Online Information Visualization: Automatic Mapping of Data Entities to Representation Artefacts

Owen Gilson[1], Nuno Silva[2], Phil W. Grant[1], Min Chen[1], and João Rocha[2]

[1] Department of Computer Science, University of Wales Swansea, UK
{csowen, P.W.Grant, M.Chen}@swan.ac.uk
http://cs.swan.ac.uk/∼csowen/OnlineInfoViz
[2] ISEP Instituto Superior de Engenharia, Instituto Politécnico do Porto, Portugal
{Nuno.Silva, Joao.Rocha}@dei.isep.ipp.pt

Abstract. In order to allow Online Information Visualization of Semantic Web data we must have a method of automatically mapping entities in the source data to representation artefacts in the target format.
Online Information Visualization aims to increase the accessibility of otherwise machine-readable only data by creating a "good enough" visualization quickly and simply. The system requires no user interaction and has no prior knowledge of the information domain. We present a technique which assess the characteristics of each entity in the source data and maps them to the representation artefacts deemed appropriate in the target format. Our technique is independent of the source data format and target visualization format. We provide a worked example and user evaluation of the SVG visualization of a real-life data format (BBC top 40 music chart).

1 Introduction

The Semantic Web [Berners-Lee and Fischetti, 1999] is providing an increasing amount of data in machine readable formats such as XML and RDF, allowing machines to reason about data and thus providing additional services. However, in this process we must not forget the needs of human users. Humans still need to be able to view this data as well as machines able to process it. Human users should be able to access data at every stage that a machine processes information on the Semantic Web. This provides users with an additional level of confidence about the validity of the machine processing task. This is especially important as a complement to the upper levels of the Semantic Web stack (i.e. trust and proof)[ref].

Information Visualization is concerned with generating visual representations of data to allow new insights to be gained into what is being conveyed. [Card et al] state that: "Information visualization is the use of computer-supported, interactive, visual representations of abstract data to *amplify cognition*". With

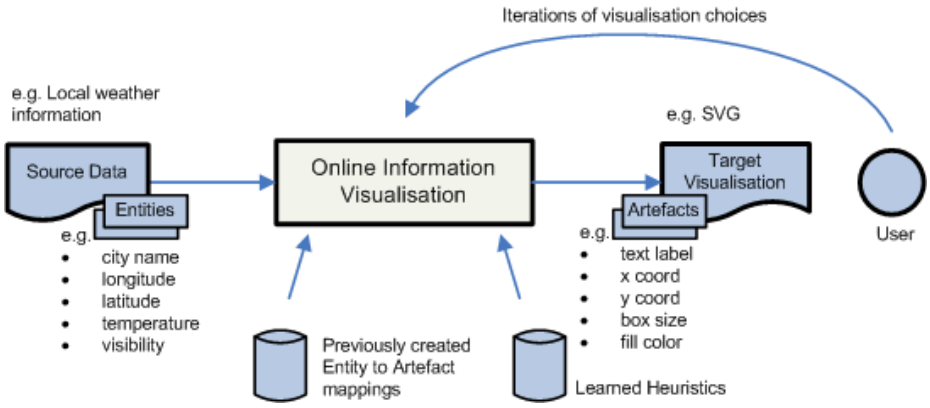

Fig. 1. Overview of Online Information Visualization

respect to the human needs of the Semantic Web, we believe that Information Visualization can be used to *increase accessibility*. We call this technique *Online Information Visualization*.

Traditionally, Information Visualization is an off-line process where visualizations are gradually developed over time to meet the requirements of specific information domains. This aspect makes it unsuitable for Online Information Visualization.

For Online Information Visualization to be successful, it must be:

1. **Fast**. Users provide the system with a data source and a visualization will be returned with no further information required.
2. **Domain independent**. The system should be able to provide a visualization without prior knowledge of the information domain or source data format.
3. **"Good enough"**. The initial emphasis of the visualization should be accessibility of the data. Providing deeper cognitive insights is a later objective.
4. **Interactive**. The user should be able to alter and update a visualization choice through an easy to use, interactive user interface.
5. **Iterative**. The visualization can be tweaked and updated to increase cognitive value and be more applicable to the information domain by successively iterating the visualization rules.
6. **Collaborative**. Customization choices should be stored by the system, so that a recognized data format is always rendered using the best rules which have previously been developed.
7. **Able to learn**. The system should learn from general heuristic about what makes a good visualization. eg. time based values should be on the x axis.

An overview of Online Information Visualization is shown in figure 1.

In Section 2 we provide an overview of related work. In Section 3 we give a full description of the Online Information Visualization process. We then describe

in Section 4 the mapping functions used in the process and provide a worked example in Section 5. In Section 6 we provide some further examples. In Section 7 we describe a proof of concept and provide results from a user evaluation which we conducted. In Section 8 we give concluding remarks and our thoughts on future work.

2 Related Work

2.1 Ontology Mapping

There has been a considerable amount of work conducted in trying to produce automatic translation and mapping systems. Much recent work, particularly in the semantic web community, has been concerned with ontology mapping and alignment. The goal of this research has been to define and systematize a general approach to ontology mapping [ref: MAFRA, etc]. This has produced general models of operation with defined dimensions:

Core mapping processes: Lift and Normalization; Similarity Measurement; Semantic Bridging; and Execution and Post-processing.

Complimentary operations: Evolution; Cooperative Consensus Building; Domain Constraints and Background Knowledge; and Graphical User Interface

Different parts of the research community have focussed on different areas to increase the quality and efficiency. For example, semantic bridging is concerned with creating mappings between semantically similar entities by measuring the similarity of the concepts which they represent.

"The automation dimension concerns the ability of the system to propose semantic bridges between ontologies entities according to the previous inputs. (...) No complete automation of the process is possible and therefore human intervention upon the proposals is envisaged" [N.Silva thesis: 4.3.3.1 Automation(of Semantic Bridging) (pg.56)]

Our approach aims to try to produce a totally automated mapping (at least initially). This gives the user a concrete visualization to work with where the initial quality is not as important as having the ability to quickly and easily produce an output. This contrasts with conventional mapping techniques which are often used for data integration and database mapping tasks. Here the primary emphasis needs to be on quality and accuracy of the mapping. The demands of Online Information Visualization are far less restrictive in this respect. This gives our research a wide scope in which to explore.

[Rahm and Bernstein, 2001] authors suggest a taxonomy for the classification of schema matching approaches, i.e., the approaches applied in deriving matches between schema entities. We focus on instance and contents based techniques which examine frequency and types of element-level data.

2.2 The Formal and Informal Semantic Web

The methods for representing semantically rich data on the semantic web vary in their formality. The most is OWL [ref] which is an ontological representation of knowledge based on Description Logic [ref]. The least formal are Microformats [ref] which are light-weight formats with markup that allows the expression of loosely coupled semantics in an XHTML web page.

Whilst our approach covers all XML based data formats, we believe that there is a great opportunity for Online Information Visualization to become the accessible, "friendly-face" of the semantic web. In particular, it complements the emerging popularity of informally defined semantically rich data such as microformats and OPML [ref].

2.3 Structure vs. Semantics

Much of the work on visualizing the semantic web has been concerned with RDF data. This data is in the form of a labeled-directed graph structure with visualizations being concerned with examining the structure of this data [Geroimenko and Chen 2006]. The user is then able to analyze the data to find clusters, patterns and trends. While this can provide useful information about the data, we believe that there is also a benefit from visualizations which consider additional semantics of the data. For example, in a dataset concerning records of people, the age of a person should ideally be visualized differently from the number of children the person has. In this way, the visualization becomes richer and more meaningful, also allowing other representation formats (textual and audial) to be considered.

We therefore view Online Information Visualization as the representation of data which has a high level of semantics, where we attempt to model the data according to these semantics.

3 The Online Information Visualization Process

In this section, we give an outline of the process and describe how it aims to address the 7 characteristics described in Section 1. Each stage of the Online Information Visualization Process is shown in figure 2. To illustrate the process a dataset of sports fans will be used. An extract of the source file is shown below.

```
<fans>
    <person name="alice" age="28" tallness="1.41" nationality="welsh"
    scarves="2" games="19" shirt-size="M" />
    <person name="bob" age="37" tallness="1.02" nationality="scottish"
    scarves="4" games="33" shirt-size="S" />
    <person name="colin" age="16" tallness="1.84" nationality="irish"
    scarves="6" games="8" shirt-size="XL" />
    ...
</fans>
```

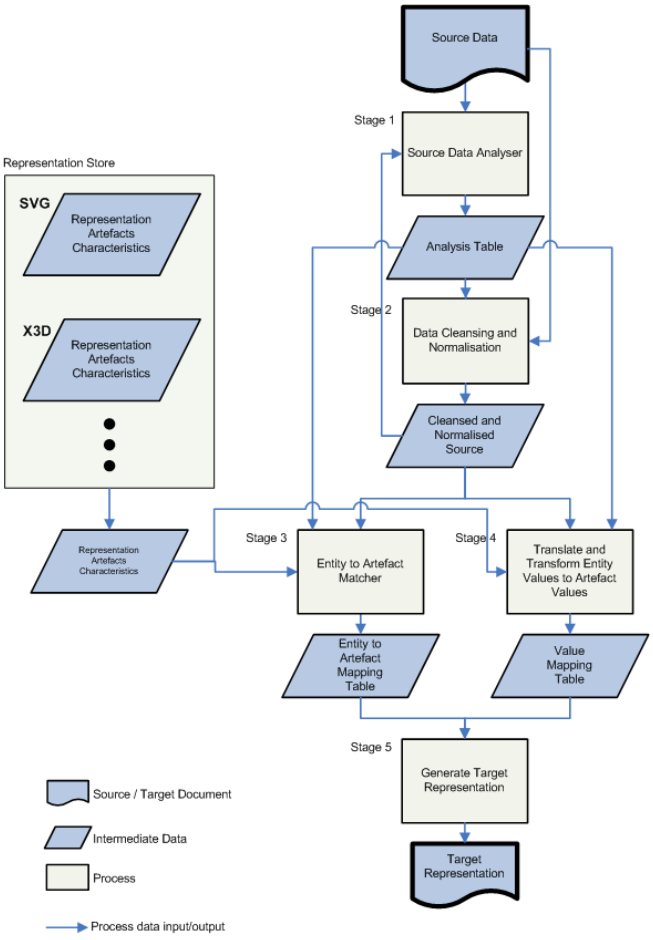

Fig. 2. The Online Information Visualization process

3.1 Source Data Analyzer

During this stage we analyze the source data in order to create an analysis table. We must analyze the nature of the source data to attempt to extract semantics. There are a number of possible ways to do this, such as using a schema processor [Buswell, 2003] or using an existing research tool which address these issues [YALE, XXXX]. We choose an alternative approach which we think is well suited to semantically XML rich data sources.

We consider XML elements and attributes as a generalized concept called an Entity. This is similar to the "Layered Normal Form" which an XML Normal Form [Thompson, 2001]. An Entity has a value, a name, and an XPath. It also has a parent entity, and in some cases it has child entities. In this way, we can concentrate on the values and structure of the data rather than its XML specific

representation method. [Possibly include ontology diagram from I-Know paper here?]

For each entity, we calculate the following:

Value Type: Integer, real, string or URL.

Unique values: This is the number of unique values when considering all entities of the same name. It is provided as an absolute value and as a proportion of all records.

Structure Semantic: This only applies to elements which have child elements. In this case, the entity is representing a structural semantic. This is either a root of the document (which must only occur once), or a container of other entities.

Value Category: A value category is one of 3 values:

1. **orderedNumeric** Numeric values which by their very nature are ordered. For example, *age* (0 to 120) expressed as an integer.
2. **orderedDiscrete** Non-numeric values which have discrete values with an implied ordering. For example, *t-shirt-size* (S, M, L, XL, XXL).
3. **unorderedDiscrete** Non-numeric values which have discrete values but with no implied ordering. For example, *name-of-person* (Bob, Mary, John).

The source data entity analysis for the Fans dataset is shown in table 1.

XPath	Value type	Value category / Structure semantic	Uniqueness	Min	Max
fans	-	Container (root)	-	-	-
fans/person	-	Container (object)	-	-	-
fans/person/@name	String	UnOrdered Discrete	26 values (1.0)	-	-
fans/person/@age	Integer	Ordered Numeric	21 values (0.81)	16	61
fans/person/@tallness	Integer	Ordered Numeric	23 values (0.88)	1.02	1.97
fans/person/@nationality	String	UnOrdered Discrete	4 values (0.15)	-	-
fans/person/@scarves	Integer	Ordered Numeric	10 values (0.38)	1	10
fans/person/@games	Integer	Ordered Numeric	23 values (0.88)	2	42
fans/person/@shirt-size	String	UnOrdered Discrete	5 values (0.19)	-	-

Table 1. Source data Entity analysis for the sports Fans dataset

3.2 Data Cleansing and Normalization

During this stage, we cleanse the data which in turn helps with the semantic analysis which occurs during the previous stage. The Source Data Analyzer is often able to inform this stage where Data Cleansing and Normalization maybe required. In fact, after this stage is complete, we feed this data back to the Source Data Analyzer. This is so that the newly cleansed and normalized data can be re-analyzed.

Data cleansing is not needed for this example dataset. However, we must provide the system with additional semantics for the `shirt-size` entity. The system has determined that this is an entity with category *unOrderedNumeric*. However, in actual fact the values have an applied ordered. Therefore, in order to visualize this entity as accurately as possible a human user must change the category to be *orderedNumeric*.

3.3 Entity to Artefact Matcher

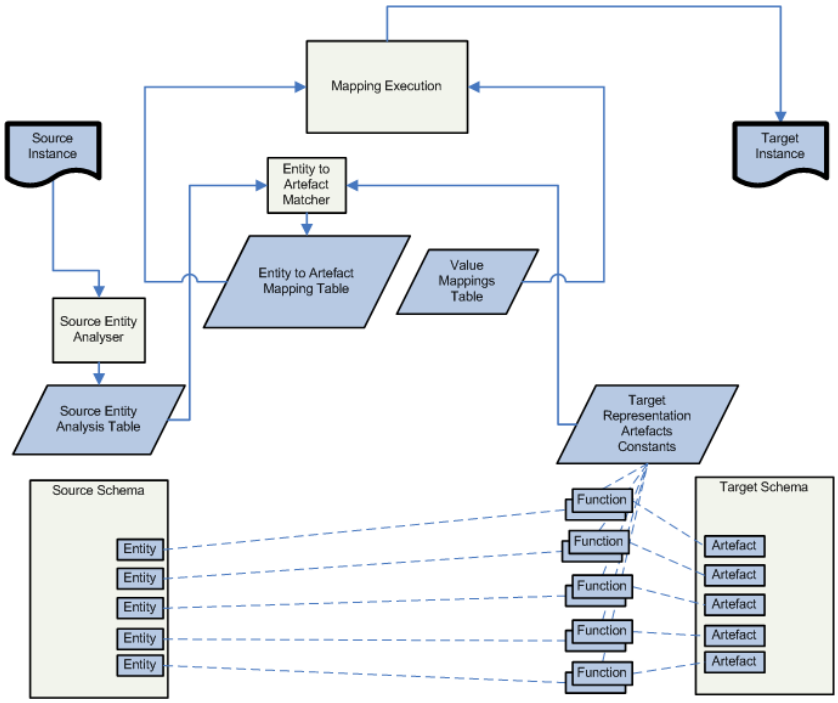

Fig. 3. The mapping process (data centric)

Representation Artefact Capabilities For every representation format we have a table which shows the capabilities of each representation artefact to handle each category. See table 2 for the table created for SVG.

For each representation artefact, we determine which categories it will accept. Most representation artefacts will accept only one category. For example the `x` coordinate artefact in SVG only accepts values of category *orderedNumeric*. However, certain artefacts such SVG `fill` color and `text` label can accept multiple categories.

Artefact Name	Artefact Category	Entity Catgeory		
		ON	OD	UOD
x	ON	***	**	*
y	ON	***	**	*
width	ON	***	**	*
height	ON	***	**	*
fill	ON *	***	**	*
fill	OD ***	**	***	*
fill	UOD **	*	**	***
title	UOD	*	**	***

Table 2. SVG artefact categorization and mapping function prioritization (*** is most favored, * is least favored)

Representation Artefact Priorities At this point, all core functions whose output type matches the value category of the representation artefact are available to the artefact. For example the x coordinate artefact in SVG has associated functions: ON to x, OD to x and UOD to x. The system chooses which function to used by consulting the table. Combinations marked *** are the most favored option, while combinations marked * are least favored option.

Representation Artefact Functions For each representation artefact we have a function. The class diagram in Figure 4 shows the origin and categorization of each function. As can be seen, for each representation artefact in SVG has one or more function associated with it.

In Section 4 we describe the core functions which are available to map Entities to Representation Artefacts in any target representation format. In many cases, a Representation Artefact can have more than one function. For example, *fill-color* can have a orderedNumeric, orderedDiscrete, or unorderedDiscrete value. As such, all functions below are available.

There are two versions of the unorderedDiscrete to unorderedDiscrete function. One is "matched" which means that input values are matched to output values. This is used when there is a semantic link between entity value and artefact value. For example *nationality* and *fill-color*. The other function is "unmatched" which is used when there is no requirement for a semantic link between entity value and artefact value. For example, *name-of-person* and *fill-color*.

The calculation of which Data Entities to map to which Representation Artefacts is a combinational problem[ref]. The system must consider which combination produces the best result based on the prioritization table of the target representation format. In future versions of the system past mappings and heuristical rules will also be included in this process.

3.4 Mapping Entity Values to Artefact Values

In many cases the values of the source data can not be used directly in the Target Representation. Instead a value mapping or translation process must

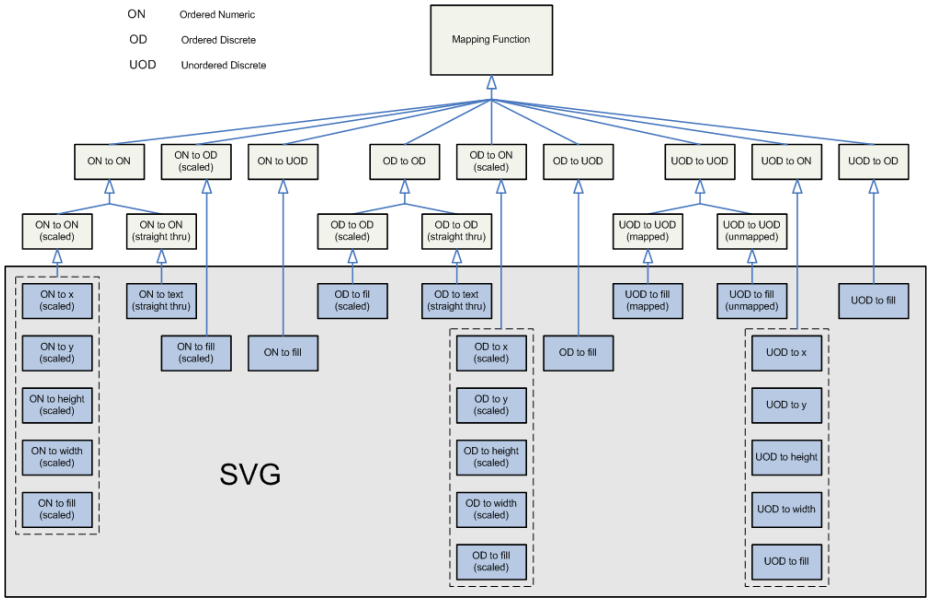

Fig. 4. Mapping Functions Class Diagram

occur. For example, the `nationality` entity is mapped to the `fill color` visual artefact, therefore we must create a mapping between the nationality names and appropriate colors.

3.5 Generate Target Representation

The input to this stage is the Mapping Table from the previous stage and the Source Data. We generate the Target Representation by creating Representation Artefacts (in the target representation format) with the values supplied by the Source Data and the mapping table produced in the previous stage. The output of this stage is the final Target Representation which is shown in figure 5.

At every stage of the Online Information Visualization Process, the system will have made assumptions about the nature of the data based on probabilistic decision making. As such, there are likely to be some choices which are not what the user would have chosen given the choice. To address this, the user has the ability to alter settings and change any of the assumptions and decisions made by the system, therefore improving the overall suitability of the target representation. In this way, the system learns from the user's intervention, thus creating a more accurate process for future visualizations.

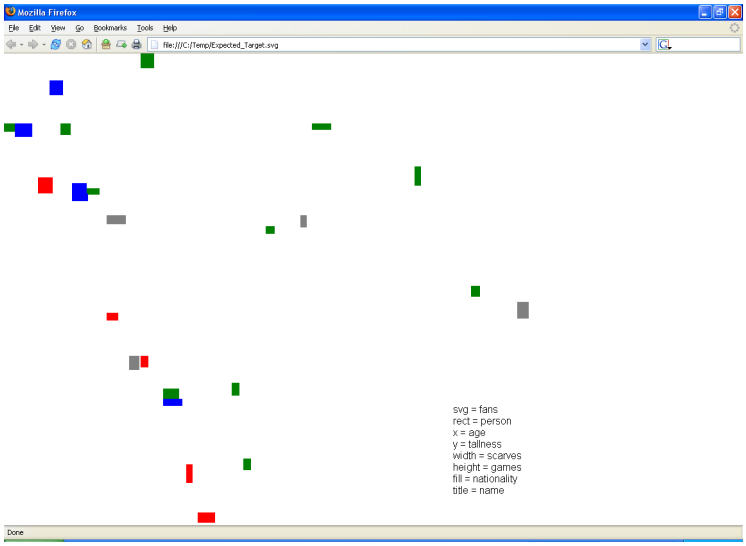

Fig. 5. SVG visualization of the Sports Fans dataset

4 Core Mapping Functions

The core mapping functions described in this section are all partial order preserving mapping. All values are scaled linearly. When we describe numeric values, we do not make distinction between integer and real numbers. Therefore,

4.1 orderedNumeric to orderedNumeric

This function takes input values within the domain defined and outputs them as scaled values in the range defined. Example: `age` (0 to 120) to `x-coordinate` (0 to 800). In this case, `age` "0" returns `x-coordinate` "0"; `age` "30" returns `x-coordinate` "200"; and `age` "120" returns `x-coordinate` "800".

Domain minInputValue $\leq$ inputValue $\leq$ maxInputValue
Range minOuputValue $\leq$ outputValue $\leq$ maxOutputValue

Parameters:

1. inputEntity
2. minInputValue
3. maxInputValue
4. minOutputValue
5. maxOutputValue

4.2 orderedDiscrete to orderedNumeric

This function takes input values within the domain defined and outputs them as scaled values in the range defined. Example: `t-shirt-size` (S, M, L, XL, XXL) to `rectangle-height` (10 to 50). In this case, `t-shirt-size` "S" returns `rectangle-height` "10"; `t-shirt-size` "XL" returns `rectangle-height` "40"; and `t-shirt-size` "XXL" returns `rectangle-height` "50".

Domain any value from discreteInputValue$_1$ to discreteInputValue$_n$
Range inOuputValue $\leq$ outputValue $\leq$ maxOutputValue

Parameters:

1. inputEntity
2. discreteInputValue$_1$, discreteInputValue$_2$, ..., discreteInputValue$_n$
3. minOutputValue
4. maxOutputValue

4.3 orderedNumeric to orderedDiscrete

This function takes input values within the domain defined and outputs them as scaled values in the range defined. Example: `temperature` (0 to 100) to `text` (navy blue, light blue, pink, red). In this case, `temperature` "21" returns `text` "navy blue"; `temperature` "48" returns `text` "light blue"; and `temperature` "76" returns `text` "red".

Note: The discrete values which are output by the function are chosen according to minInputValue and maxInputValue. Therefore, the output will only contain instances of every discrete output value if the following conditions are both true:

1. The number of input values is greater than or equal to N (the number of discrete output values).
2. The input values are sufficiently distributed throughout the input values range.

If the user wishes to ensure that the output will contain instances of every discrete output value, no matter what the distribution of input values is, then the function "orderedDiscrete to orderedDiscrete" should be used instead.

Domain minInputValue $\leq$ inputValue $\leq$ maxInputValue
Range any value from discreteOutputValue$_1$ to discreteOutputValue$_n$

Parameters:

1. inputEntity
2. minInputValue
3. maxInputValue
4. discreteOutputValue$_1$, discreteOutputValue$_2$, ..., discreteOutputValue$_n$

4.4 orderedDiscrete to orderedDiscrete

This function takes input values within the domain defined and outputs them as scaled values in the range defined. There can be a different number of input and output values. There can be more input values than output values or vice-versa. Example: `hotel-star-rating` (*, **, ***, ****) to `color` (light-blue, mid-blue, mid-green, dark-green, mid-red, dark-red). In this case, the `hotel-star-rating` "*" returns `color` "light-blue"; `hotel-star-rating` "**" returns `color` "mid-blue"; `hotel-star-rating` "***" returns `color` "mid-red"; and `hotel-star-rating` "****" returns "dark-red".

Domain any value from $\text{discreteInputValue}_1$ to $\text{discreteInputValue}_m$
Range any value from $\text{discreteOutputValue}_1$ to $\text{discreteOutputValue}_n$

Parameters:

1. inputEntity
2. $\text{discreteInputValue}_1$, $\text{discreteInputValue}_2$, ..., $\text{discreteInputValue}_m$
3. $\text{discreteOutputValue}_1$, $\text{discreteOutputValue}_2$, ..., $\text{discreteOutputValue}_n$

4.5 unorderedDiscrete to unorderedDiscrete (Unmatched)

This function takes input values within the domain defined and outputs them as values in the range defined. This function is beneficial when there is a large list of output values available. The number of different input values must be less than or equal to the number of different output values. Example: `name-of-person` (bob, mary, john) to `color` (aliceblue, antiquewhite, aqua, aquamarine, azure, beige, bisque, black, blanchedalmond, blue, blueviolet, brown, burlywood, cadetblue, chartreuse, chocolate, coral, cornflowerblue, cornsilk, crimson). In this case, `name-of-person` "bob" returns `color` "aliceblue"; `name-of-person` "mary" returns `color` "antiquewhite"; `name-of-person` "john" returns `color` "aqua".

Domain any value from $\text{discreteInputValue}_1$ to $\text{discreteInputValue}_m$
Range any value from $\text{discreteOutputValue}_1$ to $\text{discreteOutputValue}_n$
Conditions $m \leq n$

Parameters:

1. inputEntity
2. $\text{discreteInputValue}_1$, $\text{discreteInputValue}_2$, ..., $\text{discreteInputValue}_m$
3. $\text{discreteOutputValue}_1$, $\text{discreteOutputValue}_2$, ..., $\text{discreteOutputValue}_n$

4.6 unorderedDiscrete to unorderedDiscrete (Matched)

This function takes input values within the domain defined and outputs matched values in the range defined. There must be the same number of input values as output values. Example: `nationality` and `color`: ((Welsh, red), (Irish, green), (Scottish, dark-blue), (English, white)).

Domain any value from discreteInputValue$_1$ to discreteInputValue$_n$
Range any value from discreteOutputValue$_1$ to discreteOutputValue$_n$
Conditions m = n

Parameters:

1. inputEntity
2. ((discreteInputValue$_1$, discreteOutputValue$_1$), (discreteInputValue$_2$, discreteOutputValue$_2$), ..., (discreteInputValue$_n$, discreteOutputValue$_n$))

4.7 Straight through functions

Straight through functions are functions which output the exact same data as is input. They are used when dealing with representation artefacts in the target which can be given any value. For example, the `title` attribute in SVG. Straight through functions exist for:

- orderedNumeric to orderedNumeric
- orderedDiscrete to orderedDiscrete
- unorderedDiscrete to unorderedDiscrete

5 Real-life Example - BBC Top 40 Chart Music to SVG

In this example, we take a real-life XML feed [ref] of the BBC Top 40 pop music chart and use the Online Visualization process to produce an SVG representation. The main challenge posed by this dataset over the Sports Fans dataset used in the example in Section 3 is that it demonstrates the Data Cleansing and Normalization stage and how it interacts with the Source Data Analysis stage.

A snippet of the source is shown below:

```
<top-forty>
    <chart position="1">
        <lastweek>1</lastweek>
        <weeks>(5)</weeks>
        <image>http://www.bbc.co.uk/radio1/media/images/artists/
        gnarlsbarkley/060323_cd_crazy_70.jpg</image>
        <artist>Gnarls Barkley</artist>
        <album>Crazy</album>
        <uri>http://www.gnarlsbarkley.com/</uri>
    </chart>
    . . .
<top-forty>
```

The source data consists of 40 chart songs (only 1 shown above) with its current position, last week's position, number of weeks in the top 40, a URL to an image of the artist, the name of the artist, the name of the song, and a URL for more information about the artist.

5.1 Source Data Analysis

In this stage, each element in the source is analyzed. This results in the data shown in table 3.

XPath	Value type	Value category / Structure semantic	Uniqueness	Min	Max
top-forty	-	Container (root)	-	-	-
top-forty/chart	-	Container (object)	-	-	-
top-forty/chart/@position	Integer	Ordered Numeric	40 values (1.0)	1	40
top-forty/chart/lastweek	Integer(0.7), Text(0.3)	Ordered Numeric	29 values (0.73)	1	30
top-forty/chart/weeks	String	UnOrdered Discrete	12 values (0.3)	-	-
top-forty/chart/image	URL	UnOrdered Discrete	8 values (0.2)	-	-
top-forty/chart/artist	String	UnOrdered Discrete	40 values (1.0)	-	-
top-forty/chart/album	String	UnOrdered Discrete	40 values (1.0)	-	-
top-forty/chart/uri	URL	UnOrdered Discrete	40 values (1.0)	-	-

Table 3. Source data Entity analysis for the BBC Top 40 music chart dataset (before data cleansing and normalization)

It can be seen that the system has detected some symptoms of ambiguity with three of the entities. The entities: *lastweek*, *weeks* and *image* have been given probabilistic values for their uniqueness. Also, *lastweek* have multiple probabilistic values for its Value Type.

At this stage, it is possible for the system to alert the user that there are some ambiguities in the analysis of the data. There are 3 possible actions which can be taken:

1. **Continue with the original source data**. The system can simply continue regardless and attempt to visualize the data as is. The benefit of this is that no user interaction is necessary as the system will attempt to provide a "good enough" visualization with the information it already has.
2. **Alter the source data manually**. The user can edit the original source so that the data causing the ambiguity is changed to be more uniform with the rest of the data. This is a good solution if there is a small volume of source data as it is a relatively quick process.
3. **Apply data cleansing and normalization functions**. The user can create simple data cleansing functions in order to change ambiguous data so that it is more uniform with the rest of the data. This is a good solution if there is a large amount of data where it would not be practical to update the data manually. This is also a good solution if data of the same format is to be visualized again in the future. This is because the data cleansing functions can be stored by the system and applied when the same data format is detected.

[Revise this paragraph. It states that "Continue with original source data" is the only way forward. However, need to consider that automatic data cleansing and normalization functions maybe applied. Any other aspects of this paragraph which need changing?]

Of the 3 possible actions, the first one – Continue with the original source data, is the one which meets the seven criteria for Online Information Visualization which we set out in the Introduction. In fact, we envisage an iterative approach where the first pass through the system creates a "good enough" visualization using action 1. In the second pass through the system, the user may experiment with various fixes to the data. And in the third pass through the system, the user may decide to create data cleansing function by taking action 3. In this way, the user creates a lasting record of the functions necessary to make the data well suited for visualization. These functions can optionally be made available to other users of the system.

For the purposes of the worked example, we will assume that the user decides to alter the source data manually.

5.2 Data Cleansing and Normalization

There are two problems with the source data as it is originally provided:

Firstly, the *lastweek* Element is numeric most of the time (28 of 40 records), except when the song is a new entry (12 of 40 records), in which case it is set to the string "NEW". The Source Data Analyzer gives a probabilistic value to the Value Semantic based on how many records meet each criteria. As can be seen, the Source Data Analyzer gives more likelihood to the Element being numeric. So in this case, the user must intervene and provide a numeric value in place of NEW. In this case, we choose, 0.

Secondly, the *weeks* Element is deemed to be a Text value by the system. This is because every number is surrounded by parenthesis. In order to gain the true semantics from this Element, we must remove the parenthesis. Additionally, when the chart song is a new entry, then the value is set to a dash (–). Again, in order to gain most semantic value from this element, we need to change each dash to a 1. This is still semantically correct, since it accurately represents the number of weeks the song has been in the charts.

After defining operations for data cleansing, we re-analyze the data to produce table 4:

It can be seen that the Entities, *lastweek* and *weeks* have been updated in the table. The ambiguities which existed before have been resolved. This will improve the quality of the semantics derived by the Entity to Artefact Matcher. However, the *image* entity still has unresolved ambiguities. We will see that this will not have a big impact on the final visualization since we know that the entity is a URI which is sufficient information to allow a good semantic match to be found.

The remaining stages will not be described as the process is similar to that described in Section 3. The final visualization outputs are shown in figures 6,

XPath	Value type	Value category / Structure semantic	Uniqueness	Min	Max
top-forty	-	Container (root)	-	-	-
top-forty/chart	-	Container (object)	-	-	-
top-forty/chart/@position	Integer	Ordered Numeric	40 values (1.0)	1	40
top-forty/chart/lastweek	**Integer**	**Ordered Numeric**	**29 values (0.73)**	**1**	**30**
top-forty/chart/weeks	**Integer**	**Ordered Numeric**	12 **values (0.3)**	**2**	**15**
top-forty/chart/image	URL	UnOrdered Discrete	8 values (0.2)	-	-
top-forty/chart/artist	String	UnOrdered Discrete	40 values (1.0)	-	-
top-forty/chart/album	String	UnOrdered Discrete	40 values (1.0)	-	-
top-forty/chart/uri	URL	UnOrdered Discrete	40 values (1.0)	-	-

Table 4. Source data Entity analysis for the BBC Top 40 music chart dataset (after data cleansing and normalization)

7 and 8. The data files themselves and further example of visualizations are provided in the WebAppendix [ref].

6 Proof of Concept and User Evaluation

[Need to cover: subjects, environment, examples, questions, results, conclusions]

In order to conduct a user evaluation, we have built a Proof of Concept system which demonstrates the overall approach end-to-end. The system consists of a set of simple utility applications for Source Data Analysis and Entity to Artefact Matching. The system is integrated and driven from Altova's MapForce [ref] application which produces an XSLT file of the mapping. This is executed to produce a file in the target representation format.

We conducted an informal user evaluation on 6 subjects who came from a technical but non-computer science background. We used data from the BBC Top 40 chart music XML feed [ref]. We asked each subject to evaluate the quality of 3 visualizations:

1. Output from the proof-of-concept with no human assistance (no data cleansing, no manual adjusting of entity to artefact mappings)
2. Output from the proof-of-concept with minor human assistance (data cleansing, but no manual adjusting of entity to artefact mappings)
3. Output from the proof-of-concept with human assistance (data cleansing, and manual adjusting of entity to artefact mapping)

We found that users were able to comprehend some aspects of all the Online Information Visualizations produced. This was helped particularly by the user seeing the table stating which data entities corresponded to which representation artefact. All users detected at least one feature of the data.

Visualization 1 - Figure 6. The following cognitive insights were observed by users:

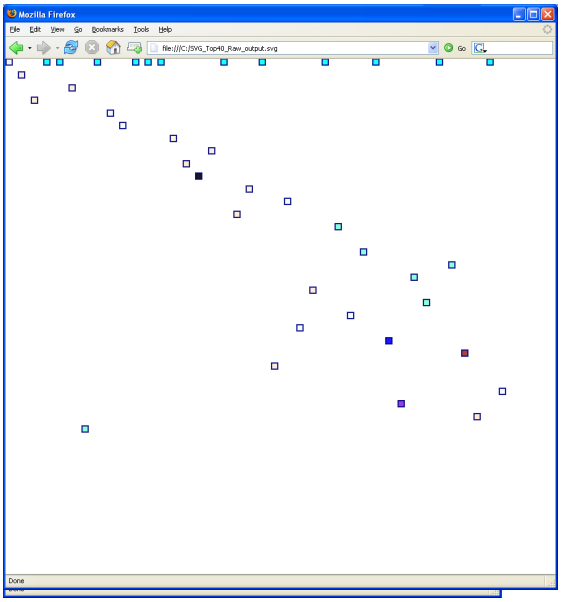

Fig. 6. Visualization 1 : No data cleansing, no manual adjustment of mappings

- Users could see a linear trend along the line x=y (when the origin is the top left hand corner). This represents the variation between the current chart position and last week's chart position.
- Some users noticed the outlier object which represents the chart's highest-climber (in this case, the band Snow Patrol).
- In addition, some noticed that there was certain objects at the top of the visualization. This represents songs which are new entries).

Visualization 2 - Figure 7. Users were able to see the same trends as Visualization 1 but also the following:

- The small squares exactly on the line x=y representing new entries.
- The different widths of the bars indicating the number of weeks the song has been in the charts.

Visualization 3 - Figure 8. Users agreed that Visualization 3 was the clearest (and most pleasing to look at), but gained no further data insights over Visualization 2.

These results are positive since they indicate that, although not perfect, Online Information Visualization has the potential to be used in circumstances where Traditional Visualization is not appropriate or possible. Also, we believe that through further development and added heuristics, we can increase the value

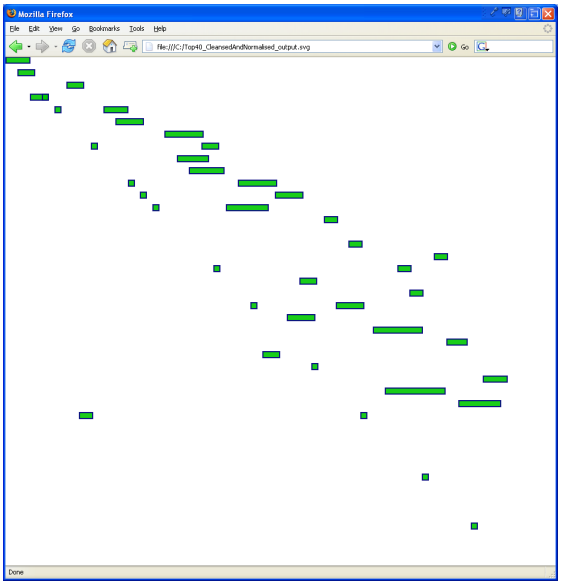

Fig. 7. Visualization 2 : Data cleansing, no manual adjustment of mappings

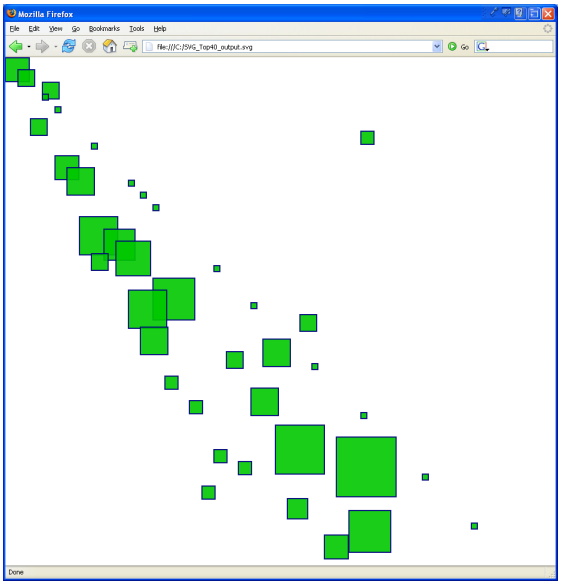

Fig. 8. Visualization 3 : Data cleansing, and manual adjustment of mappings

of Online Information Visualization towards the level offered by programmer developed Traditional Visualization.

7 Conclusions

In this paper we have defined the 7 characteristics of Online Information Visualization, comparing it with Traditional Information Visualization and thus providing a motivation for our work.

We have provided a description of our approach to this problem by breaking-down the process into a set of individual phases. We have described a set of core functions which are able to be specialized and then associated with the representation artefacts of any target representation format. These core functions are therefore able to be customized specifically for the target format which may in fact not use all of the available core functions.

We have devised a process for analyzing the source in order to categories Entity types into 3 fundamental types. Also this process can be used to detect parts of the data where data cleansing and normalization maybe required. We have then devised a simple rule-based algorithm for matching source entities to representation artefacts.

We have created a simple prototype using XML/XSD and XSLT technologies, coordinated through Altova's MapForce application. We have created multiple modules for performing utility tasks. In order to validate the value of the Online Information Visualization we conducted a basic user evaluation exercise which has shown encouraging results.

The main contributions of our work are:

1. A process which is independent of the Source Data Format
2. A process which is independent of the Target Representation Format
3. Exploits established research areas:
   (a) Ontology / Schema mapping
   (b) Data mining
   (c) Data cleansing
   (d) Information Visualization

However, as well as an integration of existing research areas, our approach is novel in that it:

- Introduces a new, general process based on the application of a set of generic functions chosen according to the target representation format.
- Involves the user in the Semantic Web
- Overcomes and embraces the idiosyncrasies of human created "semantic data"

Acknowledgments

This work took place during a research fellowship at the Knowledge Engineering and Decision Support Research Group (GECAD) at the Polytechnic Institute of Porto, Portugal. This research is funded by the University of Wales Swansea Postgraduate Research Studentship under the supervision of Dr. Phil Grant and Prof. Min Chen. The authors would also like to acknowledge FCT, FEDER, POCTI, POSI, POCI and POSC for their support to R&D projects and the GECAD unit.

References

1. Berners-Lee, T. and Fischetti, M.; "Weaving the Web The Original Design and Ultimate Destiny of the World Wide Web"; 1999.
2. Geroimenko, V., Chen, C.: "Visualising the Semantic Web"; Springer, Verlag / London (2006)
3. Mierswa, I., and Klinkenberg, R., and Fischer, S., and Ritthoff, O.: "A Flexible Platform for Knowledge Discovery Experiments: YALE – Yet Another Learning Environment"
4. Thompson, H.: "Normal Form Conventions for XML Representations of Structured Data"; Proc. of XML 2001, IDEAlliance, Alexandria, VA USA.
5. Ackerman: "The Visible Human Project," Medicine Meets Virtual Reality II: Interactive Technology & Healthcare: Visionary Applications for Simulation Visualization Robotics, San Diego: Aligned Management Associates, 5, (1994).
6. WebAppendix: Appendix to "Online Information Visualization: Automatic Mapping of Data Entities to Representation Artefacts" http://www.cs.swan.ac.uk/∼csowen/OnlineInfoViz/

8 To Do

1. future work
2. references
3. decide on British or American spelling
4. Use this: The mapping process defined here is a general solution in that there are no constraints on the source or target formats.