ifcXMLNetwork: A Visual Interface for Exploring and Understanding ifcXML Data

Nayantara Duttachoudhury
nduttac@cs.ubc.ca

1 Domain, task and dataset

IFC (Industry Foundation Classes) is a data model developed by the International Alliance for Interoperability and is used for modelling in the building industry. IfcXML is its XML specification [1, 4]. IFC files can downloaded from building models created in Autodesk Revit [2]. Autodesk Revit is a Building information modelling (BIM) [3] software for architects, structural engineers, MEP engineers, designers and contractors. IFC/ifcXML specifications based convertors can be used to convert IFC to ifcXML [4, 5]. Besides IFC, there are other standards that building data can be exported into such as Microsoft Access, gbXML or DWL. But it is seen that ifcXML contains information not supported by other standards [1], so it is preferred. There are many challenges with ifcXML data. Compared to other standards, it is big and has the most complex schema.

We can understand the schema by drawing an analogy to relational databases, even though ifcXML data is unstructured. ifcXML is a domain specific type of XML data. Like tables in relational databases, we have elements in ifcXML. Each element has a certain number of data instances in it, like tables have rows. Each element has attributes, like names of columns in tables which have separate values for every row. One attribute is reserved for the id, which is the global identifier for each data instance. Many data instances also have an idRef (reference identifier). This acts as a pointer to another data instance in a different element. In relational databases, we have foreign keys in tables which point to primary keys of other tables. For ifcXML, instead of pointers between elements, idRef's are pointers between specific data instances (rows in tables). Properties of objects are often not attached directly to a single element but related indirectly through idRefs. Some of these reference paths are very long. Analyzing how different elements and their attributes are linked through IdRefs to get information about objects and their properties is the biggest challenge with ifcXML data, as seen in figure 1.

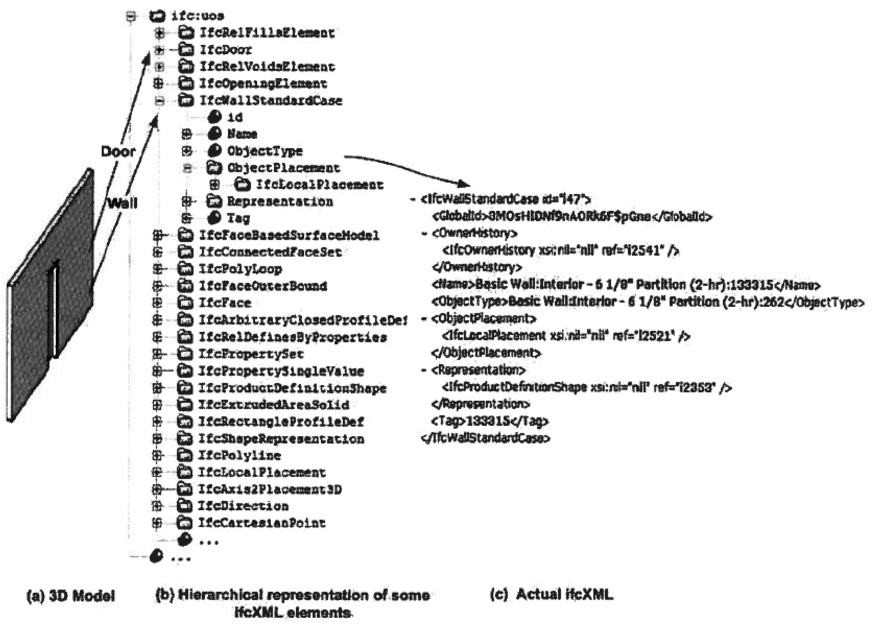
For ease of understanding, here is a recap of the terminology:

- Schema: The structure of the ifcXML data.
- Objects: Real world objects in the building. E.g. air terminal in room 007 in the basement.
- Properties: Information about objects. E.g. capacity of air terminal.
- Elements: Similar to tables in relational databases
- Data instances: Rows of data in tables.
- Attributes: Columns in a table with separate value for each data instance(row).

The purpose of ifcXMLNetwork is to help understand ifcXML data better by visualizing it. The dataset used is the ifcXML file of the mechanical model of the CIRS building in UBC.

2 Personal Expertise

This project relates to my research. When I started working with ifcXML data, I realized that it's biggest problem is that there is no way of understanding how different elements are connected



```
- <IfcWallStandardCase id="i47">
    <GlobalId>3MOsHIDNf9nAORk6F$pGne</GlobalId>
  - <OwnerHistory>
      <IfcOwnerHistory xsi:nil="nil" ref="i2541" />
    </OwnerHistory>
    <Name>Basic Wall:Interior - 6 1/8" Partition (2-hr):133315</Name>
    <ObjectType>Basic Wall:Interior - 6 1/8" Partition (2-hr):262</ObjectType>
  - <ObjectPlacement>
      <IfcLocalPlacement xsi:nil="nil" ref="i2521" />
    </ObjectPlacement>
  - <Representation>
      <IfcProductDefinitionShape xsi:nil="nil" ref="i2353" />
    </Representation>
    <Tag>133315</Tag>
  </IfcWallStandardCase>
```

Figure 1: Structure of ifcXML data.

to each other. I tried XML viewers, but they were not enough. That's when I wished I had a visualization technique to better understand these files.
I do not have much expertise in visualization, but I have used softwares like Tableau to understand the data. But these softwares require the input data to be in a particular format.

3 Proposed Infovis Solution

My proposed infovis solution will consist of 3 different views. The first view will show the overview network according to the ifcXML file. Each node in the network will be an element from the ifcXML file and edges connecting nodes will be weighted according to the number of data instances connected between the elements through idRef's. The size of the nodes will change according to the number of data instances in that element. And edge weights will be shown with varying saturation. The second view will consist of a search bar and filters. The third view, called the data view will show information about a selected node.
Selecting a node (element) in the network will show all the corresponding data instances in the data view. From here, a user can select a particular data instance and add it to the filter view. A user can select more data instances across different elements from the network and add them to the filter view. The user can also directly search for a data instance if the global identifier is known. Once all data instances have been selected, the user clicks on 'redraw graph' in the filter view. This generates a new network in the network view showing the data instances selected in the filter view and how they are connected to other elements in the graph. The overview network can be brought up anytime by clicking on 'Main Graph'.

4 Scenario of use

Rob, a civil engineer has the IFC (or ifcXML) file of the CIRS building in UBC. He wants to see if all the required properties of a particular air terminal have been defined in the model. He tries to convert the IFC file to a .rvt file to view in Revit [2]. But this transformation leads to loss of information. Next he uses an IFC viewer like Solibri [6]. But Solibri has naming conversions of it's own. This leads to inconsistencies in the data. Also, Solibri cannot capture all the information from the IFC file. Now, the only solution for Rob is to manually extract information from the ifcXML data. But the ifcXML is very huge with information spread across the file.

Instead Rob can use ifcXMLNetwork to view the ifcXML file in form of a network. Rob can easily find the air terminal using the global id by searching for it through the search bar. He can add this data instance to the filter view. Since, he is interested in all the information relating to this air terminal, he doesn't need to narrow his search. He clicks on 'Redraw graph' which shows him a smaller network of only elements whose data instances are linked to the data instance of the air terminal as shown in figure 3. Here, he can check if all the required properties have been defined or not.

5 Implementation

The data from the ifcXML files will be reduced using MATLAB and transformed to simple csv files. The interface will be implemented using R and SHINY. SHINY provides a great platform to create multiple views easily.

6 Interface Screenshots

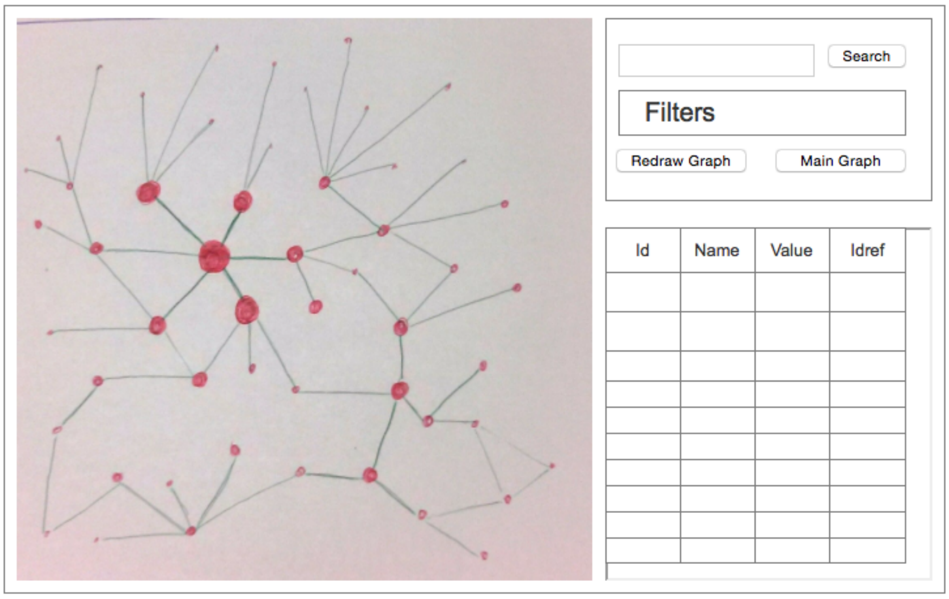


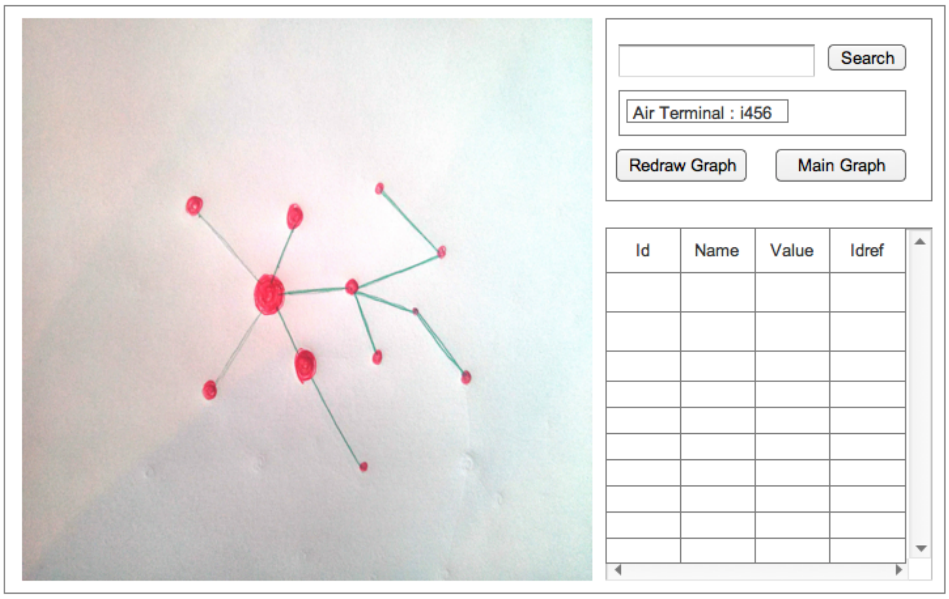
Figure 2: Screenshot of the interface when it shows the overview network.

7 Milestones and Schedule

- Nov 7th 2014: Complete data cleaning and transformation
- Nov 14th 2014: Learn to use R and SHINY and create basic interfaces. **Status update due**
- Nov 21st 2014 : Complete implementation of network in R and SHINY.
- Dec 5th 2014 : Design interface and incorporate all functions of the visualization.
- Dec 12th 2014 : Complete writing project report. **Final presentation due**
- Dec 15th 2014 : **Final report due**

8 Previous work

XML visualization is not a new concept. There are many online XML viewers that extract the hierarchical form of XML data and visualize it as a tree. But each XML file is different from another. And XML files across different domains are very different. Domain specific visualization of XML data



Id	Name	Value	Idref

Figure 3: Screenshot of the interface when it shows the filtered network.

has been done. Dipper et al. [7] have applied visualization to linguistic XML data. CGView [**?**] talks about genome visualization from XML data. Softwares architectures based on XML data have also been visualized [9]. These visualizations are distinct from each other, even though they all have XML data in common. This shows that XML data visualization is very domain specific. Visualization of ifcXML data is a field that remains unexplored. State of the art IFC viewers like Solibri [6] extract information from IFC files and display a 3D model of the building. Basic information about objects in a building can be found by exploring the IFC data through Solibri. But it fails to answer many questions whose answers are deeply embedded in the data. Also since Solibri has a nomenclature of it's own, there are data inconsistencies. In such cases, the only option is to explore the ifcXML data.

References

[1] JIEMIN ZHANG. Evaluations on XML Standards for Actual Applications. 2013

[2] Autodesk Revit. http://www.autodesk.com/products/revit-family/overview

[3] Building Information Model http://www.autodesk.com/solutions/building-information-modeling/overview

[4] Industry Foundation Classes http://www.buildingsmart.org/standards/ifc

[5] ifcXML http://www.buildingsmart-tech.org/specifications/ifcxml-releases

[6] Solibri http://www.solibri.com/products/solibri-model-viewer/

[7] Dipper, Stefanie, and Michael Götze. "Accessing heterogeneous linguistic data?generic XML-based representation and flexible visualization." Proceedings of the 2nd Language and Technology Conference 2005. 2005.

[8] Stothard, Paul, and David S. Wishart. "Circular genome visualization and exploration using CGView." Bioinformatics 21.4 (2005): 537-539.

[9] Houlding, David Ian. "Method and system for providing visualization of underlying architecture of a software system." U.S. Patent No. 7,415,697. 19 Aug. 2008.