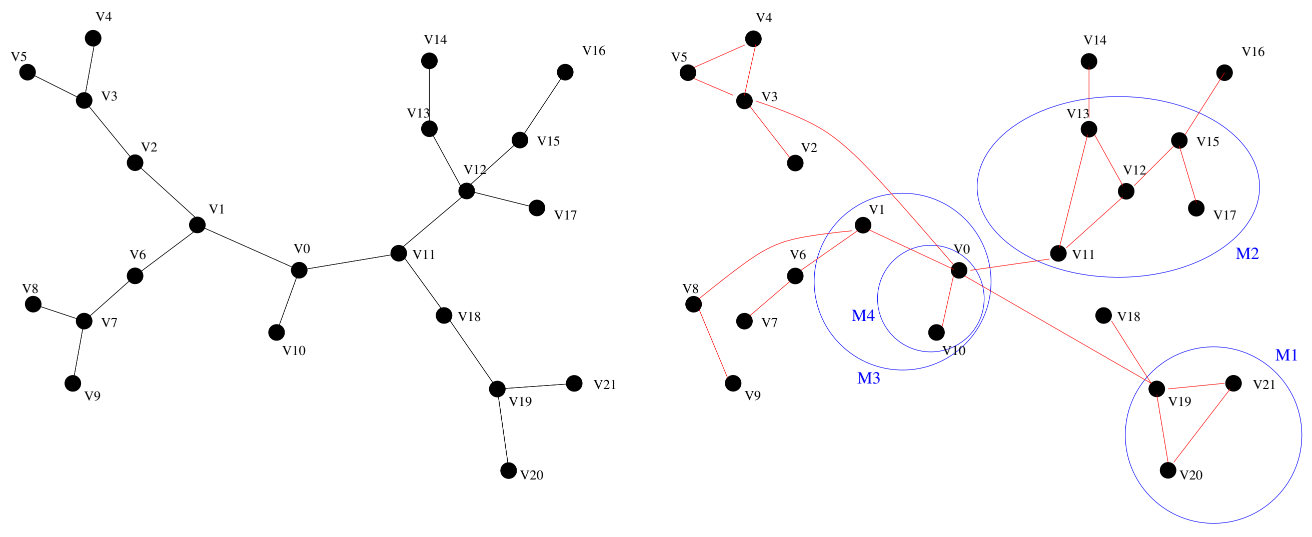
1 Introduction

The problem of network design and optimization has a strategical importance especially for military networks. In this case, design for fault tolerance and security plays a role more and more crucial than the equivalent importance required for commercial and research networks. The reengineering of a large communication network is a complex problem, which consists of different aspects that can be strongly affected by the way of describing data. The plainest way to describe a communication network is to model the relationship among sites and links by means of a weighted undirected graph, but unless some more assumptions are taken, this approach can raise several problems.

A first issue is encountered when an analysis and a visualization of the network has to be realized: practical networks include hundreds or often thousands of nodes and links, so that even a simple description and documentation of the network structure is hard to maintain and update. In this case the network is usually described in form of a hierarchy of sub-networks that are represented by collapsing some sub-networks into single nodes or single links to be described in separate documents. Such a hierarchical approach to network (and graph) description can be formalized into a complex but flexible graph structure called *structured graph* [2]. This representation is useful not only for documenting and drawing a large network but also for making computations on it without reconstructing the whole graph plan every time some computation has to be performed on it. Furthermore, the hierarchical representation allows to perform computations in compliance with the network organization.

A second issue is that a network is not a mere collection of sites and links with some capacity, but frequently some additional parameters are given, as the number of users relying on each site, the average traffic from a site towards the rest of the network, the links bit-rate and the request for special connections. Such additional information must be included in the computation, therefore a formalism to describe networks should provide labels and features to represent these parameters.

Finally, a last issue is related to the network optimization, i.e. the search for the logical network that makes the best use of the links capacity. The logical network defines how the switching elements of the physical network should be interconnected in order to minimize the number of times a transmission is processed. The problem of finding the logical network can be reduced to the problem of designing a graph over a given network, the vertices of which are the nodes of the network, and the edges of which represent *virtual paths* [3]; the virtual path layout is often found as the final result of a recursive steps sequence accepting as starting input the description of the physical network. The intermediate steps consist of subproblems close to the initial formulation, but involving smaller instances of a network, namely *clusters* [1]. Some more steps may be required to run capacity optimization algorithms. Keeping in mind this sequential approach, some kind of formalism to describe the input/output format is needed to re-establish the correct relationship which occurs between the result of a computation and the next one; moreover, such a tool should respect the historical succession of graphs obtained as intermediate results.



(a) physical network (b) logical network

Figure 1. An example of logical layout built over the physical network

Particularly, the description should be able to provide a mapping of the virtual networks resulting from an optimization algorithm onto the underlying physical networks.

From these issues is fairly evident that the mere flat graph formalism is unsuitable to describe all the features of a communication network and that the practice requires a language to describe hierarchies and traffic parameters. Moreover, the design of a description language based on a widely accepted standard for the exchange of data, as XML [8] is, promises to enhance the interoperability between reengineering tools. In the following, we will draw an overview of the existing XML-based formalisms for graph and network description and we will try to use them, or to extend them if necessary, in order to tackle some of the problems introduced by the examples.

2 XML descriptions for Communication Networks: State of the Art

One of the major achievement in the research for a unified approach in data description has been obtained by the World Wide Web Consortium with the definition of the eXtensible Markup Language (XML) [8]. The XML technology enhances the possibilities of software re-use and the interoperability between applications. Therefore, an XML description of data is effective in the optimization of a communication network.

Many graph description formalisms have been developed all along the Nineties, but all these languages do not support direct data exchange and none of them was expressly studied to describe networks. On the other hand, since the relationship among sites and links of a network is modeled by a graph, XML formats for graph description are a strongly related matter when a style for describing communication networks is proposed. Since 2000 the understanding of the relevance of an XML formalism for graph description has produced several efforts: XGMML [5], GraphXML [4], GRXL [6]; the one which best fits our purposes is *Graph Exchange Language* (GXL, [7]). GXL can be used to represent typed, attributed directed graph, and provides many facilities for representing labels on nodes and edges. The most interesting innovative feature in GXL is that its syntax includes a support for hypergraphs and hierarchical graphs; for this reason GXL turned out to be a useful description formalism to enhance the interoperability between structured communication networks reeingeneering tools.

The physical network showed in Figure 1.a can be easily described by a GXL document. According to the GXL syntax, each node or edge may have a child element `type` which points to an appropriate schema providing additional information on the role and the meaning of the parent element. Such a reference mechanism could be helpful when there is the will to develop applications for networks displaying. When required by the network features, it is possible to represent oriented links (this information is stored in the `isdirected` attribute within edge tags). In the following, we look at an excerpt of a document showing how GXL could be used to describe the logical network in Figure 1.b

```
<?xml version="1.0"?>
<!DOCTYPE gxl SYSTEM "http://www.gupro.de/GXL/gxl-1.0.dtd">
<gxl xmlns:xlink="http://www.w3.org/1999/xlink">
  <graph id="logical-network" edgeids="true" hypergraph="false" edgemode="defaultdirected">
    .....
     <node id="M3">
        <graph id="graph3" edgeids="true" hypergraph="false" edgemode="defaultdirected">
           <node id="M4">
              <graph id="graph4" edgeids="true" hypergraph="false" edgemode="defaultdirected">
                 <node id="V0"/>
                 <node id="V10"/>
                 <edge id="vp12" from="V0" to="V10" isdirected="false"/>
                 ...
                 <edge id="vp15" from="V19" to="V0" isdirected="false"/>
              </graph>
           </node>
           <node id="V1"/>
           ...
        </graph>
     </node>
     ...
  </graph>
</gxl>
```

Here the possibility of exchanging hierarchically structured graphs plays the main role. This is the consequence of a syntax that makes possible for nodes to have `graph` elements as children. Edges contained in a subgraph are represented as usual, while edges crossing different levels of the hierarchy are itemized within the most internal subgraph containing one of the incident vertices. An analogous feature allows to build edge-structured graphs. In the example below, we use this last feature to map every virtual path in the logical network to the corresponding path in the physical network.

```
<edge id="vp13" from="V3" to="V0" isdirected="false">
  <graph>
    <edge id="e15" from="V2" to="V3" isdirected="false"/>
    <node id="V2"/>
    <edge id="e14" from="V1" to="V2" isdirected="false"/>
    <node id="V1"/>
    <edge id="e2" from="V0" to="V1" isdirected="false"/>
  </graph>
</edge>
```

In this fragment of GXL code, the physical path is described as a subgraph of the corresponding edge in the logical network, but practically it is part of another graph (the physical network).

Nevertheless, this solution is not the best practicable, at least for two reasons: first GXL nodes and edges are identified by their `id` attribute, which is of type `ID` and cannot be declared more than once within the same document, according to the XML specification. Then, since replicas of the same element are not allowed in a valid document, only topologies with physical links hosting at most one virtual path can be described. Secondly, from a semantic point of view virtual paths and physical links are entities belonging to different domains. It should be inferred from the document that subgraphs of an edge belong to a parent graph described elsewhere, but this is not plain, because the scope of edge and node `id`s does not go beyond the current document. Hence, it would be preferable to embed the logical network in the physical layout without mixing the representation of physical and logical links in the same document. As a matter of fact, we cannot derive this kind of description from the current release of GXL. On the other side, among all the XML formalisms for graph representation, GXL turned out to have the most suitable predefined features to describe communication networks, such as the facility for representing hierarchies on nodes, that is helpful for managing large networks in a structured way. Moreover, GXL makes the insertion of user defined tags easier by a set of empty macros that might be specified in order to obtain specializations of the language. Then, our idea is to lever the GXL extensibility to find an adequate description of how the bundle of virtual paths embeds itself in the physical network. Particularly, we are interested in adding tags for re-establishing the correct relationship between an optimization step and the following one; in addition it is important to overcome the GXL lack of predefined *structs* by providing new features explicitly designed to describe the traffic information.

3 Network Exchange Language (NXL)

In the previous section we outlined some problems occurring when we try to represent the optimization intermediate results sequencing in GXL. The main argument was that, even if GXL supports hierarchies on edges, the possibility to fold the actual routing in a virtual link produces a semantically ambiguous document. In addition, the same physical link is frequently involved in providing a support to different virtual paths and we need to express a measure of how much it is committed to serve each of them (e.g. by declaring a percentage of bandwidth). One solution could be to equip the edge with an attribute linking to another graph describing the actual routing. This way, the physical path would not be expressed as a subgraph but as a reference, and the virtual graph and the physical one would be treated as separate entities. Unfortunately, this idea in practice could not apply on large networks, because of an unconsidered proliferation of references pointing to subgraphs of the physical network. This would result in a waste of disk space, as the collection of documents would carry no more information than the whole physical network itself. However, such a reference mechanism could be effective in some cases and we introduce it as a discretionary alternative to explicit hierarchical nesting. A more suitable approach could consist of introducing new tags explicitly tailored to represent how the virtual network resulting from an optimization algorithm is mapped onto the underlying physical network. At the same time, it is necessary to design edge attributes to represent links capacity and bandwidth allocation percentages.

```
<vmap id="map1">
  <realref xlink:href="physicalnet.xml"/>
  <virtref xlink:href="logicalnet.xml"/>
  <vpath id="e1">
    <from xlink:href="physicalnet.xml#V13" xlink:show="new" xlink:actuate="onRequest"/>
    <to xlink:href="logicalnet.xml#V19 xlink:show="new" xlink:actuate="onRequest"/>

    <phyelem="physicalnet.xml#e8" xlink:show="new" xlink:actuate="onRequest">
        <bandwidth>30,/bandwidth>
    ...
  </vpath>
  <vpath id="e2">
  ...
  </vpath>
</vmap>
```

vmap is a new element designed to describe the mapping between two documents. The vpath tag originates from the virtual path model as it is presented in [3]; in the same perspective, the sequence of phyelems implements the idea of *induced path* and refers to edges in the physical network. In other words, the sequence is responsible for describing both routing and percentages of allocated bandwidth. The bandwidth element is a basic facility for dealing with communication networks, more tags describing values could be introduced to the double goal of providing the physical network description with a detailed information on its link capacity (e.g. number of TDM frameworks, slots, frequency sampling, etc) and of specifying how these resources are exploited by the virtual paths. Moreover, we need to enrich GXL such that the traffic input can be described and we make available keywords designed to this goal.

4 From GXL to NXL: Implementative Details

The GXL Document Type Definition (DTD) is equipped with a list of predefined ”empty” macros for extending information to GXL documents. These extension points consist of parametric entities to which elements and attributes refer within their definition and can be used to add subelements or attributes to the corresponding graph components. The rest of the DTD gives the syntax for graph components, attributes and references. The parametric entities might be redefined (by adding specifications in quotation marks instead of their empty body) in order to obtain specializations of GXL. Using this mechanism results in a DTD that deviates from the predefined GXL standard. In the following, we consider the problem of producing a template for the mapping istance informally described in the previous section and we extend DTD parametric entities to introduce new tags to represent virtual paths and the corresponding routing (instead of reporting macros redefinition, we prefer to enhance readability by rewriting the whole graph definition):

```
<!ELEMENT graph(type?,attr*,(node|edge|vpath|rel)*)>
<!ATTLIST graph
 id ID #REQUIRED;
 %graph-attr-extension;
>
<!ELEMENT node(type?,attr*,(graph|graphref)*)>
<!ELEMENT edge(type?,attr*,(graph|graphref)*, capacity*)>
<!ATTLIST edge
 id ID #REQUIRED
 from IDREF #REQUIRED
 to IDREF #REQUIRED
 fromorder CDATA #IMPLIED
 toorder CDATA #IMPLIED
 isdirected (true|false) #IMPLIED
>
<!ELEMENT graphref EMPTY>
<!ATTLIST graphref
 xlink:type (simple) #FIXED "simple"
 xlink:href CDATA #REQUIRED
>
<!ELEMENT capacity(int)>
<!ELEMENT vmap(realref,virtref,vedge*)>
<!ATTLIST vmap id ID #REQUIRED;>
<!ELEMENT realref EMPTY>
<!ATTLIST realref
 xlink:type (simple) #FIXED "simple"
 xlink:href CDATA #REQUIRED
>
<!ELEMENT virtref EMPTY>
<!ATTLIST virtref
 xlink:type (simple) #FIXED "simple"
 xlink:href CDATA #REQUIRED
>
<!ELEMENT vpath(from, to,phyelem*)>
<!ATTLIST vpath id ID #REQUIRED;>
<!ELEMENT from EMPTY>
<!ATTLIST from
 xlink:type (simple) #FIXED "simple"
 xlink:href CDATA #REQUIRED
 xlink:show (new|replace|embed|other|none) #IMPLIED
 xlink:actuate (onLoad|onRequest|other|none) #IMPLIED
>
<!ELEMENT to EMPTY>
<!ATTLIST to
 xlink:type (simple) #FIXED "simple"
 xlink:href CDATA #REQUIRED
 xlink:show (new|replace|embed|other|none) #IMPLIED
 xlink:actuate (onLoad|onRequest|other|none) #IMPLIED
>
<!ELEMENT phyelem(bandwidth)>
<!ATTLIST phyelem
 xlink:type (simple) #FIXED "simple"
 xlink:href CDATA #REQUIRED
 xlink:show (new|replace|embed|other|none) #IMPLIED
 xlink:actuate (onLoad|onRequest|other|none) #IMPLIED
>
<!ELEMENT bandwidth(int)>
```

We reported only the definitions that deviate from the standard GXL. In most cases, we omitted for sake of brevity the extensible macros (or better, we think of them in the act of showing their effect on the DTD and we rewrite their content). Particularly, `node` and `edge` have been provided with a `graphref` child element to grant the option of referencing another graph instead of describing it as a subgraph and `edge` has been equipped with a child element to contain information about its capacity (this is a useful feature when physical networks are described and more children elements might be inserted in the same way to specify some more tecnical information about links). `vmap` is a special element assuming that the physical network and the logical one have been described somewhere else (linked through `realref` and `virtref`) and its goal is to provide a mapping between these two entities, but it can be used also to state the relationship between an optimization step and the following one. Routing can be istanced through `vpath`, whose children elements `from` and `to` refer to nodes in `realref` and `phyelem` child element points to an edge in `realref`. References to nodes and edges are implemented by adding a # and the id of the addressed element as a suffix to the URI of the linked XML file [10]. `show` and `actuate` attributes for `from`,`to` and `phyelem` elements are used to communicate to a displaying application the desired presentation of the remote fragment of document: whether the application should load it in a new window and under which post-loading triggering event (e.g. when a user clicks on the reference) [9]. Now we consider the problem of describing traffic information and we aim to make possible a description for storing data about communication throughput of a node in the network; practically, we extend node parametric entities to introduce new tags for the representation of traffic input in a communication network (again, for readability we rewrite the whole node definition):

```
<!ELEMENT node(type?,attr*,graph*, communications)>     <!ELEMENT outgoing-traffic(int)>
<!ATTLIST node                                          <!ELEMENT traffic(throughput)>
 id ID #REQUIRED;                                       <!ATTLIST traffic destination IDREF #REQUIRED>
 reg ID #REQUIRED;                                      <!ELEMENT ptp(throughput)>
 level (NORMAL|HIGH|TOP) #REQUIRED                      <!ATTLIST ptp destination IDREF #REQUIRED>
>                                                       <!ELEMENT throughput(int)>
<!ELEMENT communications(users, outgoing-traffic, traffic*, ptp*)>
<!ELEMENT users(int)>
```

The two DTD fragments above merge into a unique DTD which leads to the definition of a new language for exchanging networks.

5 Conclusions

The adopted approach pivots on existing XML-based languages facilities for describing hierarchical graphs and aims to re-use hierarchies to define a formalism for structured communication networks. The result is a new language for exchanging networks (NXL), that extend GXL by including many innovations designed to manage traffic information and capacity features of communication networks. The main extension is an element representing the description of the relationship which occurs between the topology of a network and its optimization, which plays an important role in providing an easier handling of the optimization process. NXL does not overcome a drawback of GXL that becomes evident when it is necessary to create manually files storing large networks, that is the unavoidable insertion of long lists of edges and nodes itemized without a predefined logical organization. This happens because nothing like an adjacency list is supported up to now by the language. A possible solution is the definition of a dummy partition (when a preexisting organization in subnetworks is not available) to divide the creation in steps that can be handled in a structured way with the hierarchical mechanism. On the other side, this language presents all the advantages of existing XML technology in parsing, extracting and displaying data, such as DOM applications and cascading style sheets (CSS). In addition, the use of scripts for implementing preliminary checks over networks (e.g.bandwidth limitations and feasibility of a virtual configuration) provides a way to decrease the computational cost of the subsequent optimization steps.

References

[1] M. Ancona, W. Cazzola, P. Raffo, and I. B. Vasian. Virtual Path Layout Design Via Network Clustering. In *Proceedings of International Conference Communications 2000*, pages 352–360, Bucharest, Romania, on 7th-9th of Dec. 2000. IEEE.

[2] M. Ancona, L. De Floriani, and J. S. Deogun. Path Problems in Structured Graphs. *The Computer Journal*, 29(6):553–563, June 1986.

[3] O. Gerstel, I. Cidon, and S. Zaks. Optimal Virtual Path in ATM Networks with Shared Routing Table Switches. *Chicago Journal of Theoretical Computer Science*, Oct. 1996.

[4] I. Herman and M. S. Marshall. GraphXML – An XML Based Graph Interchange Format. Technical report, Apr. 2000.

[5] J. Punin and M. Krishnamoorty. Extensible Markup and Modeling language (XGMML) Draft Specification. Technical report, XML.org, 2001.

[6] P. Rodgers. An XML Specification for Grrr Programs. Technical report, Sept. 2000.

[7] A. Winter. Exchanging Graphs with GXL. LNCS 2265, pages 485–500, Sept. 2002.

[8] World Wide Web Consortium. Extensible Markup Language (XML) 1.0. W3C Recommendation available at `http://www.w3.org/TR/REC-xml`, Feb. 1998.

[9] World Wide Web Consortium. XML Linking Language (XLink) Version 1.0. W3C Recommendation available at `http://www.w3.org/TR/xlink/`, June 2001.

[10] World Wide Web Consortium. XPointer Framework. W3C Recommendation available at `http://www.w3.org/TR/xptr-framework/`, Mar. 2003.