3D Visualization of Semantic Metadata Models and Ontologies*

Charalampos Papamanthou[1,2], Ioannis G. Tollis[1,2], and Martin Doerr[2]

[1] Department of Computer Science, University of Crete, Heraklion, Greece
{cpap,tollis}@csd.uoc.gr
[2] Institute of Computer Science, FORTH, Heraklion, Greece
martin@ics.forth.gr

Abstract. We propose an algorithm for the 3D visualization of general ontology models used in many applications, such as semantic web, entity-relationship diagrams and other database models. The visualization places entities in the 3D space. Previous techniques produce drawings that are 2-dimensional, which are often complicated and hard to comprehend. Our technique uses the third dimension almost exclusively for the display of the isa relationships (links) while the property relationships (links) are placed on some layer (plane). Thus the semantic difference between isa links and property links, which should be as vertical or as horizontal as possible respectively, is emphasized. Special reference is made on a certain model, the CIDOC Conceptual Reference Model.

1 Introduction

Semantic graphs (also called models) describe relationships between entities and are very important in many applications [1, 7–9]. They contain a great amount of information and the visualization of such models is very crucial in order to understand the details they incorporate. There have been several attempts to automatically produce drawings of these models. Previous techniques produce drawings that are 2-dimensional, and often complicated and hard to comprehend [5, 6]. We present a technique that produces 3-dimensional drawings that are clear and understandable. In these metadata models there are usually two different types of relationships: the *isa* relationships and the *property* relationships. Our technique uses the third dimension almost exclusively for the display of the isa relationships (links) while the property relationships (links) are placed on some layer (plane). These links have different meaning. In detail, an isa link represents a type of inheritance between the connected entities, whereas the property links correspond to different relationships between the entities. Since it is not always feasible, a few property links may use the third dimension.

Many automated tools for the display of such models have been developed. The tools are naturally categorized into two types: textual and visual [1]. Examples of textual RDFS browsing tools are Protege 2000 [2], Ontoedit [3] and

* This work was supported in part by INFOBIOMED code: IST-2002-507585 and the Greek General Secretariat for Research and Technology under Program "ARISTEIA", Code 1308/B1/3.3.1/317/12.04.2002.

J. Pach (Ed.): GD 2004, LNCS 3383, pp. 377–388, 2004.

Ontomat[1]. However, text-based environments have not proved effective [3, 4]. On the other hand, visualization based tools provide the user with a more general view of the whole model. There are many tools visualizing RDF browsing such as IsaViz [5], FRODO RDFSViz [6], and OntoViz[2].

All these visualization tools produce 2D visualizations of the input model. However, 2D visualizations of these models are complex and confusing, especially when the number of entities of the specific model is large. Furthermore, in two dimensions, it is difficult to understand the meaning of each link, as they are all looked upon under the same perspective. Our approach uses the third dimension for the visualization of the isa links and the input model will be considered as a system of interactions between entities, which push the nodes of the graph towards a specific optimal position. Till now, to the best of our knowledge, no 3D models for the automated visualization of semantic metadata models have been presented.

In this paper, we present an algorithm that produces 3D solutions to the semantic metadata and ontologies visualization problem. The algorithm tries to implement intuitive preference directions (isa links upwards, property links horizontal), that assist the intellectual orientation and understanding of the human spectator. If the number of entities and properties is large, this is virtually impossible to be accomplished in 2D representations. Our algorithm can be applied in order to visualize many ontology models, including the ABC ontology and model described in [7]. We present our results for a conceptual reference model, the CIDOC CRM.

The **CIDOC Conceptual Reference Model (CRM)** provides definitions and a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation [8, 9]. The CIDOC CRM is a directed graph where nodes represent entities and edges represent property and isa relationships. This graph is isa-disconnected (it contains more than one isa-connected components) and has many interesting characteristics. The complete definition of CIDOC CRM can be found in [9]. An example model of CIDOC CRM is shown in Figure 1.

2 The Problem

Our target is to create a 3D-mapping of a metadata model. The model describing entity relationships is given in various formats, such as XML or RDF.

As the description of the model consists of two kinds of relationships (isa and property links) between entities, we can extract a directed graph $G = (V, E)$ such that V is the set of entities and E is divided into two sets of edges, I, which denotes the set of isa links and P, which denotes the set of property links. Throughout the paper n will denote the number of the original model entities (vertices) whereas m will denote the number of the original model links (edges).

[1] http://annotation.semanticweb.org/ontomat

[2] Ontoviz is a visualization plug-in for Protege 2000.

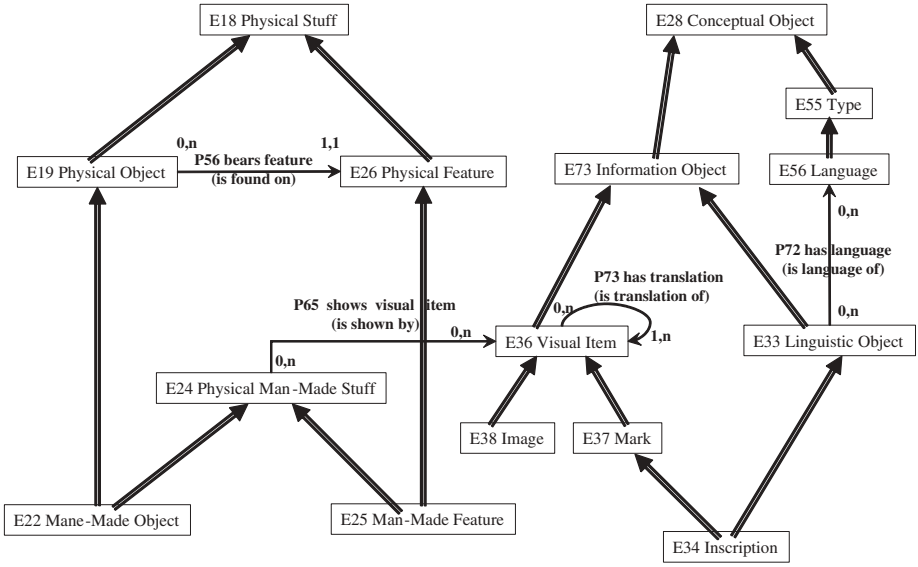

Fig. 1. A CIDOC CRM model (mii) consisting of 15 entities (nodes).

Therefore, we reduce the problem of metadata model 3D visualization to the computation of a 3D embedding of the respective directed graph. Thus, we want to create a 3D visualization of G (i.e., to compute (x, y, z) for all $v \in V$) such that for all edges $(v, u) \in I$ it is $z(v) > z(u)$. The inequality represents a parent-child relationship between two nodes that are connected with isa links. In this way, the importance of isa links is pointed out. Also it is desirable (but clearly not always feasible) to have $z(v) = z(u) + 1$ for each $(v, u) \in I$ (i.e., all the isa-connected nodes lie in adjacent layers). Additionally, we want to minimize the number of edges $(v, u) \in P$ that connect nodes that lie in different layers. This means that our main objective is to place nodes that are connected with property links on the same layer. In this way, an absolute discrimination between the two types of edges is achieved. Finally, minimizing edge crossings per layer is desired. Due to the computation of the (x, y, z) coordinates, crossings between isa links are unlikely to occur, as the probability of independent placed line segments crossing in the 3D space is almost zero. The satisfaction of all three constraints is not always feasible. Thus, in the solution that we will present, some trade-offs have to be made in order to achieve a feasible 3D embedding of the graph. Figure 2 shows an ideal case of a 3D embedding: For all edges $(v, u) \in I$ it is $z(v) = z(u) + 1$ (i.e., all isa links connect nodes lying in adjacent layers). Also, for all edges $(v, u) \in P$ it is $z(v) = z(u)$ (i.e., all property links lie on the same layer). Finally, no crossings between property links per layer and between isa links exist.

The constraints presented are general constraints that have to be taken into consideration in order to produce an aesthetically good and readable drawing. During the description of the algorithm, some more constraints will appear that

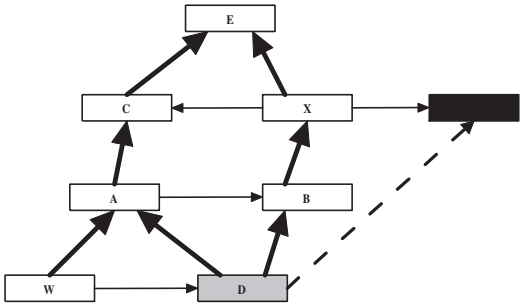

Fig. 2. An ideal embedding. If either a property or an isa link is added between the gray and the black node, the ideal embedding cannot be preserved.

will aim to improve the final drawing. The main body of our algorithm is as follows:

Algorithm 3DVis
Input: A directed graph $G = (V, E = I \cup P)$
Output: A 3D embedding of G

1. **Layer Assignment Stage**
   For each $v \in V$ compute $z(v)$.
2. **Pseudo Nodes and Pseudo Edges Addition Stage**
   Decompose each *isa* link (i, j), such that i, j do not lie in adjacent layers, by introducing *dummy isa* nodes at the layers between i and j.
   Decompose each *property* link (i, j), such that i, j do not lie in the same layer, by introducing one *dummy property* node either at the layer of node i or at the layer of node j.
3. **Coordinate Assignment on Layers Stage**
   Assign (x, y) coordinates to the nodes of the graph (including pseudo nodes).
4. **Return** the computed 3D embedding of $G = (V, I \cup P)$.

In the following sections, we describe the three main stages of the algorithm.

3 Layer Assignment

Layer assignment is equivalent to determining the z-coordinate of each node. The layers are numbered $0, 1, \ldots, L-1$, where L is a positive integer determined by the algorithm. Many layer assignment algorithms have been proposed with the *longest path* layering algorithm being the dominant one [10]. However, the longest path layering, though initially implemented, is not finally used, as it does not take into consideration the property links of each node.

In order to assign layers, both property and isa links should be taken into consideration as we want to place the nodes in a way that minimizes the total number of layers intersected by the edges. Thus the number of pseudo nodes

(dummy property and dummy isa nodes) is also minimized. If the graph had only isa links then a technique similar to that proposed in [11] for layered drawing of digraphs could be used (see also [10]).

Let $G = (V, I \cup P)$ be the input graph model to visualize, where V is the set of nodes (entities) of the model, I is the set of isa links and P is the set of the property links. Our first aim is to run an algorithm in order to identify the isa-connected components of the induced undirected graph. This can be easily done, by applying a depth first search that computes the isa connected trees in the depth first search forest.

Suppose now that we have identified the isa DFS trees that correspond to the input model. In the general case, we suppose that we have k such connected subgraphs $G_j = (V_j, E_j)$, $j = 1, \ldots, k$. Let n_j denote the number of nodes of each connected subgraph (i.e., $n_j = |V_j|$). It is clear that $\sum_{j=1}^{k} n_j = n$. We want to compute $z(i)$ for all $i = 1, \ldots, n$. Throughout the paper, we assume that z *increases* by moving downwards. One very important consideration is that the z-coordinates of the nodes must always satisfy the order imposed by the isa hierarchies. The isa hierarchy for all the nodes of the graph is derived from the input model.

Additionally, we would like to impose an upper bound L to the z-values. L could be the number $\max_{i=1,\ldots,k}\{n_i\}$ (i.e., the number of the nodes of the largest DFS tree). Another approach for the definition of L could be the longest path of the directed acyclic graph that is produced by the model if we ignore all the property links. Furthermore, we must define an objective function of each solution. It is desired that a layer assignment is achieved such that the vertical distance between nodes which are connected with property links is minimized. Additionally, we want to ensure that vertical distance between nodes that are connected with isa links is also minimized. Due to the nature of the models we want to visualize, minimizing vertical distance between nodes which are connected with property links is more important than minimizing vertical distance between nodes which are connected with isa links. Thus we minimize the former with priority α and the latter with priority β, where $\alpha > \beta$. Therefore, we define the following integer non-linear optimization problem of the function f:

$$\min f(z) = \alpha \sum_{(i,j)\in P} (z(i) - z(j))^2 + \beta \sum_{(i,j)\in I} (z(i) - z(j))^2$$

$$s.t.\ z(i) > z(j)\ \forall (i,j) \in I; z(i) < L\ \forall i \in V; z(i) \in \mathbb{Z}\ \forall i \in V$$

The first constraint ensures that the isa hierarchy is preserved, the second constraint imposes an upper bound to the values the depths of the nodes can range, while the third one forces an integer solution. Figure 3 shows two legal layer assignments of the same model. The superiority of the second one is clear, since all property links lie on the layers.

The optimization problem, which is at least NP-hard [22], can be solved with integer programming approximation algorithms. However, if we force direction to the property links, it can be solved optimally by using the method proposed in [11]. We have solved the problem using an optimization package, called Lingo (version 8) (see **http://www.lindo.com**).

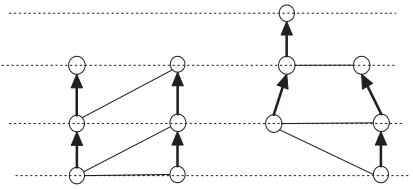

Fig. 3. Two layerings z_1, z_2. The isa hierarchies in both layerings are preserved. Note that $f(z_1) = 2\alpha+4\beta$, while $f(z_2) = \alpha+4\beta$. Our layer assignment algorithm will choose the second layout.

4 Pseudo Nodes and Edges Addition

After the layers of the nodes have been computed, we must assign the (x, y) coordinates to each node. Let z_0 be the layout that minimizes f. We have to consider the following two cases:

Case 1 – Property Links
There can exist property links (i, j) such that $z_0(i) \neq z_0(j)$ (i.e., i, j are not placed on the same layer and $f(z_0) > \beta \sum_{(i,j)\in I} (z_0(i) - z_0(j))^2$). Suppose we have p such links. In this case, every one of these p property links will originate from a certain layer and will terminate to another one. Thus we introduce the idea of a *dummy property* node, which will actually appear as a bend in the final visualization. For each pair of nodes i, j that are connected with a property link such that $z_0(i) \neq z_0(j)$, a dummy property node $c(i, j)$ is placed at layer $\min\{z_0(i), z_0(j)\}$. Additionally, it is desired that $c(i, j)$ lies exactly above (in terms of (x, y) coordinates) of the node $v = i$ or $v = j$ such that $z_0(v) = \max\{z_0(i), z_0(j)\}$. A dummy property node $c(i, j)$ is thus defined as a pseudo node of an original node i which is connected to a node j through a property link such that $z_0(i) \neq z_0(j)$. Following this approach, all property links will be drawn as orthogonal lines at the cost of introducing one bend per property link. Additionally, an upward *property tree* is formed (see Figure 4b). In the case that a node j has more than one property destinations which all lie on the same layer, different from $z_0(j)$, then all the created copies are merged into one.

Another approach is to create a *property path* between the respective nodes. In this way, the number of dummy nodes would increase and the semantic difference between the property and isa links would not be so clear. Additionally, this approach is not so useful, as in such metadata models, the number of the property links is much less than the number of isa links.

Case 2 – Isa Links
There can exist isa links (i, j) such that $z_0(i) > z_0(j)+1$ (i.e., i, j are not placed in adjacent layers and $f(z_0) > \alpha \sum_{(i,j)\in P} (z_0(i) - z_0(j))^2 + \beta|I|)^3$. For each isa link (i, j) such that $z_0(i) - z_0(j) > 1$ we define a set $F^{(i,j)}$ of $z_0(i) - z_0(j) - 1$

[3] $|X|$ denotes the cardinality number of set X.

dummy isa nodes that will be placed on layers $z_0(j)+1, z_0(j)+2, \ldots, z_0(i)-1$. Note that $F^{(i,j)} = \emptyset \Leftrightarrow (i,j) \in I \wedge z_0(i) - z_0(j) = 1$.

Another way to describe this is as follows. Suppose we draw a virtual straight line segment between two nodes i and j that are connected with an isa link and lie in different layers. Dummy isa nodes are introduced on each layer pierced by this virtual line segment. These nodes are stored in $F^{(i,j)}$. $F^{(i,j)}$ defines a path of $z_0(i) - z_0(j) - 1$ new isa links $i, i_1, i_2, \ldots, i_r, j$ where $r = z_0(i) - z_0(j) - 1$.

Finally, the total number of added dummy isa nodes is $d = \sum_{(i,j)\in I \wedge F^{(i,j)} \neq \emptyset} |F^{(i,j)}|$. It is clear that $d \leq \sum_{F^{(i,j)} \neq \emptyset} (L-1)$. In Figure 4, we can see how the addition of the dummy nodes is achieved.

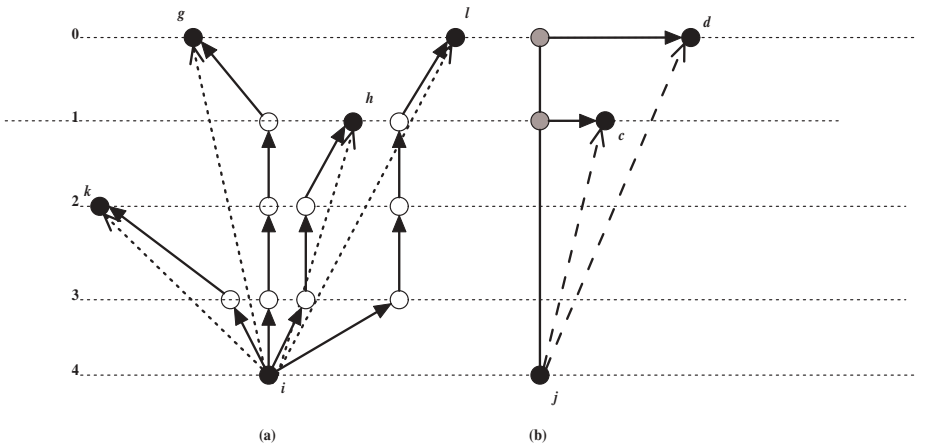

Fig. 4. Dummy nodes and edges addition. In (a), node i has 4 isa links: (i,k), (i,g), (i,h), (i,l). In (b), node j has has 2 property links (j,c) and (j,d).

Therefore, the total number of the inserted dummy nodes is $p+d$. It is easy to prove that no dummy nodes are added if and only if $f(z_0) = \beta|I|$. In this case $p = 0$ and $F^{(i,j)} = \emptyset \ \forall (i,j) \in I$. The existence of dummy nodes is important in the next phase, as they play the role of placeholders and bends in the final drawing.

Finally, one more step has to be performed. As the model behaves as a dynamic system, we must connect each dummy property node $c(i,j)$ of two property connected nodes i, j with a new link between $c(i,j)$ and the original node (i.e., the node copied). This can be done if we introduce a pseudo-isa relationship between each dummy property node and the node from which it has been copied. There will therefore be p pseudo-isa relationships (as many as the dummy property nodes). The final number of isa links will be $|I'| = |I|+p+d$. Thus, after the addition stage is complete, our graph will consist of $|V| + p + d$ nodes, exactly $|P|$ property links and $|I|$ isa links. In the following sections, V, I will denote the *augmented* sets of nodes and isa links respectively. It is important to state that in the final visualization, there would be only $|I'|-p$ isa links, as the pseudo-isa links only contribute to maintain the dynamic aspect of the model.

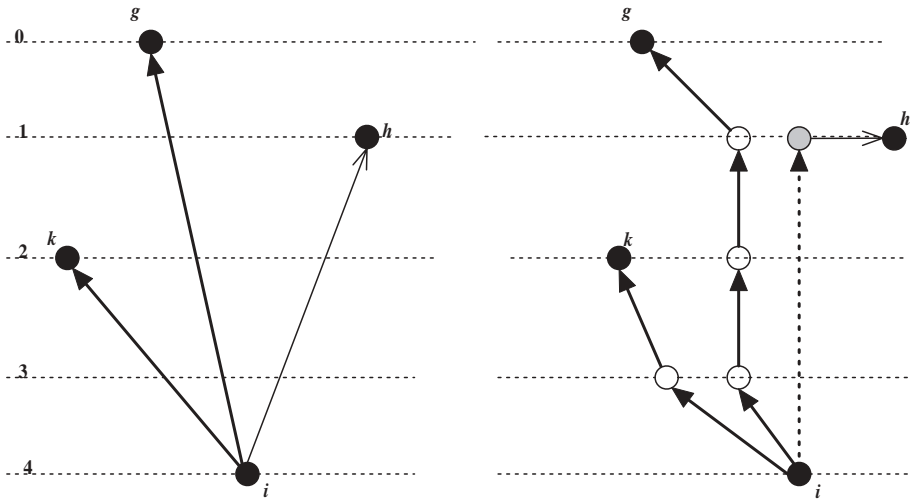

Fig. 5. The function of the pseudo node and edges addition algorithm. The initial model has 3 relationships $ISA(i,k)$, $ISA(i,g)$ and $PROPERTY(i,h)$.

The overall function of the pseudo-node and edges addition algorithm is depicted in Figure 5.

Finally, we could use an alternative approach to this problem. Instead of adding dummy nodes for each isa link (i,j), we can create an *isa tree* from a node i to all its isa destinations, similar to the property tree. The difference of this approach is evident only in the case of a node i having h isa links of type $(i,j_1),(i,j_2),\ldots,(i,j_h)$. This approach reduces the number of the dummy isa nodes. However, the connection between all isa-connected nodes is not very clear. Finally it depends upon the application as to which one of the two alternatives will be chosen. In our case, the first alternative is more satisfying. Figure 6 depicts the dummy isa nodes addition according to the second alternative. Note that the approach of using isa and property trees is trying to mimic the ideal visualization case described in Figure 2.

5 Coordinate Assignment on Layers

In this section, we present the method for the computation of the (x,y) coordinates of each node. Our main objective here is to minimize, with weight α, the 3D distance of all the pairs of nodes that are connected with isa links. In this way, the *isa children* of a node v will be placed symmetrically around v. Furthermore, with lower priority β, we want to minimize the horizontal distances between nodes that are connected with property links. Note, that after the introduction of the dummy property nodes, property links connect nodes lying on the same layers.

This minimization is made under the constraint that nodes do not overlap. Hence, we must ensure that for each layer there is a minimum horizontal distance,

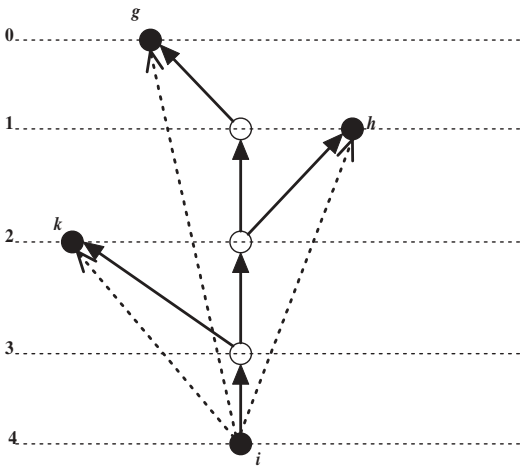

Fig. 6. Forming an ISA tree. Initially, there are 3 isa links: (i, k), (i, g), (i, h).

say 1, between nodes belonging to a certain layer. Additionally, it is desired that for each node i that has a dummy property node j to another layer it is $p(i) = p(j)$ which means i and j have identical x and y coordinates (i.e., it is $x(i) = x(j)$ and $y(i) = y(j)$).

Furthermore, we would like to have the dummy nodes of each isa link (i, j) placed on a straight line. In this way the hierarchy is clearer. For this purpose, we examine two cases: **(a)**, If node i has only **one** isa neighbor j such that $z_0(i) > z_0(j) + 1$, then we want to ensure that

$$p(i) = p(i_1) = p(i_2) = \ldots = p(i_r) \tag{1}$$

where $i_1, i_2, \ldots, i_r$ are the nodes introduced in section 4 and $r = z_0(i) - z_0(j) - 1$. **(b)**, If there are **more** than one isa links originating from i, say h, and we want to ensure that for each isa neighbor j_k, $k = 1, \ldots, h$, the dummy isa nodes that belong in $F^{(i,j_k)}$ are related as follows

$$p(i_1) = p(i_2) = \ldots = p(i_r) \tag{2}$$

where now $r = z_0(i) - z_0(j_k) - 1$, $k = 1, \ldots, h$. Note that only in the first case the origin of the isa link is placed on the same vertical straight line with the dummy isa nodes. This is not feasible in the second case, as we want to separate the multiple isa links originating from i. In both cases, the destination of the isa link is not forced to lie in a specific position. A desired layout for the second case can be seen in Figure 4. This is a technique inspired by polyline drawings [21].

All these observations finally lead to the following non-linear optimization problem:

$$\min g(x, y) = \alpha \sum_{(i,j)\in I} d_{ij} + \beta \sum_{(i,j)\in P} d_{ij}$$

$$s.t. \quad (x(i) - x(j))^2 + (y(i) - y(j))^2 \geq 1 \ \forall (i, j) : z_0(i) = z_0(j) \tag{3}$$

$$p(t) = p(c(i, j)) \ \forall c(i, j), \ t = arg\max\{z_0(i), z_0(j)\}; C_1; C_2 \tag{4}$$

where $d_{ij} = \sqrt{(x(i)-x(j))^2 + (y(i)-y(j))^2 + (z_0(i)-z_0(j))^2}$ is the 3D Euclidean distance of nodes i, j, C_1, C_2 are the constraints imposed by (1),(2).

Constraint (4) ensures minimum distance per layer, whereas the other constraints ensure that the added nodes are placed on straight lines according to the rules we have presented. This problem, which is also at least NP-hard [22], has also been solved with Lingo 8, in polynomial time on the size of input [12].

Instead of solving the above optimization problem, we can use a force directed algorithm. This can be achieved by using some extra dummy nodes, which will be regarded as fixed. There are many force directed algorithms proposed such as [13–16]. Finally, we can examine if the hierarchical approach [17] or maybe an orthogonal drawing approach [18–20] could be implemented. These are all goals of future work.

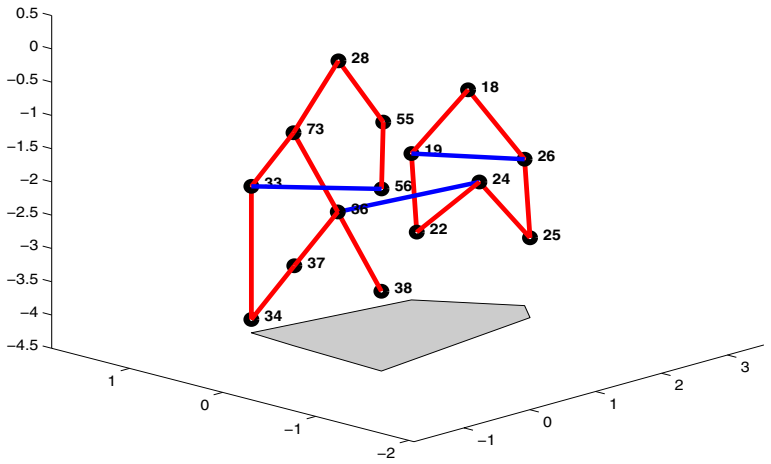

Fig. 7. 3D Representation of the Mark Inscription Information (mii) Model (Figure 1).

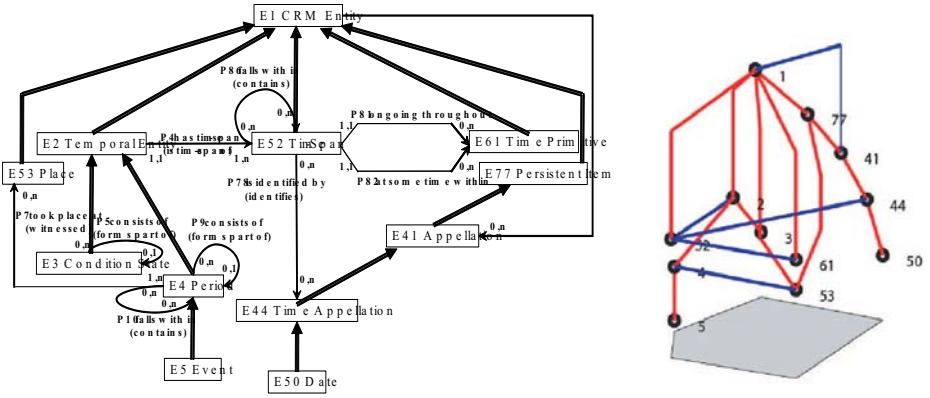

Fig. 8. 3D Representation of the Time Span Information (tsi) Model.

6 Time and Experiments

We have implemented our algorithm and have run it on several examples. The time needed by our algorithm is divided into three main parts: **(1)**, The time needed for minimizing f. Lingo uses efficient approximation algorithms and performs the layer assignment in polynomial time. **(2)**, The time needed for determining dummy nodes. Let L^* be the maximum number of layers needed after the layer assignment stage is complete. This time is obviously $O(mL^*)$, as we need to check all links (isa and property links) in order to decide if the addition of dummy nodes is necessary or not. **(3)**, The time needed for the minimization of g. This part is the most expensive one, as there are many constraints and the size of the problem has increased (due to the introduction of dummy nodes and edges). However, Lingo provides an approximation in reasonable time.

Here, we present some visualization examples produced by our software with reference to two CIDOC CRM models. Isa and property links are depicted with red and blue lines respectively. The time our algorithm needed in order to produce the 3D drawings is 4 seconds in average.

References

1. Alexandru Telea, Flavius Frasincar, Geert-Jan Houben, "Visualizing of RDF(S) based Information" , Proc. IEEE IV '03, IEEE CS Press, 2003
2. N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen, "Creating semantic web contents with protege-2000". IEEE Intelligent Systems, 16(2):60–71, 2001
3. Y. Sure, S. Staab, J. Angele. "OntoEdit: Guiding Ontology Development by Methodology and Inferencing" In: Proceedings of the International Conference on Ontologies, Databases and Applications of SEmantics ODBASE 2002
4. S. Card, J. Mackinlay, and B. Shneiderman, "Readings in Information Visualization". M. Kaufmann, 1999
5. E. Pietriga, "Isaviz: a visual environment for browsing and authoring rdf models", The Eleventh International World Wide Web Conference (Developer's day), 2002.
6. Frodo RDFS Visualization Tool, http://www.dfki.uni-kl.de/frodo/RDFSViz/.
7. Lagoze C, Hunter J., "The ABC Ontology and Model" , Proceedings of the International Conference on Dublin Core and Metadata Applications 2001, p. 160–176.
8. Nick Crofts, Martin Doerr, Tony Gill, Stephen Stead, Matthew Stiff (editors), "Definition of the CIDOC Conceptual Reference Model", March 2004 (3.4.10).
9. Doerr M., "The CIDOC CRM – An Ontological Approach to Semantic Interoperability of Metadata", AI Magazine, 4, issue 1 (2003).
10. G. Battista, P. Eades, R. Tamassia, I. Tollis, "Graph Drawing – Algorithms for the Visualization of Graphs" , (1999).
11. E.R. Gausner, E. Koutsofios, S.C. North and K.P. Vo, "A Technique for Drawing Directed Graphs", IEEE Trans. Softw. Eng., 19, 214–230, 1993.
12. Mihir Bellare and Phillip Rogaway, "The complexity of approximating a nonlinear program", Journal of Mathematical Programming B, Vol. 69, No. 3, pp. 429–441
13. P. Eades and X. Lin, "Spring Algorithms and Symmetry", In Proceedings of COCOON 1997, vol 1276 of Lecture Notes in Computer Science, pp. 109–112. Springer.

14. T. Fruchterman and E. Reingold, "Graph Drawing by Force Directed Platcement", Softw.-Pract. Exp., 21, no. 11, 1129–1164, 1991.
15. T. Kamada and S. Kawai, "An Algorithm for Drawing General Undirected Graphs", Inform. Process. Lett., 31, 7–15, 1989.
16. T. Kamada, "Visualizing Abstract Objects and Relations", World Scientific Series in Computer Science, 1989.
17. K. Sugiyama, S. Tagawa and M. Toda, "Methods for visual understanding of hierarchical systems", IEEE Trans. Syst. Man Cybern., SMC-11, no. 2, 109–125, 1981.
18. Achilleas Papakostas, Ioannis G. Tollis, "Algorithms for Incremental Orthogonal Graph Drawing in Three Dimensions", J. Graph Algorithms Appl. 3(4):81–115
19. Achilleas Papakostas, Ioannis G. Tollis, "Algorithms for area-efficient orthogonal drawings", Comput. Geom. 9(1-2):83–110 (1998)
20. Achilleas Papakostas, Ioannis G. Tollis, "Interactive Orthogonal Graph Drawing", IEEE Trans. Computers 47(11):1297–1309 (1998)
21. G. Di Battista, R. Tamassia, and I. G. Tollis, "Area requirement and symmetry display of planar upward drawings", Discrete Comput. Geom., 7:381–401, 1992
22. Michael R. Garey, David S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", W H Freeman & Co., 1979