# Importing Spreadsheet data into Protégé : Spreadsheet plug-in

**Jay (Subbarao) Kola MBBS. (PhD), Alan Rector** MD, PhD.
*School of Computer Science, University of Manchester, Manchester M13 9PL, UK*

## Abstract

Tabular data offers a popular and alternative format of capturing knowledge. More recently, spreadsheets are an increasingly popular format of storing tabular data. This is especially the case for domain experts who do not want to learn the intricacies of databases, frame based models or even OWL (Web Ontology Language)[1] models. In this paper, we discuss some of the limitations of using tabular models in representing knowledge and a plug-in for Protégé that allows knowledge authors to migrate knowledge captured in spreadsheets into Protégé.

## Introduction

As relational databases grow in popularity, much of the knowledge in knowledge-based applications tends to be represented as relational tables. More recently, knowledge also tends to be captured in spreadsheets as well as databases. However, these tabular models have their demerits. OWL with its formal logic under pinning offers the following advantages, which are only briefly discussed in the paper.

- Easy, intuitive way to capture knowledge using 'sub-sumption'[s] and 'defined classes'[x].
- Ability to maintain and extend a knowledge corpus and ability to track effects of changing a section of the knowledge base.
- Ability to infer knowledge with the help of a first order logic services built into a variant of OWL (OWL-DL).

To this end, we built the "Spreadsheet plug-in" for importing knowledge held in spreadsheets into Protégé [2]. This plug-in is designed to work in Protégé-4[a], which is the latest version of the Protégé Knowledge Editor, currently being developed at Manchester University [3].

We consider a small example in Occupational Health, which relates diseases to their causative agents and the industries involved. For example, we know *Pigmentation* is caused by *Arsenic* and *Black Pigmentation* is caused by *Coal Tar*. *Coal Tar* actually consists of; *Asphalt* and *Pitch*. If we wanted to relate this in tables, we would need to create a table like Table 1.

| Clinical Finding | Cause |
|---|---|
| *Pigmentation* | *Arsenic* |
| *Black Pigmentation* | *Coal Tar Constituents* |
| *Black Pigmentation* | *Asphalt* |
| *Black Pigmentation* | *Pitch* |

*Table 1: Disease-Cause relationship table*

*Table 2: Cause-Industry relationship table* ⟶

| Cause | Industry |
|---|---|
| *Arsenic* | *Glass product manufacturing* |
| *Arsenic* | *Electronic product manufacturing* |
| *Coal Tar Constituents* | *Construction Industry* |
| *Asphalt* | *Construction Industry* |
| *Pitch* | *Construction Industry* |

If we say *Glass product manufacturing* and *Electronic product manufacturing* have exposure to *Arsenic* and *Construction Industry* has exposure to *Coal Tar Constituents*; we need to relate all these entities in another table as shown in Table 2.

We can already see that there is needless repetition of information in the tables. In reality we might want to say that *Pigmentation* can also be of other types: *Blue Pigmentation* and *Yellow Pigmentation*. We then say *Blue pigmentation* has specific cause *Silver Salts* and *Yellow Pigmentation* has specific cause *Dinitrophenol* in addition to the general causes of *Pigmentation*. This clearly means extending the existing table to include these new cases.

---

[s] Also referred to as 'parent-child' or 'is-type-of' relationships.
[x] A means of representing complex concepts using simple concepts and relationships
[a] Latest version of Protégé 4 and related plugins can also be downloaded at http://www.co-ode.org/downloads/

We might also want to say *Construction Industry* can be of different types; *Roof Construction* and *Road Construction*. This means rewriting all the tables that we have created to include the additional industries and the process is laborious and cumbersome.

Using a hierarchy relationship table (Table 3) might address some of these issues but still does not avoid all the problems with this representation. However creating restrictions between the appropriate parent classes in OWL-DL addresses this issue. For example, creating a restriction '*Pigmentation*' *has_cause* '*Arsenic*' ensures that Black Pigmentation inherits this as shown in Fig 4.

| Parent Concept | Child concept |
|---|---|
| *Pigmentation* | *Black Pigmentation* |
| *Pigmentation* | *Blue Pigmentation* |
| *Pigmentation* | *Yellow Pigmentation* |
| *Coal Tar Constituent* | *Pitch* |
| *Coal Tar Constituent* | *Asphalt* |

*Table 3: Parent-Child relationship table*

**Methodology**

The rough transformation of the same tabular information into an OWL-DL ontology using hierarchies and restrictions will look like :

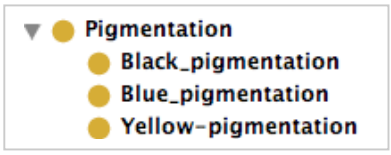


*Figure 1: Disease hierarchy*

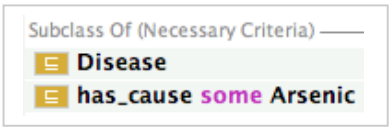


*Figure 2: Restriction on Pigmentation*


*Figure 3: Causative agent hierarchy*

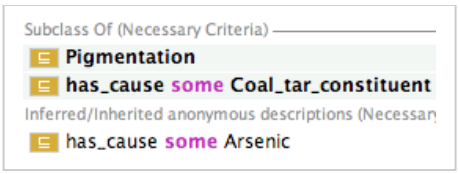


*Figure 4: Restrictions on Black Pigmentation*

In creating this transform into OWL-DL, we create class hierarchies and restrictions corresponding to the knowledge in the tables above using the Spreadsheet plug-in. We now explore how the Spreadsheet plug-in in Protégé helps us create this transform in a few easy steps. On loading the corresponding spreadsheet file into the plug-in, we are presented with an interface that looks like:

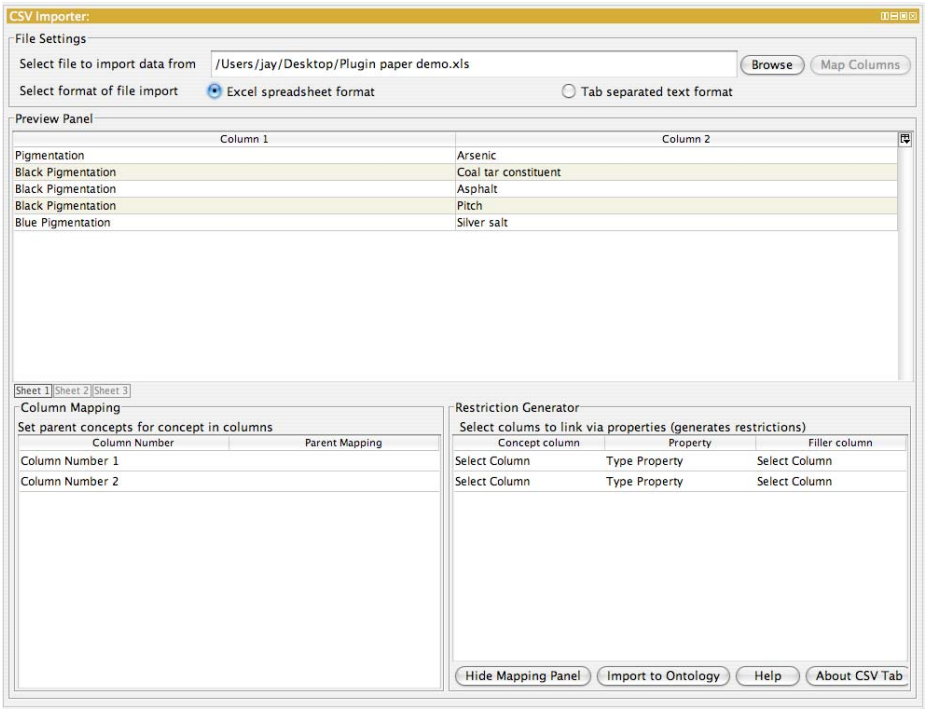


*Figure 5 : Spreadsheet plug-in interface*

**Importing concepts into an ontology :**
To import concepts in the spreadsheet into the ontology, we use the "Column Mapping" feature of the plug-in where we specify a parent concept for all concepts in a column. For example to import the concepts in our spreadsheet, we assign a parent concept "Disease" to all concepts in Column 1; Pigmentation, Black Pigmentation and Blue Pigmentation. Likewise we assign "Causative agent" as parent concept for Column 2 concepts as shown in Figure 6. We could also drag and drop existing concepts in the ontology against the column name to add concepts under this concept. We could like-wise you OWL-Thing to add concepts under OWL-Thing. This creates the concept hierarchies for the corresponding concepts.

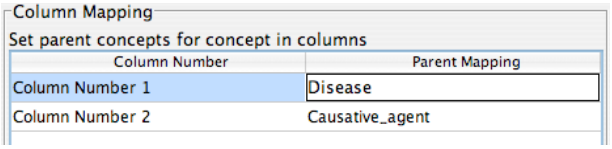


*Figure 6: Assigning parent to column concepts*

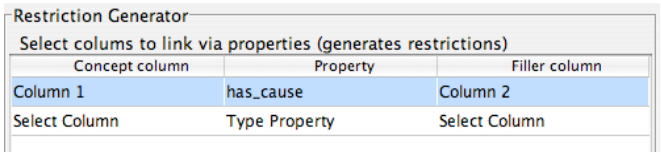


*Figure 7: Creating restrictions between column concepts*

**Creating restrictions :**
To create restrictions between concepts in columns, we use the "Restriction Generator" feature of the Spreadsheet plug-in. To generate a restriction, we specify which column concepts to add the restriction to and a corresponding column concept to add as filler value. We finally specify a name for the restriction as shown in Figure 7.

**Creating hierarchies:**
To create hierarchies of concepts, we open the corresponding worksheet in the imported Excel spreadsheet which is shown in Figure 8 .

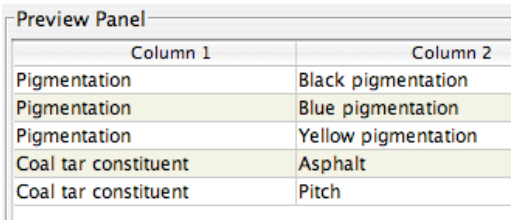


*Figure 8 : Concept hierarchy table*

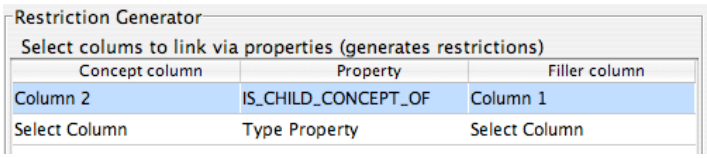


*Figure 9 : Concept hierarchy creation in Plug-in*

We then use the "Restriction Generator" panel to create a parent-child relationship between the concepts. In order to generate this, we specify a predefined relationship name in the panel. This "IS_CHILD_CONCEPT_OF" relationship generates a hierarchy mapping the concepts in the column on the left hand side of the relationship as children of concepts in the column to the right hand side. This creates the hierarchies shown in Figures 1 and 3.

Using these features of the Spreadsheet plug-in transforms the content into an OWL-DL ontology but to create an accurate representation, we need to edit some of the redundancies that are inherited from the spreadsheet. For example, we need to remove the restriction *'Black Pigmentation' has_cause 'Arsenic'* created by the plug-in since this is inherited from *'Pigmentation' has_cause* 'Arensic' since '*Black Pigmentation*' is a child of '*Pigmentation*'. Such redundancies are inherent to tabular models. These are best avoided by adopting OWL as the modelling environment and good modelling strategies.

## Conclusion

We present a short argument against modelling knowledge in tabular form and also provide an easy to use tool (Spreadsheet plug-in) which enables knowledge modellers to import their existing knowledge bases in spreadsheet format into Protégé-OWL.

## References

1. W3C Consortium OWL Specification (http://www.w3.org/2004/OWL/)
2. Protégé Ontology Editor (http://protege.stanford.edu/)
3. Protégé 4.x : http://protege.stanford.edu/download/prerelease-alpha/prototype.html