CAA
2017

Pierre-Yves Buard et al.
http://dx.doi.org/10.15496/publikation-43222

Publishing an Archeological Excavation Report in a Logicist Workflow

Pierre-Yves Buard
pierre-yves.buard@unicaen.fr

Elisabeth Zadora-Rio
e.zadora.rio@gmail.com

Jérôme Chauveau
jerome.chauveau@unicaen.fr

Julia Roger
julia.roger@unicaen.fr

Olivier Marlet
olivier.marlet@univ-tours.fr

Abstract

The logicist programme, which was initiated in the 1970s by J.C. Gardin, aims: first, to clarify the reasoning processes in the field of archeology; second, to explore new forms of publication, in order to get over the growing imbalance between the flood of publications and our capacities of assimilation. The logicist programme brings out the cognitive structure of archaeological constructs, which establish a bridge between empirical facts, or descriptive propositions, at one end of the argumentation, and interpretative propositions at the other end. This condensation process opens the way for alternative forms of publication, designed to speed up the assimilation of the chain of inference and the consultation of the database on which it stands. In this paper we propose a new publishing workflow respecting the principles of the logicist programme. We show how texts are encoded using XML markup in accordance to Text Encoding Initiative (TEI) recommendations. We explain how the relations between propositions are marked-up as hypertext references with simple qualification. Next, we describe how to extract the overall organization of the interpretation process from the XML tree as RDF triple by extrapolating from relations' links. We also show how to produce an overview diagram representing the interpretative process.

Keywords: archeological constructs, logicist workflow, XML, TEI, RDF

Introduction

It is now widely recognized that the quantity of pages currently published in our areas of research is such that we are unable to read more than a very small fraction of the literature relevant to our research interests. Instead, we consult some of it, following our own selection strategies. The paradox is that while we are perfectly aware of this phenomenon, we continue to write as if our works were to be read, without any attempt to redraft them for the alternative perspective, that is consultation. Digital publishing as such does not solve the problem; it makes it worse.

The logicist programme, which was initiated in late 1970s by Jean-Claude Gardin (Gardin 1979; Gardin 2003), was developed with a twofold aim: first, to bring out the logico-semantic structure of interpretative constructs in archaeology, in order to clarify the reasoning processes; second, to explore alternative forms of publication in order to overcome the growing imbalance between the flood of publications and our assimilation capacities. The logicist programme brings out the cognitive structure of archaeological constructs, which establish a bridge between empirical facts, or descriptive propositions, at one end of the argumentation, and interpretative propositions at the other end. This condensation process opens the way for alternative forms of publication, designed to speed up the assimilation of the chain of inference and the consultation of the database on which it stands.

The aim of this paper is to bring forward a new publishing workflow based on the principles of the logicist programme. We first show how the texts are

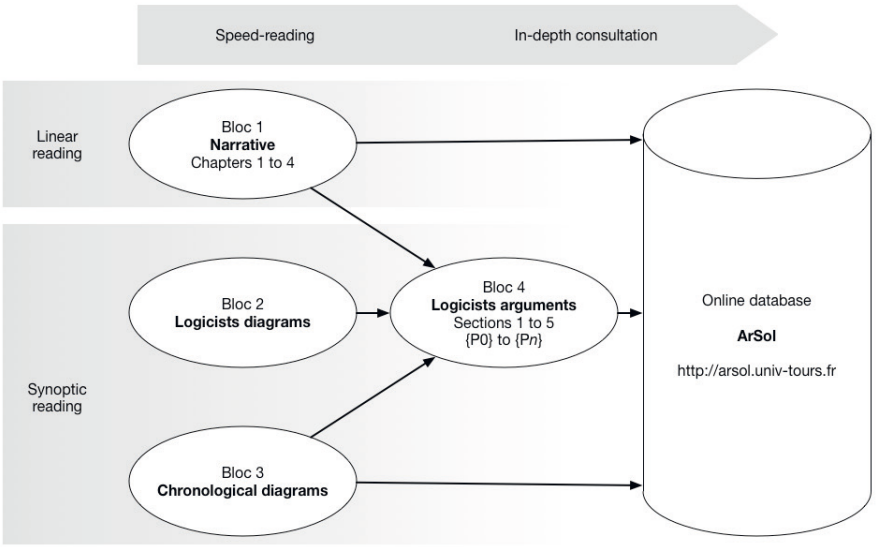

Figure 1. The logicist approach of the publication of the excavation of Rigny.

encoded using XML markup in accordance with (TEI) recommendations, and then how the relations between propositions are expressed as hypertext reference markup with simple qualification. Next, we describe how the overall structure of the interpretative construct, represented by an inference diagram, is extracted from the XML (eXtensible Markup Language) tree as RDF triple by extrapolating from the interrelation links. The digital publication of the archaeological excavation of the settlement and church in Rigny (Indre-et-Loire, France) (Zadora-Rio, Galinié et al. forthcoming) is used as a test-case to show that our workflow can provide different levels of information retrieval, allowing both speed-reading and in-depth consultation.

The Logicist Programme: A Reminder

The logicist programme is a long-term research project launched by Jean-Claude Gardin (Gardin 1979; Gardin 2003). Archaeological constructs are considered in the logicist approach as computational structures made up of two constituents: 1) a data base, i.e. a set of declarative propositions {P*0*} generally relating to empirical facts; 2) an inference tree expressing the steps leading from the initial set of propositions {P*0*} to the conclusions {Pn} through a succession of leaps {P*i*} => {P*j*} from one or several levels of the inference tree to the next. Such a tree can be read in two alternative directions: empirico-inductive, from the database {P0} to the conclusions {P*n*}, or hypothetico-deductive, from the hypotheses {P*n*} to the database {P*0*}. The logicist framework helps to apprehend the overall organization of the interpretation process and to consult readily some of its parts without having to go through lengthy presentations in standard archaeological discourse.

The First Editorial Developments in the Field of the Archeology of Techniques

The first editorial developments emerged in the early 2000s, with the creation of a new collection under the title "Référentiels" by the Éditions de la Maison des sciences de l'Homme in Paris, in partnership with "Épistèmes", a private firm founded by Philippe Blasco. The first three publications, in the field of the archeology of techniques, were designed as hybrids: a printed book with a CD-ROM containing its cognitive substance (data and inference tree) in a multi-media format based on the logicist framework (Scientific Constructs and Data), designed by Valentine Roux and Philippe Blasco (Roux 2009; Roux and Blasco 2004). The creation in 2007 of an international online journal, The Arkeotek Journal, and the launching of the Arkeotek project in the domain of the archaeology of techniques, under the direction of Valentine Roux, aimed at the developing of "logicist corpuses" made of documents structured in data and interpretation rules (Gardin and Roux 2004; Roux and Aussenac-Gilles 2013).

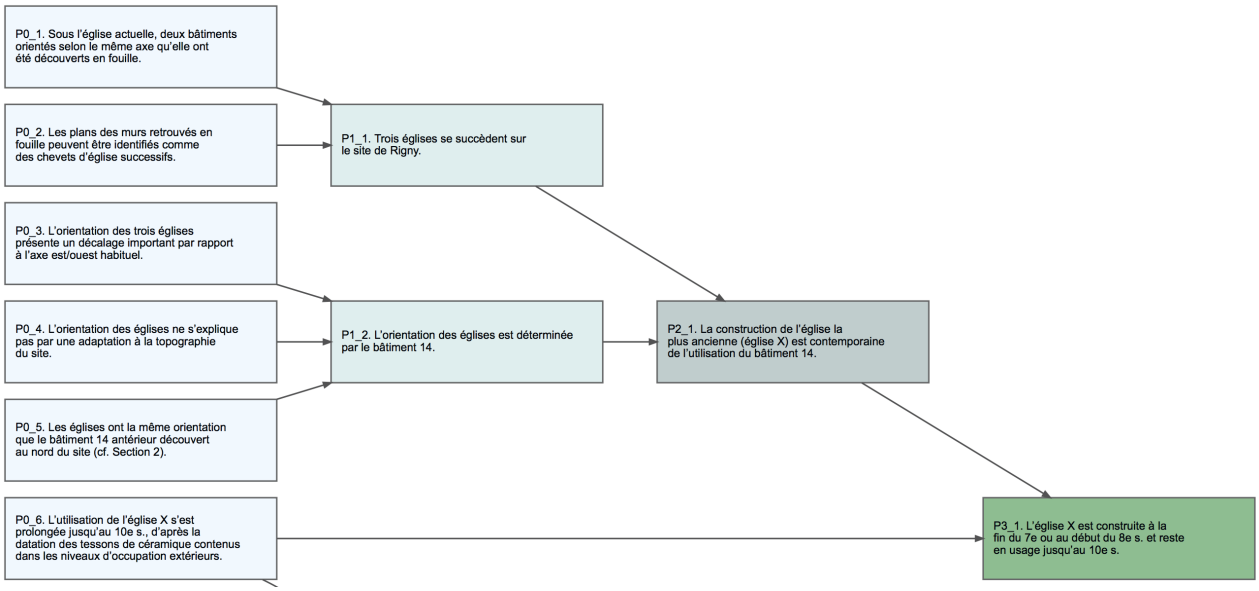

Figure 2. A fragment of a logicist diagram.

Applying the Logicist Framework to the Excavation of the Settlement and Church in Rigny

The results of an archeological excavation are the outcome of a cognitive process which is quite different from that of the archeology of techniques, since neither the selection nor the analytical framework of the corpus can be predetermined. The interpretation of the excavation data primarily consists in identifying the archeological remains ("what is there?") by ascribing to them a date, a function, and a possible morphological reconstruction. The next step consists in reconstructing the sequence of activities and events (including site formation processes) in order to answer to the question: "What happened there?" and to examine its implications in a broader historiographical context. The reading of published excavation reports is especially unrewarding and painstaking because of the lengthy descriptions of the remains, motivated by the irreversible character of excavation activities, and difficult access to the primary documentation. The need to rethink the form of archeological publications emerged in the 1970-1980's, but it is only the development of the Internet which allowed the recent online publications of databases and site records, thus opening new perspectives and possibilities.

The publication of the archaeological excavation of Rigny represents a new experiment in the "Référentiels" collection, not only because it will be the first one dealing with the results of an excavation, but also because it will be the first one to be entirely digital (Zadora-Rio, Galinié et al. forthcoming). The publication is divided into four interconnected sections, each one representing a possible access point to the content (Figure 1). The three sections in the left column ("Block 1" to "Block 3") represent different forms of speed-reading while the fourth one in the middle ("Block 4") contains a comprehensive list of all the statements, from the empirical facts (or initial propositions {P*0*}) to the conclusions (or final propositions {P*n*}).

The "narrative" ("Block 1"), which contains the methodological and historiographical considerations together with a rapid and linear outline of the results, is connected by hypertext links to the "logicists arguments" from {P*0*} to {P*n*} ("Block 4"). It is designed for speed-reading, but it also allows for the assessment of the argumentation and the retrieval of the data on which it is based if the reader chooses to follow the links. The "logicist diagrams" ("Block 2") give an overview of the argumentation presented in the form of an inference tree displayed from left to right. The hierarchical level of the propositions is indicated by the colourmap, the lightest colour being used for the initial set of propositions {P*0*} and the darkest one for the final propositions (or conclusions) {P*n*} (Figure 2). The initial propositions, or {P*0*}, on the left of the screen, list all the data supporting the interpretation. The following steps, from {P*1*} to {P*n*}, do not introduce new data, but repre-

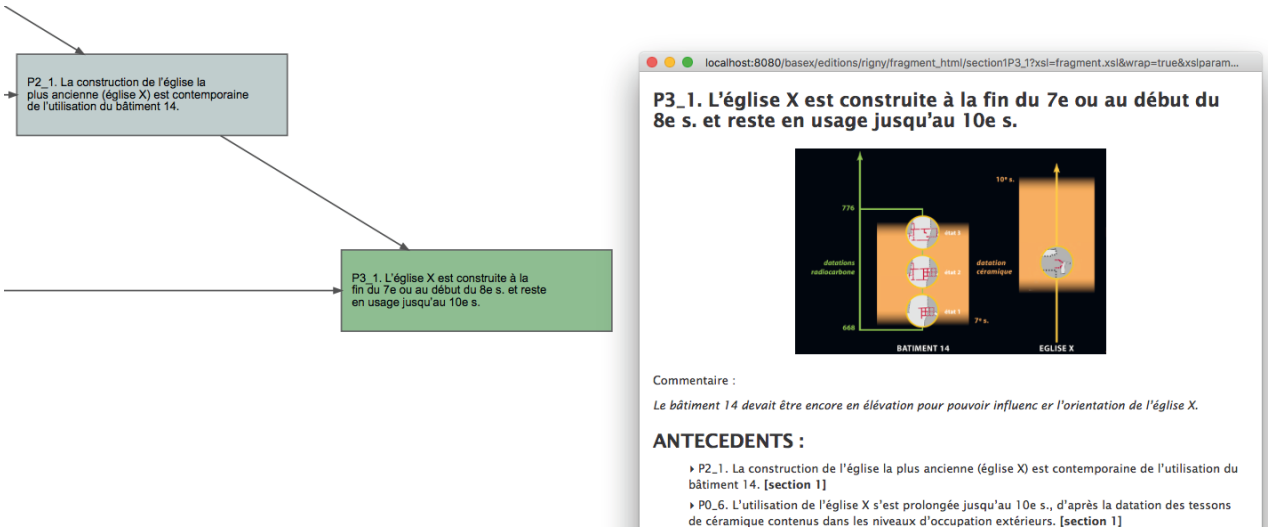

Figure 3. Retrieving the detailed arguments from the logicist diagram.

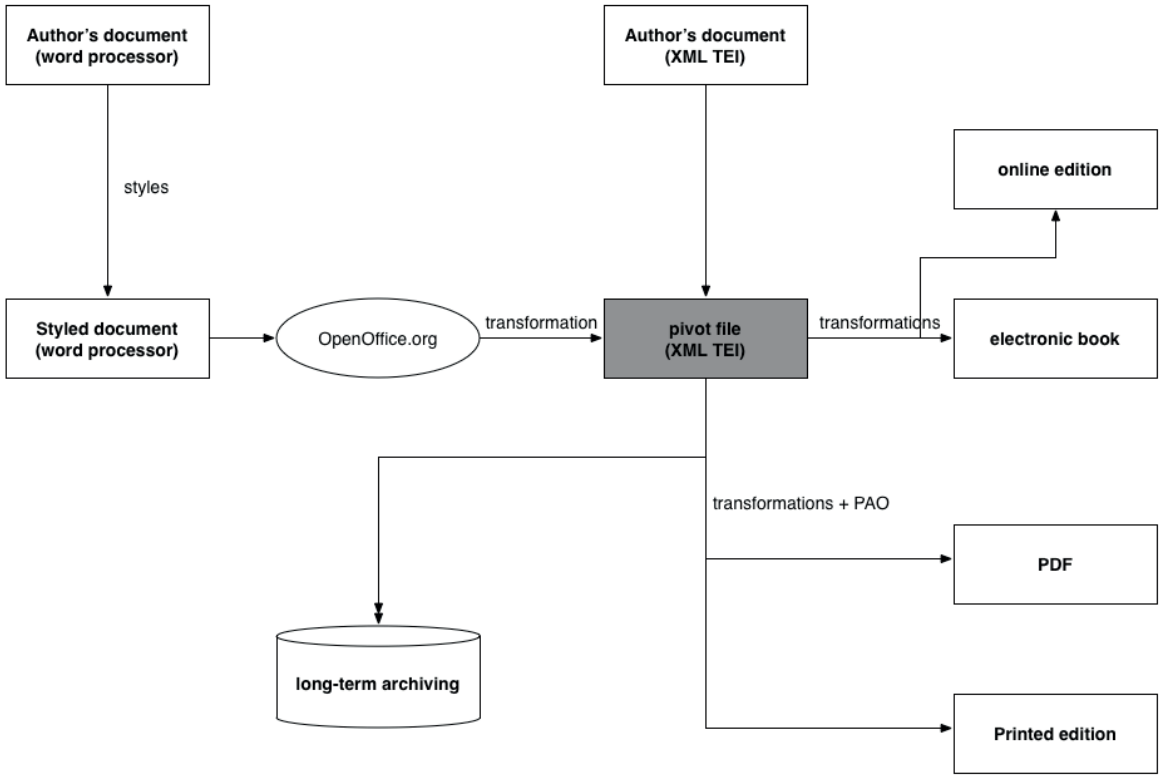

Figure 4. The Single Source Publishing model of the Pôle document numérique and the Caen University Press.

sent inference operations based on the combination of lower level propositions.

Diagrams are interactive: by selecting an intermediate proposition {P*i*} or a final proposition {P*n*}, the lower level propositions on which it is based, are also selected. As shown in Figure 3, a click on one of the boxes of the inference tree enables the access to the detailed argumentation ("Block 4"), which contains the supporting data, and also to the anterior propositions if it is an intermediate proposition {P*i*} or a final proposition {P*n*}. The logicist diagrams thus clarify the reasoning processes, and they also form an access mode especially suitable for consultation.

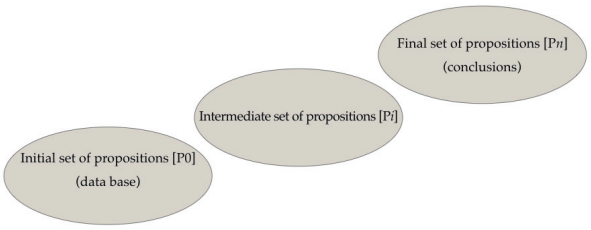

Figure 5. The reasoning process from empirical facts to conclusions.

The Single Source Publishing Model

Publishers have been dealing with multiple media for some years now, which means they need the same text to be readable as a chapter of a printed book, as a webpage, or as a part of an electronic book. In order to produce these multiple versions of a text, publishers are now increasingly inclined to work on a single file that is processed to fit multiple reading media. This method is known as the Single Source Publishing model (SSP) and within this editorial process, from a technical point of view, XML is commonly used to encode the source file.

The *Pôle document numérique* (http://www.unicaen.fr/recherche/mrsh/document_numerique/) and the Caen University Press have built a full SSP workflow based on XML technology, in conformity with the TEI recommendations (http://www.tei-c.org/). Figure 4 gives a general view of the process, leading from the author's file to the multiple reading media.

The framework of the logicist workflow is very close to the SSP model and shares the same work organization (shown in Figure 4). The main difference is that we are dealing with two types of files. The first one is the "narrative", which is, in fact, very close to a classical scientific article. The second one is a set of documents that contains the logicists arguments. These documents contain the statements (or propositions) and the pointers between them. The statements summarize the successive steps leading from one or several levels of the inference process to the next, without any rhetorical apparatus. Pointers between statements provide the links that connect them. The logico-semantic structure of interpretative constructs is thus represented by a sequence of inferences from the initial set of empirical facts, or descriptive propositions {P0}, to the conclusions, or interpretative propositions {Pn} put forward by the author, as shown in Figure 5.

Both the narrative and the logicist arguments are encoded as XML TEI files. While the narrative, which is written in natural language, does not display any special features, the logicist arguments require a specific treatment: whereas the text is encoded in a traditional way, the relations between the statements are encoded as pointers. The section below on "Building the logicist inference tree" gives a detailed presentation of the methods and tools used to develop a user-friendly environment.

For the time being, from a computer science point of view, XML gives the best ratio between expressivity and implementation complexity. It is a format which is extensively used by publishers' in their workflows all around the world. In fact, XML comes with most document types, either natively, or as an imported feature in word processors (.odt, .docx, etc.), on the web (XHTML, HTML5, etc.), in Adobe Indesign, and in epubs. It is a format that has become a real standard when it comes to publishing.

From Texts Flows to Graphs

Whereas XML gives excellent prospects and results in the field of publishing, it lacks expressivity when it comes to building complex relationships and qualifying links between multiple nodes. However, it is possible to express the structure of the argumentation and to prepare the extraction of a network made of all the propositions with the links between them by using some simple XML annotations. Figure 6 gives an XML annotation of a very basic bibliographic reference using the TEI vocabulary. The upper part of the figure displays a bibl element containing three other elements: a *title*, an *author* and a *publisher*. The lower part of the figure gives the strict XML point of view: the "contains" relation is, in fact, the only explicit relation given by the XML tree.

But what we, as humans, really understand about the encoded information within the tree is much more complex. In fact, anyone familiar with XML technology and TEI recommendations understands that we are dealing with a bibliographic reference, the title of which being represented by the text inside the *title* element, the *author* by the text inside the au-

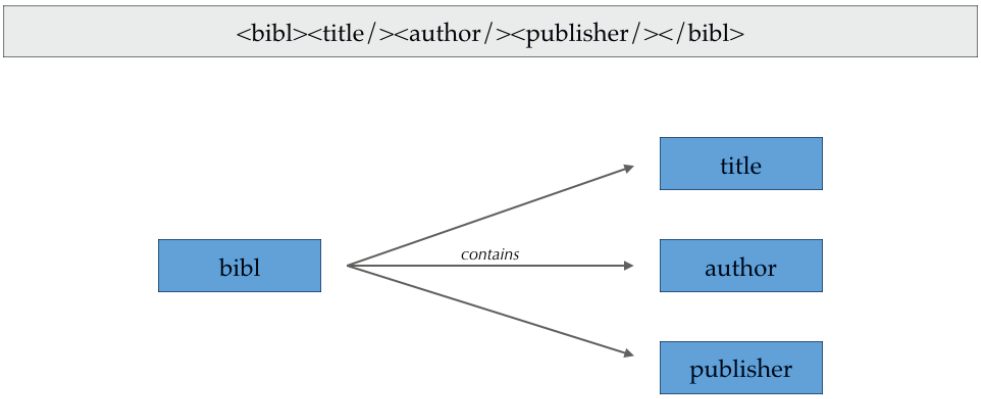

Figure 6. An example of what XML actually says about the data. 6. An example of what XML actually says about the data.

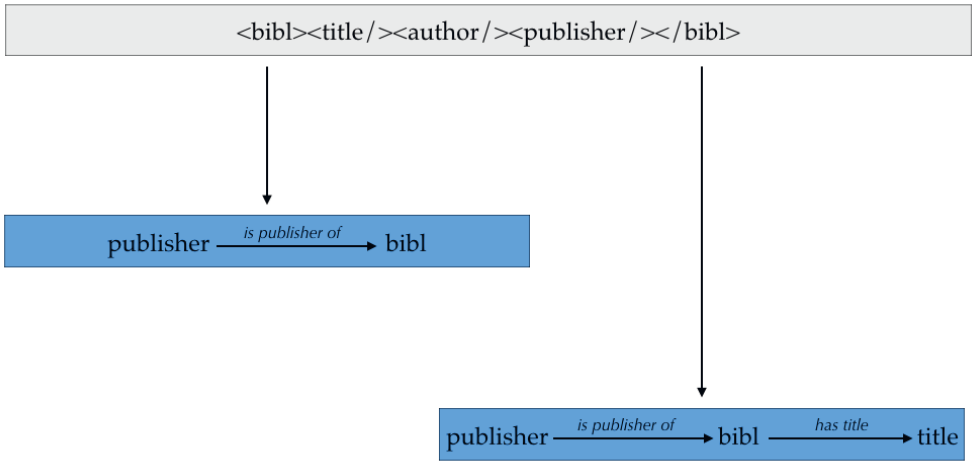

Figure 7. An example of what XML really means about the data.

thor element and the publisher by the text inside the *publisher* element.

Figure 7 shows examples of the interpretation of the XML tree or, in other words, what we, as humans, really understand about the encoded text. The "*contains*" relation between the *bibl* element and the *title* element is actually interpreted as some kind of a "*has title*" oriented relation between the bibliographic reference and the title.

We follow exactly the same pattern to exploit the XML TEI annotations in order to produce the inference tree or interrelationship diagram during the editing process. From a technical point of view, we need to transform the XML tree into an RDF graph to enhance the expressivity of the data set. The interrelationship diagram gives an overview of the interpretative construct, but it also provides an advanced search solution exploiting not only the nodes, or the textual elements, but also the edges, or the relations between textual elements.

Figure 8 provides an overview of the logical process that goes from the XML tree to the RDF graph.

Each proposition is encoded as text division, using the dedicated TEI *div* element with a specific value affected to the *subtype* attribute and a unique identifier, at the file scale, stored in the *xml:id* attribute. Inside this *div* element is another *div* element containing all the links between the current proposition and its antecedents. Each antecedent takes the shape of a pointer element (*ptr*) with a *target* attribute storing the unique identifier of the targeted proposition. The *subtype* attribute is used to store the type of relation we want to draw between the two propositions. In this example the markup builds two relations of the type "is_based_on" between the div identified by "section1P1_1" and the propositions identified by "section1P0_1" and "section1P0_2" which are not visible here, but are stored elsewhere in the XML file.

Using this simple markup for a complete XML tree gives the basis for a full representation of the interpretative construct. The attribute system thus produces some kind of layer, which is added to the XML tree and draws the links between the nodes. In

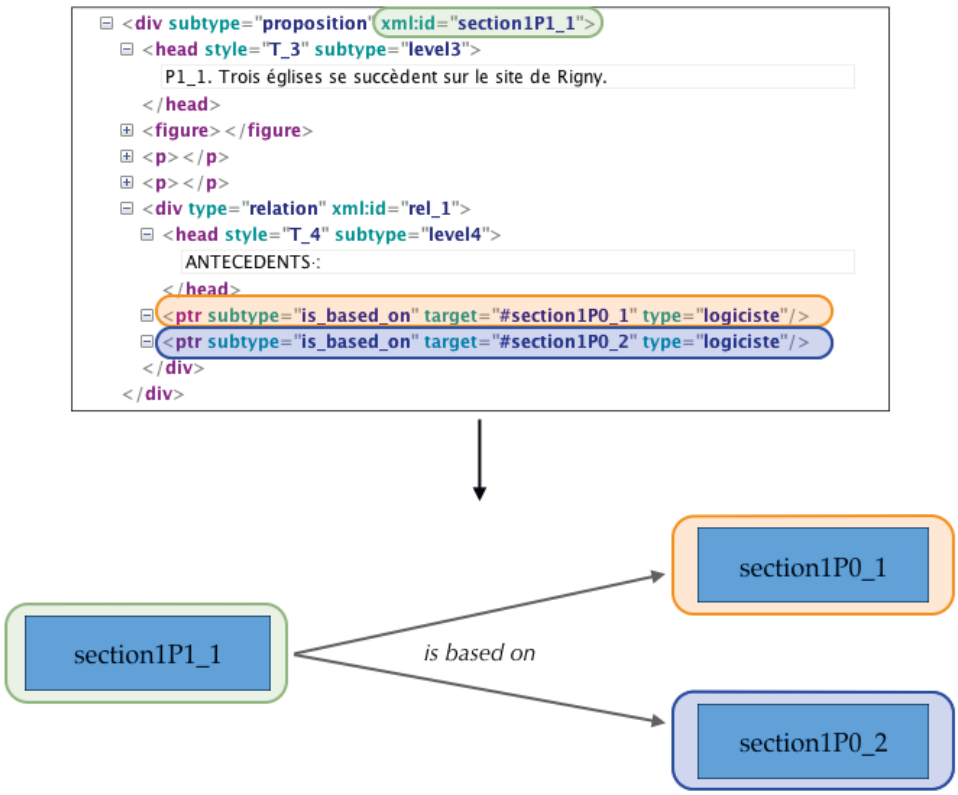

Figure 8. From XML tree to RDF graph.

the following step in the workflow, this network will be parsed to produce the logicist diagram.

Building the Logicist Inference Tree

As we have seen, it is possible to build relations between nodes in an XML tree by using pointers with specific attributes. In order to produce this markup, we developed a dedicated Graphic User Interface (GUI) module for XMLMind XML Editor software (http://www.xmlmind.com/) to allow the archeologist to build the logicist interrelations in a user-friendly environment. This module enables the user to build the relations between the statements belonging to different levels in the inference tree.

The basic pattern of the module is based on the building of a dynamic list of all the propositions that compose the document, sorted by interpretative level. We use the XMLMind XML Editor "commands" system (http://www.xmlmind.com/xmleditor/_distrib/doc/commands/index.html), which is entirely XML, in combination with XSLT (https://www.w3.org/TR/2017/REC-xslt-30-20170608/) (eXtended Stylesheet Language Transformation) and CSS (Cascading Style Sheets) (https://www.w3.org/Style/CSS/) for the text layout. Each node of the XML TEI tree has a form affected by the CSS. Every interpretative proposition, i.e. *div* elements with a *type* attribute associated with each "proposition" and a level with a value superior to 0, is associated with dynamic buttons. To add a relation between an interpretative proposition *A*, which is the subject of the relation, and another one anywhere else in the text, which is the object of the relation, the user has to click on the button associated with proposition *A* to launch the process.

All the steps in this process are parts of a command of the XMLMind XML Editor. The first step is to produce the list of all the propositions by applying a XSLT to the XML document. The XSLT parses the XML document and transforms the propositions into a list that is stored in a flat text file. The second step is to present this list to the user in a dedicated window, so that he can pick the right proposition. XMLMind XML Editor provides a specific *pick* command to do this. When the user chooses the right proposition by clicking on it in the list provided by the *pick window*, a specific XML *pointer* element, ptr with a target attribute, is inserted into the proposition A (See section "From text flows to graphs" above for a complete

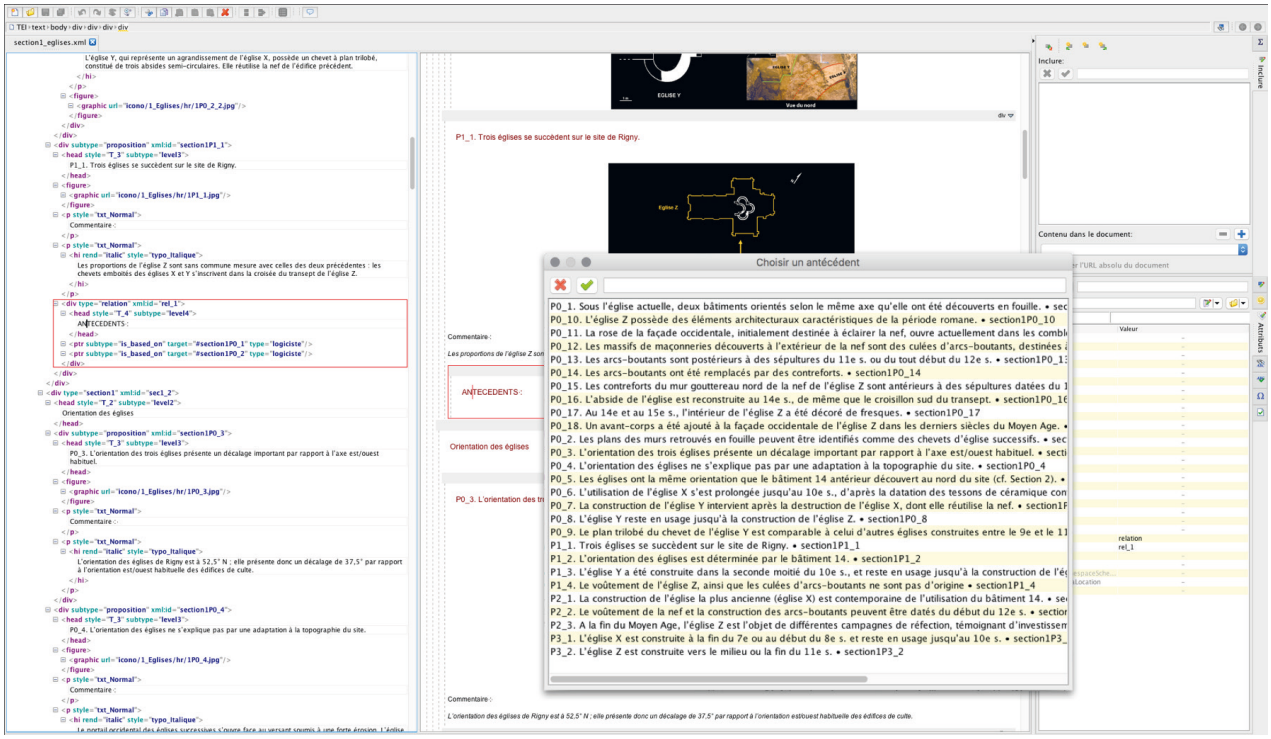

Figure 9. The Graphical User Interface (GUI) for building the relations diagram between propositions.

presentation of the XML markup of the relations between propositions). To add a new relation between proposition *A* and another one, the user only needs to repeat these actions. At the end of the process, the proposition *A* contains its original statement with a group of *ptr* elements pointing to the propositions on which the statement *A* is based upon.

For the time being, the relation qualifications are stored in the *subtype* attribute of *ptr* element and are not dynamics. In the near future we will add a new option to allow the use of custom set of relation types. This collection of pointers does not contain any original text, and must be considered as a new layer of information, which can be used to produce new solutions to retrieve the data and to assess step-by-step the interpretative construct.

In fact, as we will see in the section below, this network of enriched information can be extracted from the XML tree as an RDF graph containing the propositions' titles and the interrelationships between them.

Figure 9 gives an overview of the Graphical User Interface (GUI) developed within XMLmind XML Editor that allows the scholar to build the interrelationship diagram. The *pick window*, with the propositions list, is in the foreground.

From RDF to SVG

In this section we take a close look at all the file formats used in the successive steps of the process ending with the overview of the logicist diagram. At the end of the logicist interrelationship building process, the XML tree is enriched with a collection of pointers leading from interpretative proposition to other propositions, which either belong to the initial set of descriptive statements {P*0*} or to lower level interpretative propositions {P*n*}. This collection gives us all the necessary information to produce an RDF graph (https://www.w3.org/TR/2014/REC-rdf-schema-20140225/) based on the *subject-predicate-object* expression. Considering the following expression: "the interpretative proposition A is based on propositions B, C and D", where "A" is the *subject*, "is based on" the *predicate* and "B", "C", and "D" the *objects* of the expression, we may thus write it as follows in the RDF/XML syntax:

```
<owl:NamedIndividual rdf:about="A">
<rdf:type rdf:resource="logicisme.xml#Pro-
position"/>
<pddn:Titre>Title of proposition A</
pddn:Titre>
```

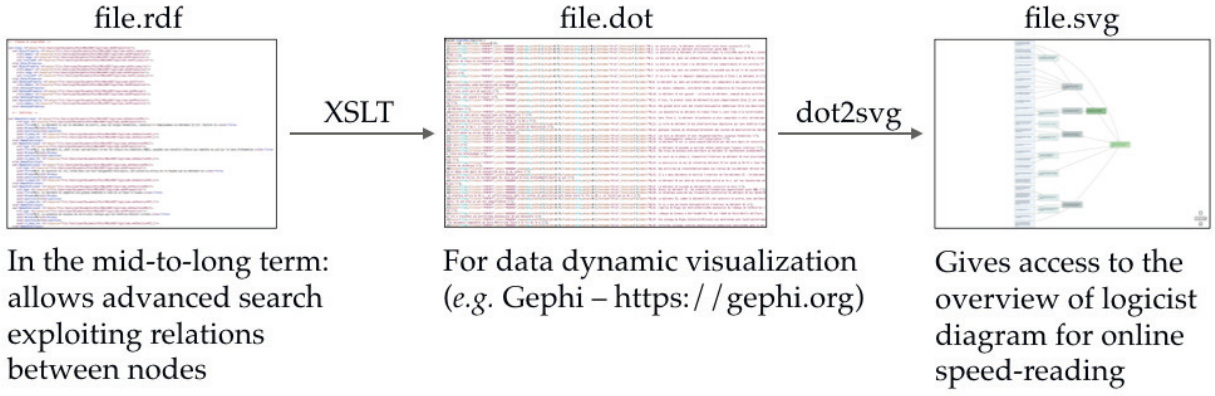

Figure 10. The three file formats used to produce a diagram overview of the interpretative process.

```
<pddn:Niveau>Interpretative level of proposition A [superior to 0]</pddn:Niveau>
<pddn:is_based_on rdf:resource="#B"/>
<pddn:is_based_on rdf:resource="#C"/>
<pddn:is_based_on rdf:resource="#D"/>
</owl:NamedIndividual>
```

(The interpretative *level* of the proposition A must be superior to 0, otherwise it would be considered a descriptive proposition).

The RDF file is particularly important in our workflow because it is the basis of our future advanced search engine. RDF adds the possibility of qualifying the relationship between nodes so that it is possible to express constraints on the request. For example, the user can request a specific string in a specific interpretative proposition level with specific relations to other propositions. The RDF is the most expressive file available at our disposal for the time being.

From the RDF file, we then produce a DOT (http://www.graphviz.org/content/dot-language/) file. DOT is a plain text graph format, which is easily written with automatic tools from another file format such as RDF. The main goal of DOT format is to provide a solution to describe graphs with all the necessary information about nodes, edges and specific organization types like ranking for example.

To write the DOT file, we define a set of rules to organize the graph in accordance with both the logicist programme and the specific case of excavation reports. The graph must be read from left to right, starting with the *descriptive propositions* {P*0*} on the left side, and including a new column of propositions for each additional interpretative level from {P*1*} to {P*n*}. This rule set is encoded in a XSLT file that is applied to the RDF/XML file, and the DOT file is the result of the transformation. DOT files can be parsed to draw graphs by using various programs to produce fixed images like DOT, NEATO (command line) (For a complete list, see: http://www.graphviz.org/Documentation.php) or Gephi (Bastian, Heymann & Jacomy 2009) (https://gephi.org/), which give an interactive visualization interface. With Gephi, users can explore and manipulate their data as a network in a fully dynamic environment. The last step of the process is the production of a diagram overview, using Scalable Vector Graphics (SVG) to display it on the publication website. A basic call of the *dot* command produces a well-formed SVG file, ready to be used online. Whereas all the files produced during the process have to be stored, because each one of them provides specific benefits, we need only to write a simple *bash* script to be able to go from the original XML TEI to the SVG with a single command. The final user will of course only have to click on a button in the GUI to invoke this *bash* command and launch the process. Figure 10 gives a short presentation of the three file formats (.rdf, .dot, and .svg) outlining their respective benefits. The RDF file is at the top of our workflow as it contains all the data, the nodes, and explicit relations between the nodes. It is the file that will allow, in the mid-to-long term, the building of an advanced search engine dealing with all the reasoning process and enabling users to formulate complex queries on data sets. The DOT file, which is a descriptive file containing all the necessary information to draw graphs, allows the use of dynamic

visualization dedicated softwares like Gephi. With this tool, users may build specific views to explore their data sets. In our workflow, the SVG file is the final shape given to the reasoning process. It is the file which is integrated to the website to give access to the overview of the logicist diagram for online speed-reading.

Conclusion

In this paper we have introduced a new solution inspired by J.C. Gardin's logicist programme, to build, manage, retrieve and assess interpretative constructs. The resulting network of relationships between propositions allows the automatic production of graphical overviews integrated in publication websites in order to provide speed-reading solutions while also allowing for in-depth consultation of basic data and the interpretation process.

We have first implemented this method for the publication of the excavation of the settlement and church in Rigny, and we are currently adapting the rule set to the field of archeology of techniques to meet the requirements of *The Arkeotek Journal*, (http://www.thearkeotekjournal.org/) directed by Valentine Roux, with excellent results. We have developed a complete reusable workflow with a dedicated GUI within the XML editor to add relations between propositions. We are currently adding a dynamic access to the relationship diagram directly within the XML editor, so that the author will be able to access a graphical overview of his inference tree while he is actually building it.

Future work will focus on the integration of the CRMinf (http://new.cidoc-crm.org/crminf/home-4/) module for inference annotation in RDF files. The main goal is to build corpuses of interpretative rules with a shared international standard to describe the relationships between propositions. We also plan to explore the benefits of an internal annotation of propositions, regarding both the concepts and their interrelations, by using the Glozz platform (Widlöcher and Mathet 2012, http://www.glozz.org/) in order to build an advanced search engine.

References

Bastian, M, Heymann S, and Jacomy, M 2009. Gephi: an open source software for exploring and manipulating networks. In: *International AAAI Conference on Weblogs and Social Media, 17-20 May 2009*. Menlo Park: AAAI Press. pp. 361-362.

Gardin, J-C 1979. *Une Archéologie théorique*. Paris: Hachette (English translation: Gardin, J-C 1980 *Archaeological constructs. An aspect of theoretical archaeology.* Cambridge: Cambridge University Press, 1980).

Gardin, J-C 2003. Archaeological discourse, conceptual modeling and digitalization: an interim report of the logicist program. In: Doerr, Martin and Apostolis Sarris (eds), *CAA 2002, The Digital Heritage of Archaeology, Computer applications and quantitative methods in archaeology, April 2002. Heraklion, Crete*. Archive of Monuments and Publications, Hellenic Ministry of Culture.

Gardin, J-C and Roux, V 2004. The Arkeotek project: a European network of knowledge bases in the archaeology of techniques. *Archeologia e Calcolatori*, 15, 25-40.

Roux, V 2009. Modélisation des constructions scientifiques sur multimédia et transfert des connaissances. In: C. Albaladejo, P. Geslin, D. Magda, P. Salembier (eds) *La mise à l'épreuve*. Versailles: Éditions Quæ "Update Sciences & Technologies", pp. 43-53. DOI 10.3917/quae.albal.2009.01.0043

Roux, V and Aussenac-Gilles, N 2013. Knowledge Bases and Query Tools for a Better Cumulativity in the Field of Archaeology: The Arkeotek Project. In: F. Contreras, M. Farjas, F.-J. Melero *Fusion of Cultures. Proceedings of the 38th Annual Conference on Computer Applications and Quantitative Methods in Archaeology, Granada, Spain, April 2010* (BAR International Series 2494). Oxford: Archaeopress. pp. 2-7.

Roux, V and Blasco, P 2004. Faciliter la consultation de textes scientifiques : nouvelles pratiques éditoriales. *Hermes*, 39: 151-159.

Widlöcher, A and Mathet, Y 2012. The glozz platform: a corpus annotation and mining tool. In: Concolato, C (ed.) and Schmitz, P (ed.) *ACM Symposium on Document Engineering (DocEng'12), 4-7 September 2012*, New York: ACM. pp. 171–180.

Zadora-Rio, E and Galinié, H (forthcoming). *L'église de Rigny et ses abords. De la colonia de Saint-Martin de Tours au transfert du centre paroissial (600-1865),* Caen: Presses universitaires de Caen.