Copyright © 2009 by ABCM

CAD SOFTWARE FOR CABLE DESIGN: A CONSISTENT MODELING METHOD TO DESCRIBE THE CABLE STRUCTURE AND ASSOCIATED INTERFACE

Rodrigo Provasi, rodrigo.provasi@gmail.com
Christiano Odir Cardoso Meirelles, christiano_odir@yahoo.com.br
Clóvis de Arruda Martins, cmartins@usp.br
Universidade de São Paulo - Departamento de Engenharia Mecânica

Abstract. *The concept and project of umbilical cables and flexible pipes are not simple tasks, due to the great variety of elements and possible arrangements. The design of such cables is based on what structural and operational functions the cable is intended to perform. Possible cables' components are electrical cables and hydraulic hoses, to control underwater equipment; protective sheaths; helically wounded tensile armors for structural resistance; and anti-wear layers. Due to such complexity, a CAD tool for specific design is very useful. This paper deals with a two-dimensional CAD, that is a module of a bigger project, which encompasses not only the two-dimensional visualization, but also includes a numerical solver and a three-dimensional visualization module. The data structure is the most important part of the software, due to the various cable configurations and component distributions. An efficient and generic structure is relevant for the cable description. This paper deals with this data structure, as well the associated software interface. Both are described throughout the paper and, to conclude, an example of a typical geometry using the software is shown.*

Keywords*: Umbilical Cable, Flexible Riser, CAD Tool, Cable Modeling*

1. INTRODUCTION

Umbilical cables and flexible pipes are largely used in offshore industry. These elements are usually compound by a certain number of layers, each one with a specific function, such as axial stiffness, internal and external pressure resistance, fluid flow, electric and hydraulic control. Normally these cables are designed for a specific scenario and, for this reason, a great number of configurations can appear. Figure 1 and Figure 2 show examples of an umbilical and a flexible pipe configurations respectively:

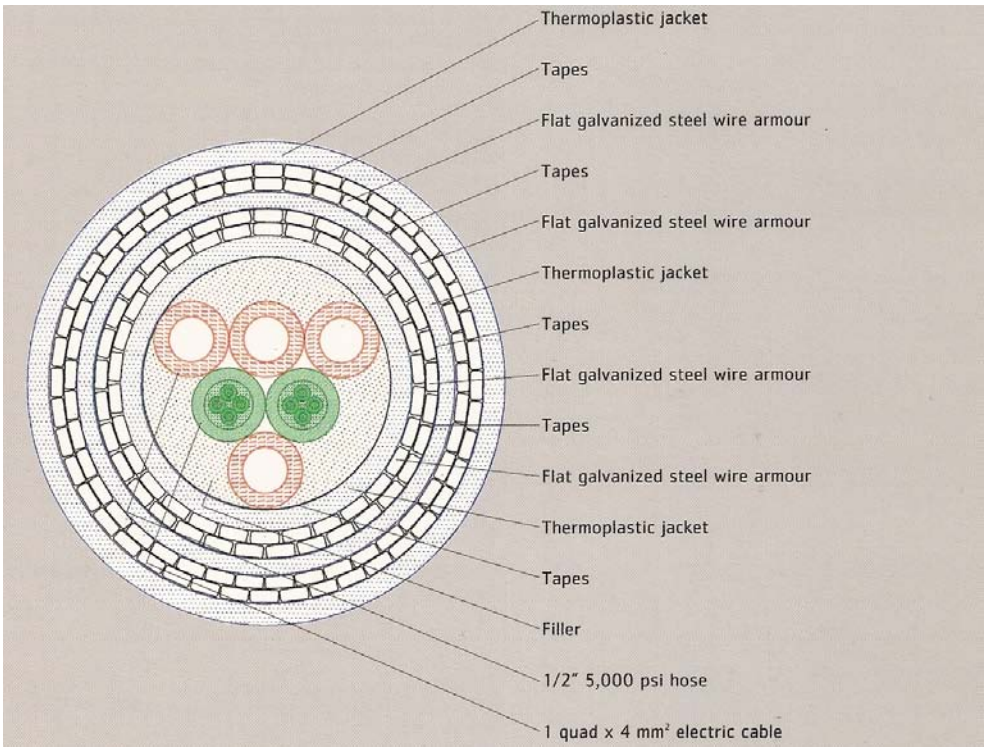

Figure 1. Example of an umbilical cable cross-section (extracted from Prysmian Catalog)

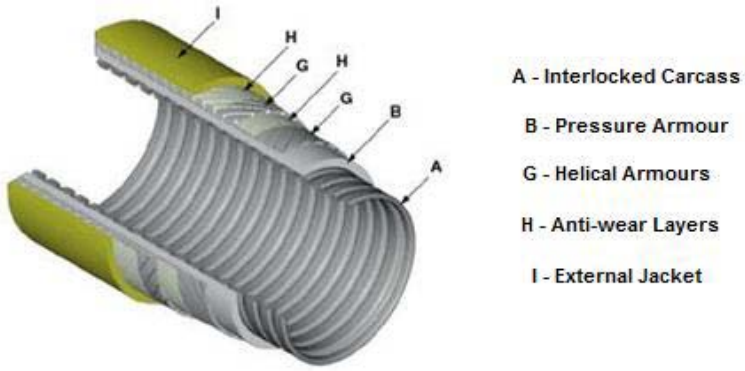

Figure 2. Typical Flexible Riser (extracted from Wellstream Catalog)

Dealing with these cables is not a simple task, not only in cable design but also in cable modeling. Figure 3 shows the process of designing such cables and also illustrates the various steps that must be taken in order to achieve an operational cable. Because the modeling and the simulation of this cables strongly depends on the cable design, a tool that comprehends all modules, from design to data and results visualization, passing by the stress and displacements calculations, is desired. Such tool is being developed and it consists of a set of three modules. The first of this modules (the two dimensional Computer Aided Design software) is completed and it is a tool for cable cross-section design. The second module is the solver, which is currently under development, and consists in a numerical model for stress and displacements calculation. The last of the modules is a three dimensional tool for post-processing the results and visualize the geometry. This paper deals with the first module: the two dimensional CAD for umbilical cable and flexible risers.

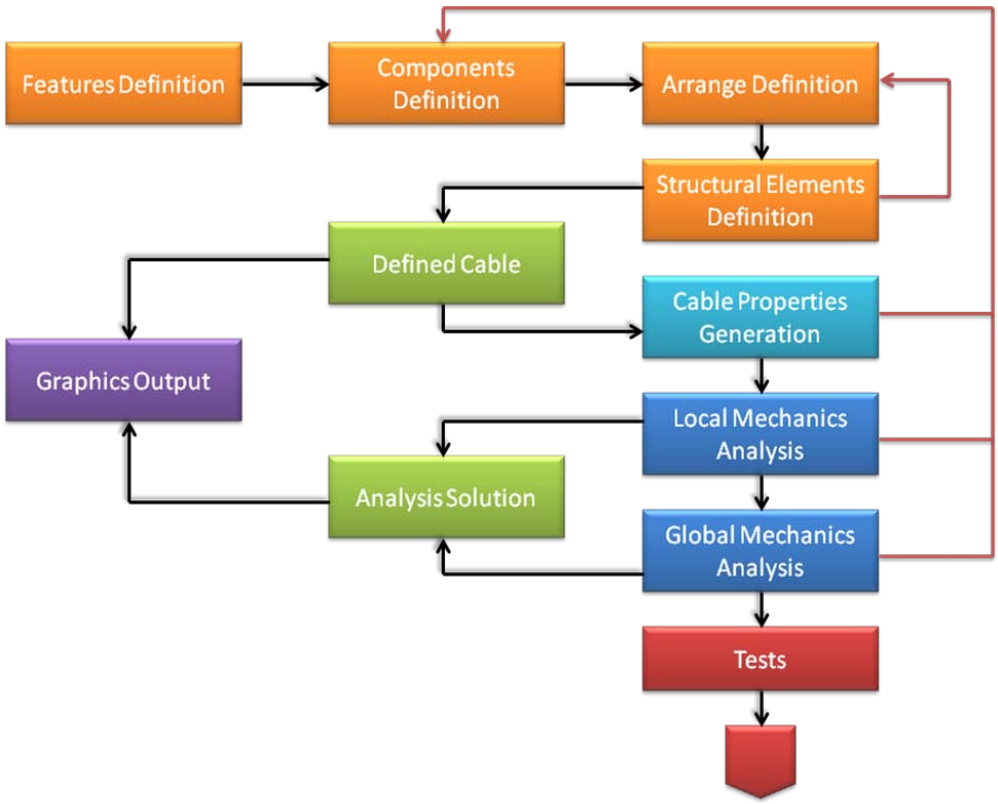

Figure 3. Cable design Process

The task of describing these cables is even more complicated in a computer software, due the variety of possible configurations and because it requires the structure to be generic (meaning that if a new component is inserted, a few or none work is required in the rest of the data structure) while it must be flexible to accept changes. This concern appears when analyzing the flow diagram in Figure 3. In the early stages (which comprehends the blocks from the Features Definition to the Cable Properties Generation and includes the Defined Cable and part of the Graphical

Output), a cable configuration must be evaluated plenty of times until a proper design is achieved. After that, the cable response and its properties are evaluated. These iterations on the process are necessary because sometimes the solution requires components that are not available in the market or the designed cable does not meet some of the requirements (such as weight, equivalent stiffness or dimensions, for example).

This paper initially shows a description of the cable's most common elements, followed by a discussion and explanation of the structure achieved during the execution of the project. The CAD interface is described briefly, showing the toolbars and dialogs of the software. To conclude, the paper brings an example of a typical umbilical cable described using the software.

2. *CABLES MOST COMMON COMPONENTS*

Umbilical cables and flexible pipes have many in common. Certain types of components are present in both cables, such as helical wounded reinforcement armors, jackets and fillers. This section gives a short description of each element and its functions.

Interlocked Carcass: the interlocked carcass is the most internal layer in flexible pipes and must resist to the external pressure and an eventual squeezing pressure. Figure 4 shows a typical profile of an interlocked carcass.

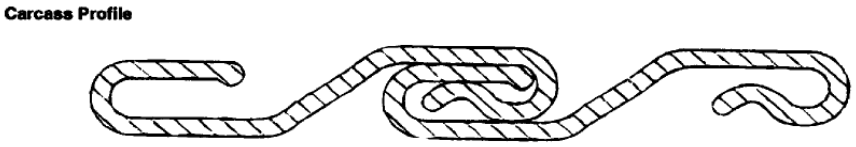

Figure 4. Typical interlocked carcass profile (from API RP 17B).

Plastic Jacket: the plastic jackets can be internal or external to the riser. The internal ones usually provide fluid tightness to the cable, while the external ones, which also provide fluid tightness, protect the internal elements from external damage.

Pressure Armor: the pressure armor is responsible for resisting internal pressures and also external pressures in case of the external sheath fails. Its presence is defined by the magnitude of the acting pressures. If the involved pressures could be supported by the helical armor, there is no need to include such a layer. This type of layer has several possible profile shapes, as the ones shown in the API RP 17B.

Helical Armor: the main function of the helical armor is to give the structure axial resistance, but it also offers a little radial resistance. It is commonly used in pairs, with opposite lay angles. The armor is compound by several tendons, usually with circular or rectangular cross-sections.

Anti-wear Layer: this type of layer is found between metallic elements, such the helical armor pairs and is used to reduce the friction between metallic components.

Fillers: the function of the fillers is to fill in the gaps between elements to guarantee their positioning.

Electric and Optical Cables: these cables provide electrical and optical signals for undersea equipments control.

Hose: hoses provide fluid injection and also hydraulic control for undersea equipments.

3. *SOFTWARE REQUIREMENTS AND DATA STRUCTURE*

This section deals with the software requirements and how they influence in data structure choice. Briefly, the software requirements is a series of tasks and features that the software must implement. The software requirements for the two-dimensional CAD are:

- Easy-to-use software, demanding few hours of training for basic operations
- Avoid failures (the software must perform a check of user operations)
- Easy user interface, using both mouse and keyboard, toolbars that can be re-arranged and hidden
- Allow both automatic and manual element adjust
- Allow modifications (such as insertion and exclusion) in a simplified way
- Allow the re-arrangement of elements
- Include features such as: cut, copy, position controlling, undo and redo
- Allow read and write binary files and import and export XML files

With these requirements in mind, a research in software engineering has been made (see Sommerville (2003)) . The selected design approach was the spiral one, because it was shown as the most suitable design process, once there could be uncertainties in the components to be implemented as well as in the interface.

As already said, the structure must be robust to avoid failures and possible inconsistencies and allow the insertion, modification and exclusion of elements in a simplified way. Many approaches to accomplish this task could be used, and the possible solutions are described in the following paragraphs.

Figure 5 shows a first approach to such structure:

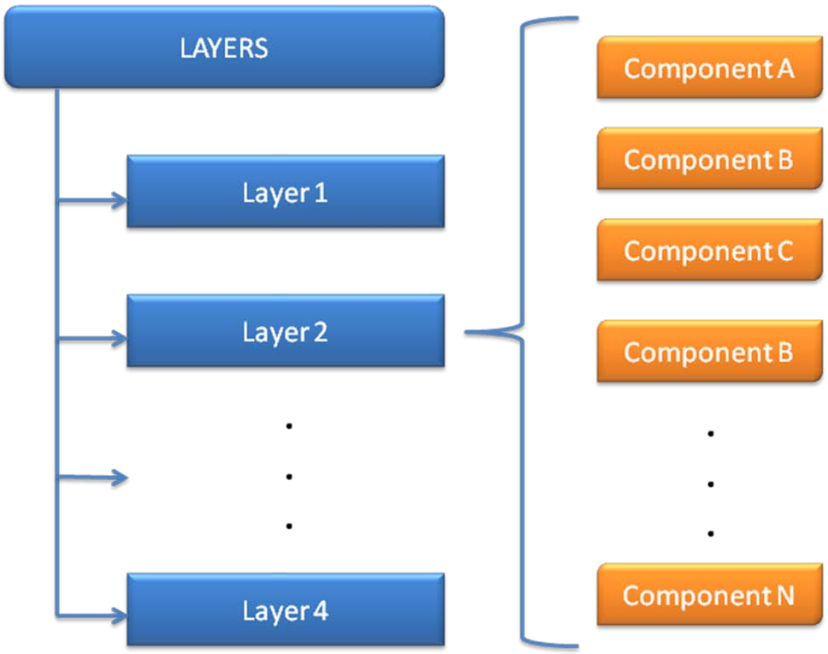

Figure 5. First possible cable structure

This approach consists in creating a list of layers and, for each layer, a list of components. The advantage of these list structures is that they are straightforward and intuitive. The disadvantage is that all components must be created as a separated component (even if it is of the same type, with the same material). In a first thought, a simple array structure could solve the problem, but, examining further, this solution does not fit the requirements of flexibility. Note that the meaning of flexibility in this context is to arrange, insert and exclude elements of the data structure as needed. The natural substitute for the array in this context is the linked list, which is the most suitable data structure for this purpose. More details in data structures could be found in Nell (2003), which treats with classic data structures and Marshall (2008), which treats with Collections and data structures in the .NET Framework (this was the major reference, due the implementation uses C# language).

This initial structure raises some questions, such as, could the performance be improved, with the flexibility of linked lists, using some kind of reusable components? The answer was obtained using two different data storages that work together. This idea is usually seen in design softwares and it is applied to this application. Figure 6 shows how this structure looks like.

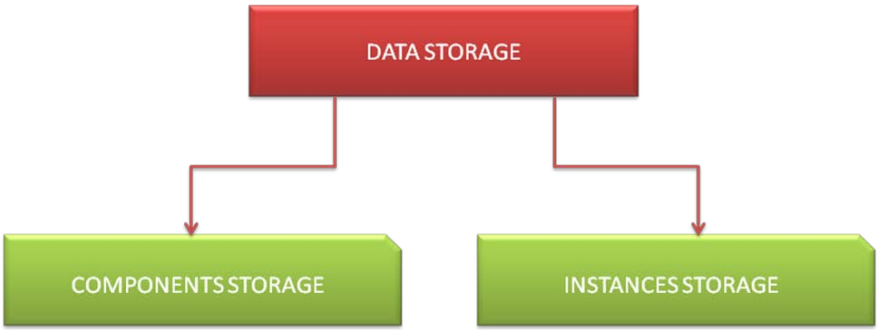

Figure 6. Basic data storage

The idea presented in Figure 6 is very simple: the data is actually compounded by two data storages; one stores the elements created to be used in the design; the other one stores the instances of these components. The advantage of this method is the possibility of reuse the components already created (for example, a typical umbilical cable has 6 to 10 identical hoses and the only difference is their positioning in the cable). The disadvantage is that the structure is more difficult to comprehend and maintain. This is the approach used in this software, due to the component reuse possibility and the flexibility desired.

Even though this structure seems simple, converting it into software is not a simple task. Figure 7 shows how the structure is implemented.

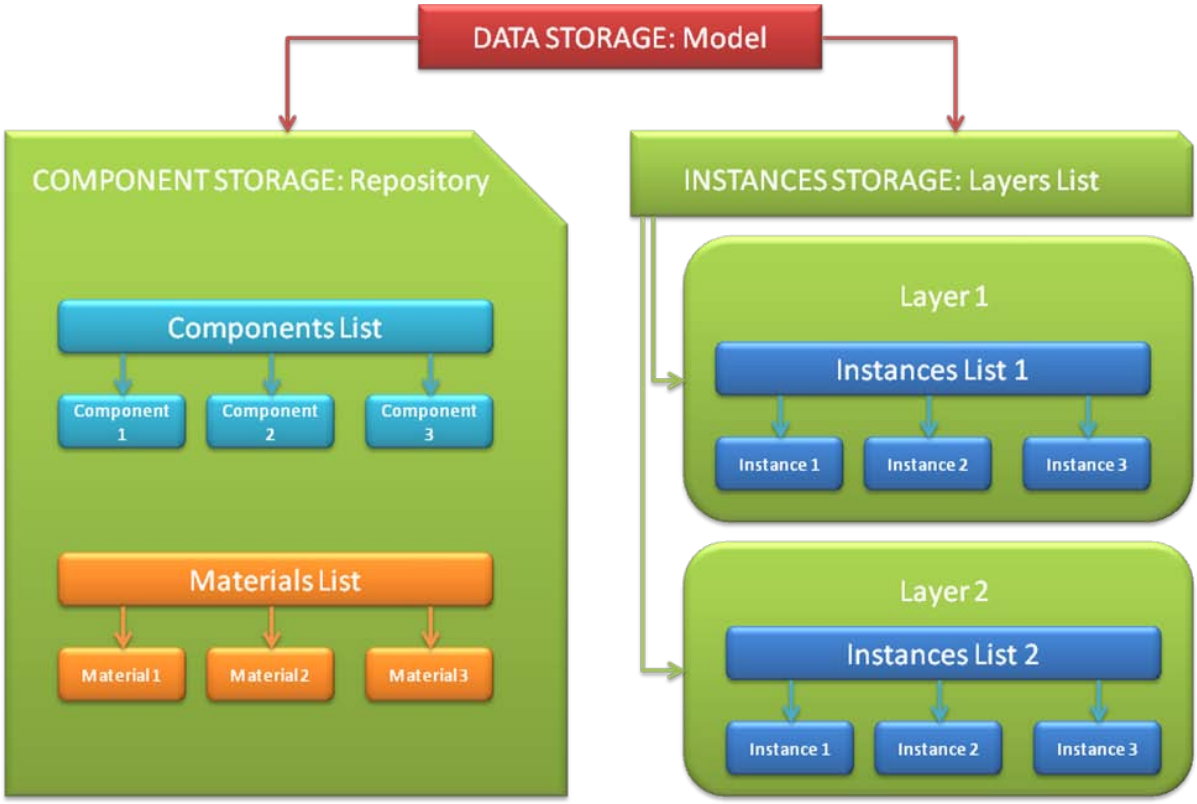

Figure 7. Detailed model structure

The greatest difficulty in dealing with this structure is that all elements are interrelated. For example, a component needs a material described in the material list; at the same time an instance depends on a certain component and, for the element visualization, some material properties (for example, color) must be retrieved when drawing an instance of a component. On the other hand, the most important feature of this structure is that it makes possible the insertion of a new type of element in a very short time and with almost no code alteration, except for the component class created. An example occurred recently, when steel tube umbilicals (STU) where designed for deep water wells: the reinforcement armor was replaced by using stainless steel hoses instead of polymeric ones. This new kind of element could be easily inserted in the code, due to this structure.

3. *CABLE MODELING PROCEDURES*

In this section, the usual proceedings of creating a cable using the software interface are described. Initially, the creation of the cable is started by defining a *Material* and a *Component*. Once the elements that will compound the cable are created, a *Layer* can be defined. A *Layer* is a recipient for instances, which are instances of the previously created components. A remark is, there must be at least one material to define a component and at least one component to define one instance, being possible to create more components after creating instances and layers.

The reusable characteristic of the model can be seen now: once one type of component is defined, many instances of this component can be created as wanted. As it was already said, this model provides a consistent way to insert multiple elements of the same *Component* with the least effort possible.

The required data for *Materials* are the Young modulus, the Poisson ratio and the density. Another property of material is *Label* and it is used for clarification. This last property allows components to be related with materials. The basic idea behind this is that, once a material is defined, a variety of components can be associated. For example,

Copyright © 2009 by ABCM

consider a steel hose. Consider also a helical wounded armor made of steel. A material (such as one named *Steel*) can be defined and can be associated to the armor and to the hose.

Components require one associated *Material* and a *Label* for identification. A variety of components can be created, such as:

- **Jacket:** jackets are used to represent thermoplastic sheaths and some elements that can be described as "equivalent cylinders". The thickness of the jacket is the required property.

- **Hose:** These represent the hoses in the cable. The thickness and inner radius of the hose are required as specific properties.

- **Armor:** these elements represent the helical wounded armor and their tendons. The total number of wires and their cross sectional information (if circular, radius; if rectangular, height and width) are required for this component.

- **Electric Cable:** They represent the electric cables. The number of inner wires and the thickness of the plastic sheath are required properties.

- **Optic Cable:** These represent the optic cables. The thickness of the plastic sheath is required.

- **Interlocked Carcass:** This one represents an interlocked carcass. This type of component is a complex one and it requires the description of the interlocked profile (which the user describes by given the profile parameters).

- **Pressure Armor:** This one represents the pressure armor component. This type of component has different standard types. It is possible to enter the parametric values of a certain type.

- **Filler:** this represent the fillers. This has only a *Label* for identification. There are variations of the filler (ring and cylinder) that require additional parameters.

Both *Materials* and *Components* are placed in the *Repository*, which hosts these elements as a library. To create an instance of a certain component, this object must exist in the repository.

The next step is to create the cable. To complete this next step, the *Model* structure provides a hierarchical approach: the cable is subdivided in several *Layers*, which serve as storages for instances. The layer inner radius and thickness of the layer must be inserted and then instances can be created inside this *Layer*.

Instances are created based on previous defined *Components*. To create an *Instance*, a *Label* must be defined, as well the base component and its position, given in polar coordinates (radius and angle in degrees). For elements such as jackets, the angle is not necessary, due to the fact that these elements and the cable are concentric.

Some highlights of these structures are:

- It is possible to modify groups of instances of the same component, just modifying de component itself.

- A generic distribution of instances is allowed, except when there is an instance overlay.

The approach of having a Repository and the Cable Tree described gives an easier way to manage components. It also makes easier to understand the cable structure.

4. *THE CAD INTERFACE*

In the previous sections, the cable structure was described and it showed how a cable is handled by the software. This section shows the CAD interface and the advantages of the approach become more evident. Figure 8 shows the complete software interface, which will be described briefly as follows.

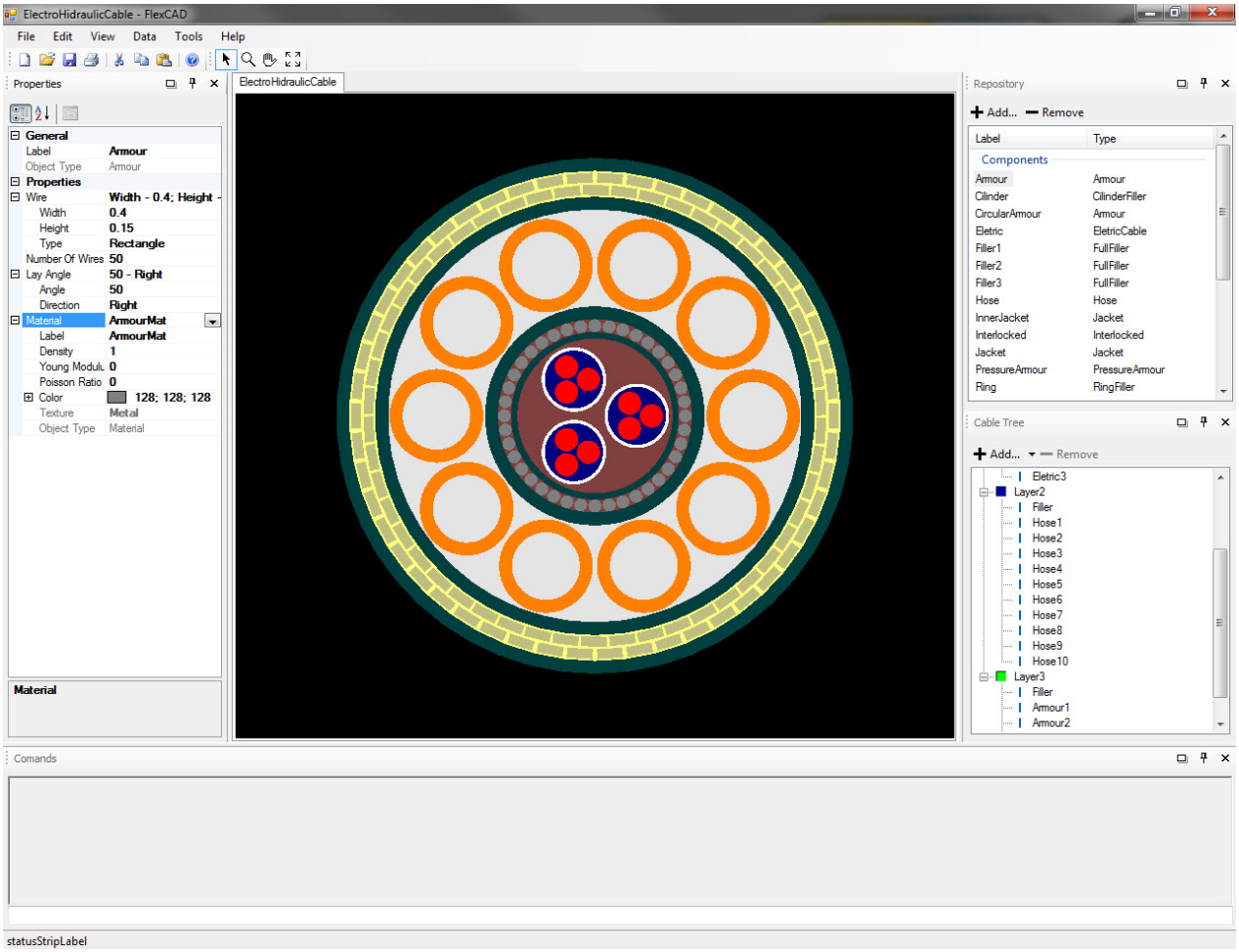

Figure 8. CAD Software main dialog with a designed cable

The interface is compound by several toolbars. The Properties toolbar (in Figure 8, the left side toolbar) shows the properties of the currently selected item. The Repository toolbar (the one at the top right in Figure 8) is responsible for the items storage in the repository; it has two buttons: one for adding a new item to the repository and one for removing an existing one. When clicking on the Add button, a window will pop up and it allows the input data of the component (or material) to be included in the repository (as shown in the Figure 9). Just above these two buttons, a list of the created components and materials are shown.

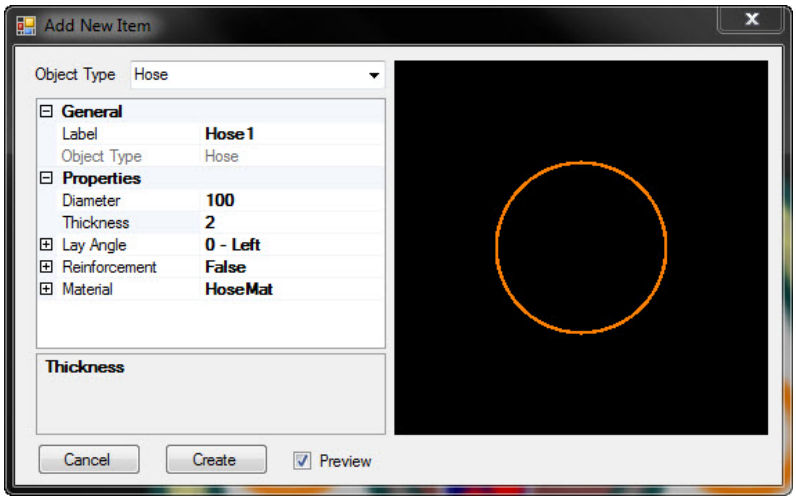

Figure 9. Add New Item dialog

Certain types of elements require additional data, for example, the interlocked carcass and the pressure barrier. This elements have a specific profile design and the software provides a parametric input. Figures 10 and 11 show the interlocked carcass profile and pressure barrier profile dialogs respectively.

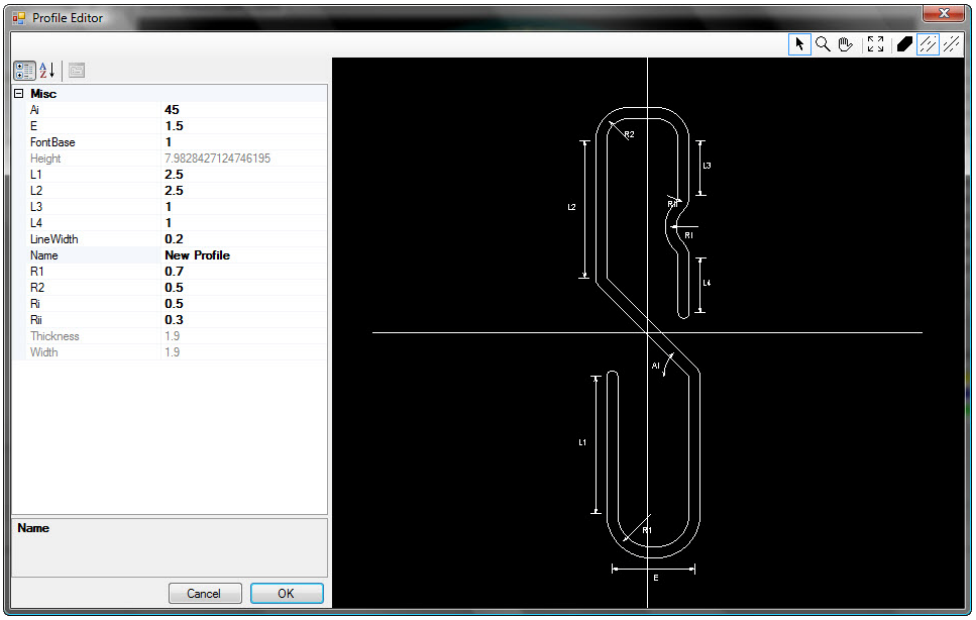

Figure 10. Interlocked Carcass Profile Editor

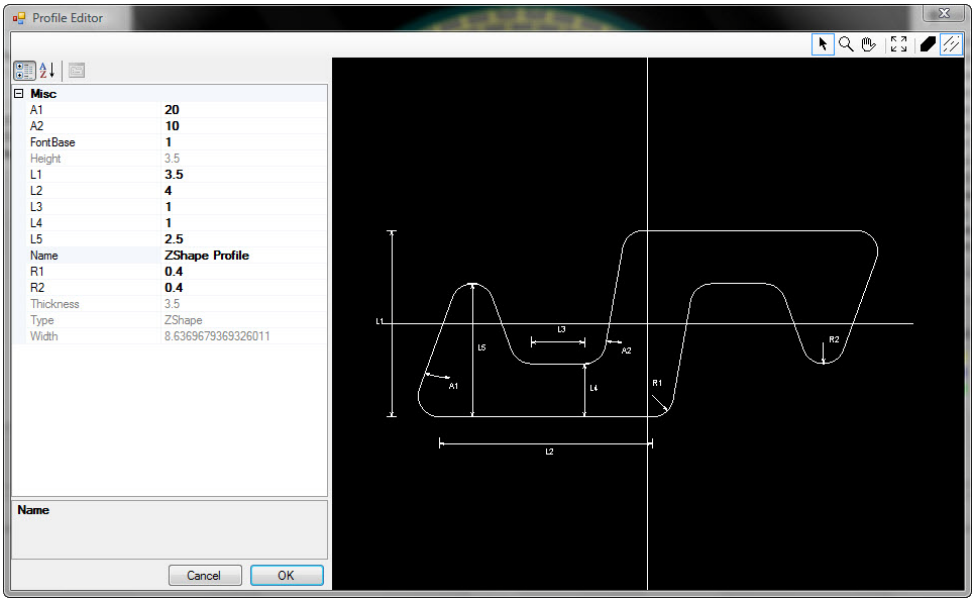

Figure 11. Pressure Armor Profile Editor

Another toolbar is the Cable Tree (bottom right one in Figure 8), in which the *Layer-Instance* structure is shown. The *Layers* are nodes of the tree and the *Instances* are leafs. There is another toolbar called Commands (located in the bottom of Figure 8), that provides command line features, recommended for advanced users. The remaining space is occupied by the tab page and this is the region where the cable is drawn. The program allows multiple cable opened simultaneously and, for each one, a new tab page is open.

The drawing consists in the layers' drawing overlaid by the instances' drawing. The user can select objects by clicking on it in the tab page, the repository or the cable tree (a note to be made: if a component is selected in repository and this component has several instances, all instances are selected in the drawing). Once an object is selected (instance, component, layer or material) the correspondent properties are shown in the Properties toolbar and

changes are reflected in all other toolbars. The software also supports drag and drop of components, generating an instance in the layer where the component has been dropped into and other functionalities such as cut, copy and paste and also the possibility of printing both cable drawing and model information.

5. EXAMPLE CABLE

To exemplify the features of the software, a cable was created. Figure 12 shows the cross-section of this cable. The modeled cable can be viewed in Figure 8, in the tab page (center of the window).

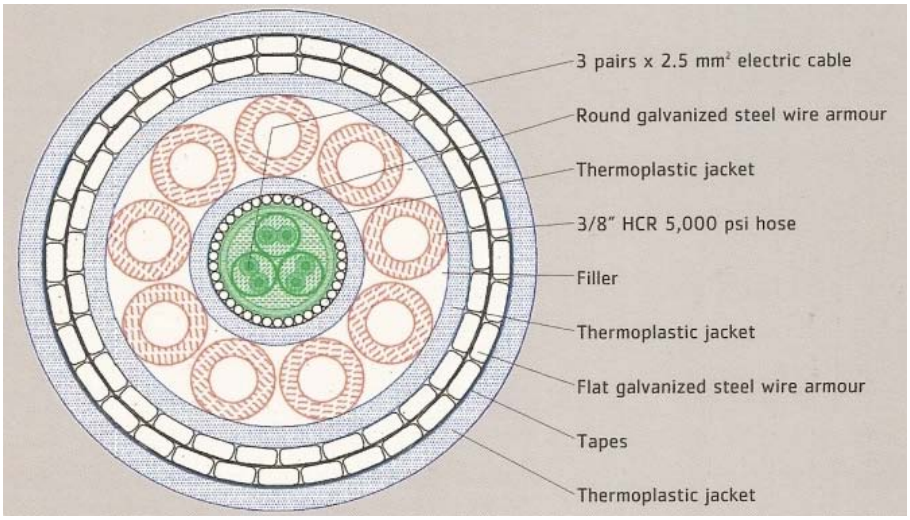

Figure 12. Electro-hydraulic Umbilical cable cross-section (extracted from Prysmian Catalog).

Both repository and cable tree are shown in detail in the Figure 13a and Figure 13b, respectively.

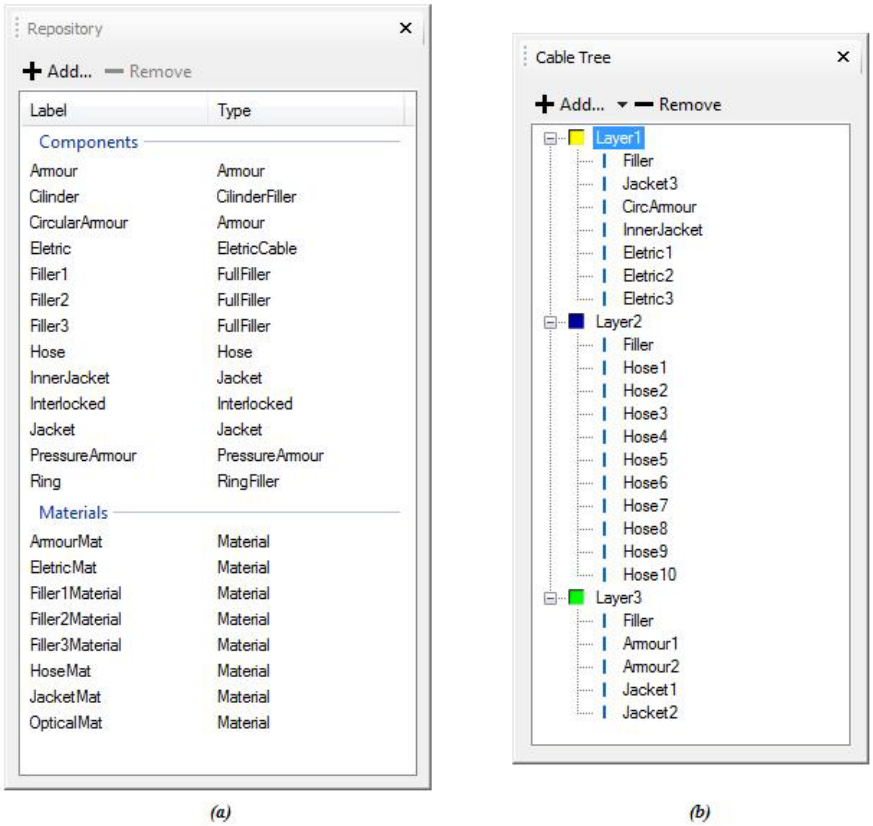

Figure 13. The modeled cable toolbars: Repository (a) Cable Tree (b)

5. CONCLUSIONS

The CAD modeling of a cable in a generic and flexible way has been made. It has been shown that the approach is suitable for implementing in a two-dimensional CAD. The paper dealt with a complex structure which will be used in future to describe the cable in the three-dimensional coordinates and also to determine displacements and stresses calculations. This stage successfully generated a solid foundation for the upcoming modules.

To prove the effectiveness of the modeling software, a typical cable cross section has been created and the compound elements has been shown. Even though the example used through this papers was an umbilical, it is suitable for modeling flexible pipes (and that explains the presence of the interlocked carcass in the possible components).

The obvious and intended sequence of the project is the development of a consistent model for calculating the displacements and stresses in the structure. Another intended module is the three dimensional one, for post-processing the results obtained using the solver. This module includes a visualization tool for preview a previously created cable, making possible an evaluation of the final elements' distribution in the cable before the project is finished.

6. ACKNOWLEDGEMENTS

First author acknowledges FAPESP for funding his PhD studies under grant number 2006/06281-8. Second author acknowledges FAPESP for funding his undergraduate research under grant number 2007/03698-8.

7. REFERENCES

PRYSMIAN CATALOG. Umbilical Thermoplastic and Steel Tube.
WELLSTREAM CATALOG. Manufactures of Spoolable Pipeline Solutions.
API RP 17B Standards.
SOMMERVILLE, IAN. Engenharia de Software. 6ª Edição. São Paulo : Addison Wesley, 2003.
DALE, NELL. C++ Plus Data Structures. Third Edition. Jones and Bartlett Publichers, 2003.
MARSHALL, DONIS. Programming Microsoft Visual C# 2008: The Language. Microsoft Press, 2008.

8. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.