A Computer Aided Process Engineering Tool for the Study of Organizational Processes

Brenda L. Flores Ríos and Ana I. Martínez García

Departamento de Ciencias de la Computación. CICESE
Carretera Tijuana-Ensenada Km. 107 Ensenada, B.C. México
{bflores,martinea}@cicese.mx

Abstract. There exist many kinds of tools for the studying of organizational processes (i.e. software process), but they do not follow the process engineering life cycle. Thus, Computer Aided Process Engineering (CAPE) is a term introduced to define and group the tools, which follow all the phases of a process engineering study. The importance of CAPE type of tools is discussed and a prototype of this kind named CAPETool is described. CAPETool follows and provides support to all the phases (capture, modeling, evaluation, analysis and redesign through IT) using a standard methodology. The tool uses a definition of a base model for a descriptive and formal representation of the processes; this makes them reusable and platform independent. The processes are modeled in a process modeling structured technique named RAD (based on roles, their interactions and activities), which provides most of the information needed in a process model representation. The RADs base model is defined in eXtensible Markup Language (XML) and double linked lists are used for keeping the RADs' roles interactions. To have reusable models is a very important feature, as we can generate from static process models, such as RADs, their corresponding dynamic models (i.e. enactment, simulation, etc.)

1. Introduction

Organizational processes (software development, project management, business, etc.) can be defined as a group of logically related tasks that use the resources of an organization to get well defined results in support of its objectives [1]. The structured definition of processes provides advantages in its description, analysis modeling and support. Some of these advantages are to be able to represent the *agents* (a person, group or a system) that perform a process element, the *activities* that are carried out, their flow and interactions between roles (group of activities that together achieve an aim), the *units of information* produced and manipulated by the process and mainly to be able to determine the possible areas for improvement, if they exist. *Process Engineering* is a field that can be used to study organizational processes. It is defined by Kawalek [2] as the collection of techniques for the analysis, design and evolution of organizational processes based on the use of *Process Modeling*, which is defined as the generic name given to the collection of techniques for modeling systems behavior. Process Modeling emerged as a result of research focused on the study of the software development life cycle and the software process itself. Its aim is to provide support in reaching the objectives of an organization by providing the means for representing, analyzing and understanding their current processes in order to find out possible areas for improvement and determine the process support needed.

In the area of Software Engineering, improving the quality and productivity of software development is a key point. Hence, theories, methods and tools have been introduced and developed to support and improve the process. Computer Aided Software Engineering (CASE) tools are one of the technologies introduced to improve the systems' production time, as they guide the *Software Engineer* through the life cycle of the software process facilitating the development of its phases.

Nowadays, there is also a great variety of tools for Process Engineering [3], some of them capture the description of the process using graphical representations, others use more dynamic Information Technology (IT) to show the behavior of a process and generate some reports for the study in progress. However, in the literature it is not possible to find Process Engineering tools that, as the CASE tools in Software Engineering, integrate all the methodological steps needed to support all the phases in a study of an organizational process using Process Engineering [3]. From these facts, emerges the need of these kinds of tools with an integrated environment that contemplates both the static and dynamic view and analysis of the process.

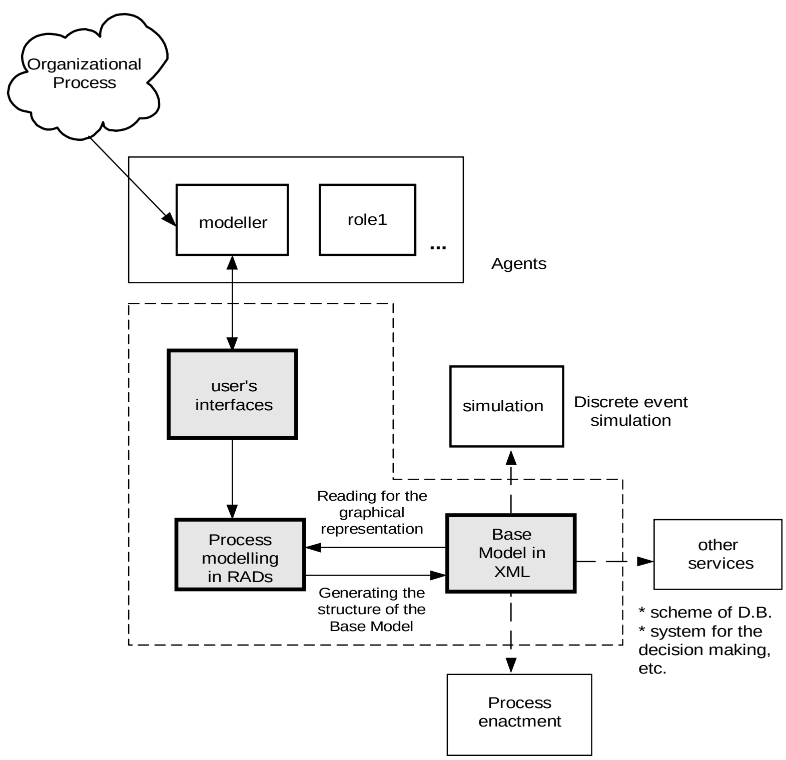

Fig. 1. General design of the tool with the base model that is used in the processes' representation.

In this work, the Computer Aided Process Engineering (CAPE) term is introduced to define the tools that provide support to all the steps involved in a complete Process Engineering study of an organizational process. Also a prototype of these kinds of tools, named CAPETool, is illustrated. CAPETool provides support through the phases of capture, modeling, evaluation, analysis, redesign of processes and the implementation of process support with IT. Thus, a CAPE tool can be defined as that which gives support to the study of processes, by providing the means for the analysis and visualization of the different perspectives (functional, informational, behavioral and organizational [4]) that characterize a process. The correct use of these types of tools, as a support to each step of a process study, will provide to the members of the engineering team involved in the study means for a better communication and support through the analysis and understanding of the organizational processes.

An important point in the design of CAPETool, as it is shown in Fig. 1, is the reusability of the static process models' representation, that can be used as a basis for the generation of dynamic models (i.e. simulation models, enactment, etc.) which will provide support to the process using IT. Inside the dotted box in Figure 1, the base model represents, in terms of a process, what is intended to model, analyze or coordinate within the organizations, in a descriptive form. In this work the eXtensible Markup Language (XML) [5] is used for the formal representation of the process models. The *Process Engineer* could use the base model of the process to generate the graphical representations of Role Activity Diagrams (RADs), simulations, coordination systems (enactment), etc.

In the following section the CAPETool prototype requirements are briefly described together with the results obtained of using the process base model in XML with the characteristics of an organizational process.

2. CAPETool

Nowadays organizations make an appropriate use of the power of IT to redesign their processes [7]. Thus, doing that an organization improves its performance and competitivity keeping the balance and a level of coordination between its main components: staff (social aspect), methodology (focus on processes and software) and tools (technical aspect). We present the development of a support tool for the study of organizational processes, which provides some means, which contribute in the improvement of processes.

The process group at CICESE has named the computer aided tool that supports all the phases of a Process Engineering study CAPETool. According to this, CAPETool must be developed under the following requirements:

(a) The tool must provide support for the process capture, description and analysis. Based on the capture and description of an organization's process it must be possible to perform an analysis to determine key areas for improvement or redesign.

(b) It must provide the means to document and analyze the impact of socio-technical changes to the process using process models simulation.

(c) It must facilitate the following and managing of the process engineering study of an organization. CAPETool must be flexible enough to incorporate or interface tools or add modules according to the needs of a particular organization.
(d) The tool must provide support to a variety of diagrammatic techniques for the modeling of processes. The process model must be represented in a base model, reusable and independent of the platform.
(e) It must have and support a process engineering methodological structure, to work as a guide in the study of organizational processes.

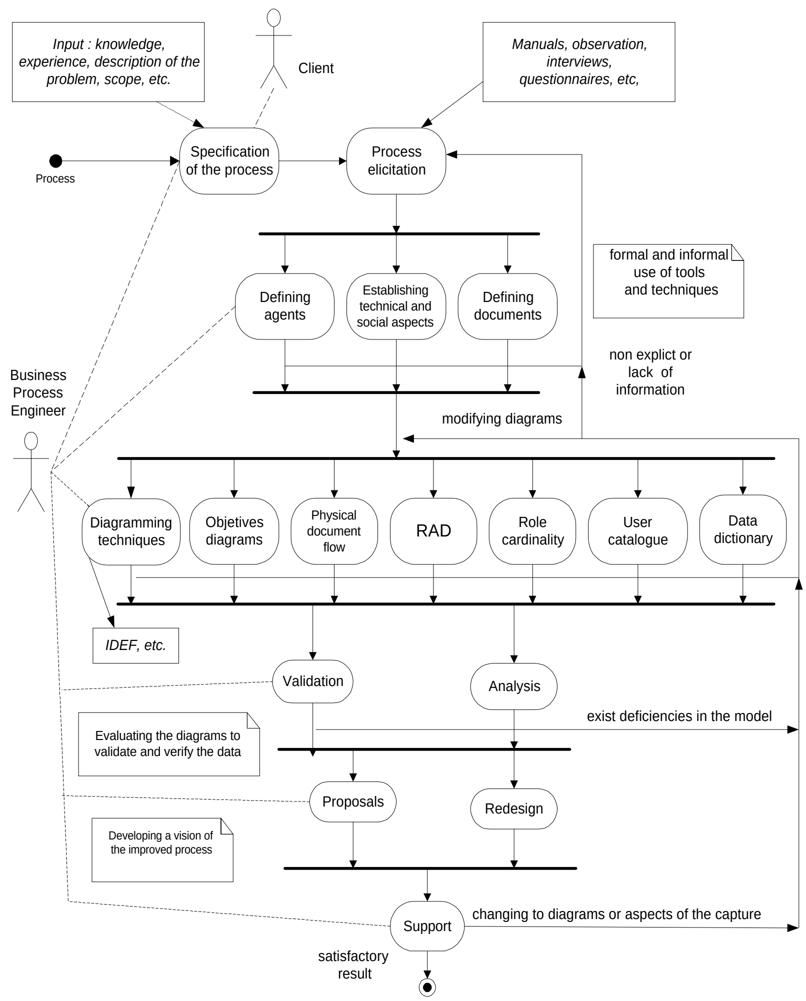

Fig. 2. Use cases of CAPETool showing its structure and relations. The use cases are oriented to the activities that the Process Engineer and the Client follow when performing an organizational process study.

The use case diagram is oriented to the activities that the Process Engineer and the Client have to follow to perform a process study. Fig. 2 shows a use case diagram of CAPETool. The key activities are: the capture of the socio-technical issues, documents and agents; the modeling of activities using a variety of diagrammatic techniques (IDEF [6], RADs [10], physical document flow diagrams, etc.) supported by the users catalog and data dictionary; the process models' evaluation to verify and validate their content (information); static and dynamic process analysis; redesign (a view of the process TO-BE); supporting the process with IT (simulation [7], Data Bases, Web, etc.) and finally the generation of reports with the general and main features of the process.
The focus of this work is on the representation of:

1. The generic base of the prototype of CAPETool.
2. The base model of the process represented in XML.
3. The mapping of a RAD model to a data structure to be represented in a base model that is reusable and platform independent.

In the next section we provide a more detailed explanation of these three points.

3. Process Base Model

The XML document defines the processes in terms of labels based on the concept of roles, activities and interactions (RADs). A parser implemented in Java can perform the reading of the process base model from the XML document. The parser translates the models' behavior in terms of the reading of data contained in the process base model structure (using labels that we defined), and then the information is introduced into CAPETool to perform the graphical representation of the model (in RADs). A simple example of a process base model of a real case study in the health sector (family health care process) is shown in Fig. 3, illustrating some of the XML labels' definition, proposed to model a process in RADs and to get the information needed from them.

```
<?xml version="1.0" ?>
<!DOCTYPE rol (View Source for full doctype...)>
<rad>
 <role name="Waiting turn in reception">
   <agent who="a">Pacient (eventual, with an appointment or
exp)</agent>
   <act id="a1">
       <nom> Handing in the appointment card</nom>
       <coordenate> <x>250</x> <y>100</y> </coordenate>
       <interaction which="1" type="E">b </interaction>
   </act>
       ...
 </role>
...
</rad>
```

Fig. 3. Definition of a XML document of a process base model that shows an example of how to represent a role (in RADs) and its activities.

XML makes possible to define our own Document Type Definition (DTD) by defining our labels and establishing the structure of the document needed. Hence, we defined a DTD for the process base model. In this DTD we specified the labels according to the elements that characterize a process modeled with RADs (Fig 3). The DTD is a file that contains the formal definition of a general RAD model; this defines the labels' names that can be used, where can they appear and how they relate to each other.

3.1 Process Modeling with Role Activity Diagrams (RADs)

RAD is a structured technique, where the description of the processes is given in terms of activities, roles and their interactions [10]. This technique represents most of the main characteristics of a process (objectives, roles, decisions, etc.)[6,9]. Thus, we selected it as the base modeling static technique to be used for the generation of the dynamic models. The graphical notation and a brief description of the RAD elements are shown in Fig. 4.

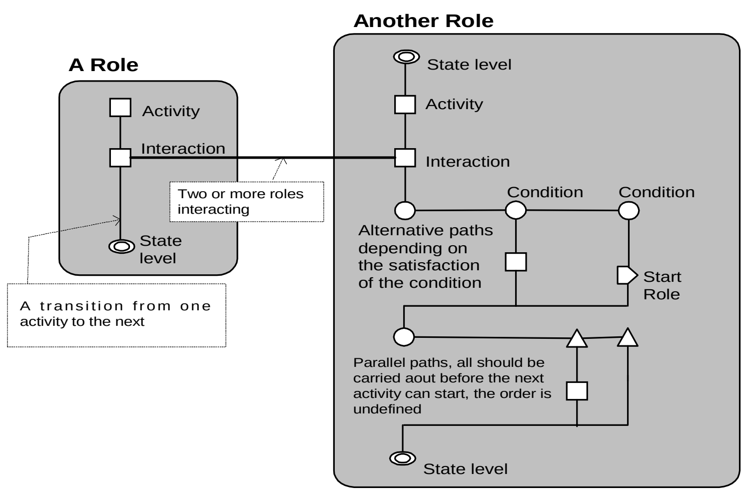

Fig. 4. A brief description of the RAD elements (activities, interactions, state labels, alternative and parallel paths.)

Roles are represented by a rectangle with a label at the top, specifying the role's name and the agent responsible to perform it. Inside the role the activities are drawn as small squares with its description at the right, the interactions between roles are represented by bold lines, which connect from the role activity that started the interaction to the activity of the role to which it is communicating. The vertical lines that connect the RAD elements are called state lines, which show the transition (states) between the elements. In the next section, we explain the details of the storage and representation of the process base model elements (RAD), extracted from a XML document, as a data structure in Java

3.2 Logical Representation of the Process Base Model to a Data Structure

Double linked lists are the data structures used in the storage of the information read from the process base model represented in a XML document. These structures are read into CAPETool, and then the RAD model is constructed graphically and the main features of the process can be observed [11].
The double linked lists are easy to use. They are very flexible with the operations of insertion, deletion and access to the elements defined for the process base model and in the representation of the variety of diagrammatic techniques.

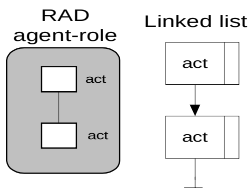

Fig. 5. A RAD with two activities and its mapping to a linked list. The two activities are represented as two nodes linked by a pointer.

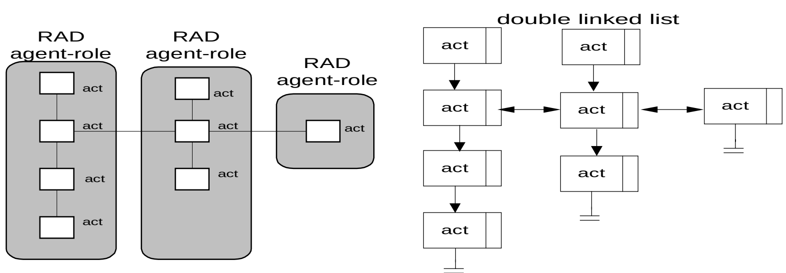

Fig. 6a. A RAD where a role has an activity linked to other roles and activities within itself.

Fig. 6b. Representation of the RAD in Fig. 5a as a double linked list.

Let us consider the case of a role with two activities (Fig. 5). This could be stored in a list with two elements. However, it must be possible to perform more operations than those allowed by a simple list. Thus, in a more complex case, such as a RAD with an activity with four links (Fig. 6a), it is necessary to have the reference to all the linked elements, as it is illustrated in Fig. 6b. Hence, a double linked list provides all the possible operations needed in the RADs representation and manipulation in the CAPETool environment.

This small example illustrates the complexity of the data structures used in the process models' representation using RADs. CAPETool is a generic prototype that is intended to work with a great variety of process models.

4. CAPETool Implementation

The implementation of the prototype was performed in Java [12]. In the development of the Graphical User Interface (GUI) we used the Abstract Window Toolkit (*AWT*) and the elements of *Swing* in Java2. Therefore the implementation of CAPETool has several forms to input (through menus, bottons, etc) and output information (by writing reports with the main features of the process base model, displaying the diagrams for the modeling techniques, etc.). The prototype is platform independent and can be executed in any operating system that supports Java (i.e. *Windows* and *Linux)*. Fig. 7 shows the main screen of CAPETool, where the options are the corresponding to those established in the definition and requirements of tools of CAPE type. The options represent a requirement (capture, modeling, evaluation, analysis, redesign and support) as described in the use case diagram (Fig. 2).

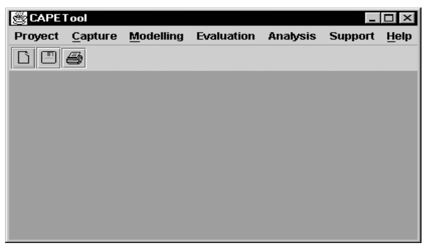

Fig. 7. Main screen of CAPETool. The menu offers the options established in the use case diagram (Fig. 2).

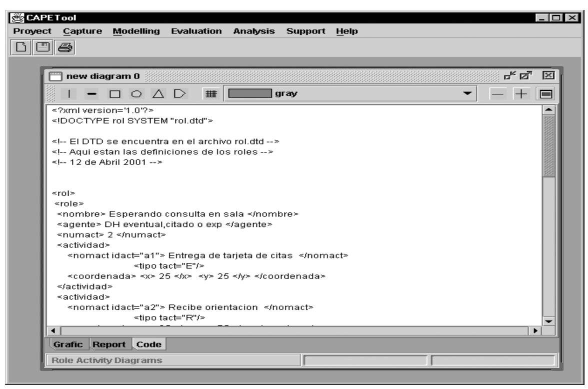

Fig. 8. RAD editor. Representation of a RAD after reading the process base model from the XML document.

At this stage one of the main features of the tool is the modeling and representation of the process using RADs; thus, the tool has integrated a RAD editor (Fig. 8), which is used to read the process base model. Its functionality has been tested with a real case process from a case study in the health sector (Fig. 3) and some other examples.

The editor presents information about the process model through three layers: *Code*. - Shows the structure of the process base model. *Graphic- Performs* the reading of the process base model and then draws the RAD (Fig. 9). *Report*. - One can have the process model reports (Fig 10) and then copy and paste it into any word processor.

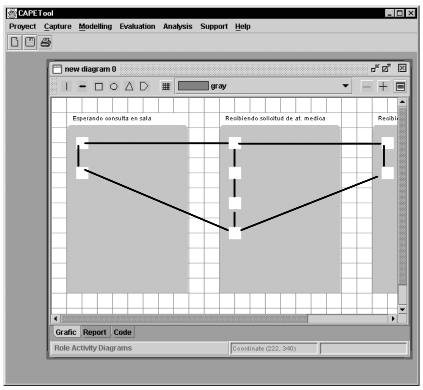

Fig. 9. The RAD process model generated from the process base model defined in a XML document built in the RAD editor.

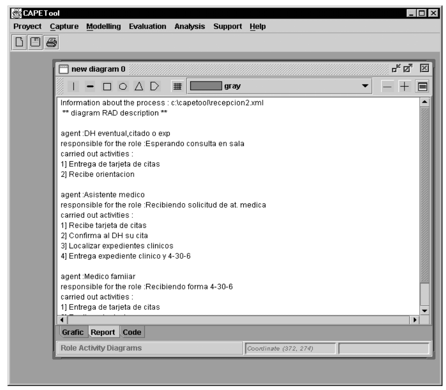

Fig. 10. Report generated with the features of the process based on the process model on RADs.

The report generated (Fig. 10) with the process main features, can be taken to any text editor or word processor (i.e. Word for *Windows* or Staroffice for *Linux*). Therefore, the generated information could be used not only within CAPETool, but also in any other tool to generate documents with information and formats that an organization considers relevant.

Through the implementation of the CAPETool prototype, we present the platform independent process base model structure in XML that is proposed and how the graphical RAD is generated from this. However we recognize that mapping the structure of all possible RADs and the representation of their graphical characteristics requires a complex DTD. The base model is still under development adding some extra features of RADs (i.e. recurrence of some elements such as alternative or parallel paths, within themselves, etc.).

5. Conclusions

In this work we have introduced CAPE as a term to identify the different computer aided tools that follow the phases of process engineering. We emphasized the importance of having a methodological approach to the study of organizational processes based on process improvement.

We also remarked the relevance of using an integrated tool that supports a process study by means of process engineering using a CAPE approach such as the prototype tool that we illustrate CAPETool. With the CAPE type of tools, support to all the steps on a process engineering study is provided without the need to change and use different environment tools that do not facilitate the reusability of models (i.e. to generate enactment or simulation models).

We also presented important implementation features of CAPETool that facilitate the reusability of process models. Features such as the RAD process base model in XML and the simultaneous generation of a report with the main aspects of the process modeled in RADs.

The example use through the illustration of CAPETool is a case study developed in the health sector. In particular the family health care process of the Family Medical Unit 32 in Ensenada, Baja California, Mexico.

Our main contributions with this work are: the CAPE approach to building tools for supporting the study of organizational processes, the definition of a generic process base model (in XML) that permits the reusability and portability of process models and a CAPETool, which uses a process base model for the formal and graphical representation of processes.

Our aim is to continue developing the process base model of CAPETool and the tool itself adding the features for the dynamic analysis of processes and continue using it in real case studies to evaluate its usability and the relevance of these types of tools.

References

1. Gladwin, B. y Tummy, K., Modelling Business Processes with Simulation Tools, Proceedings of the 1994 Winter Simulation Conference, 114-121. 1994.
2. Kawalek Peter. A Method for Designing the Software Support for Coordination. Ph.D. thesis. University of Manchester. England. 1997.
3. Kettinger William, Teng James T. and Guha Subashish. Business Process Change: A Study of Methodologies, Techniques and Tools. Volume 21. Number 1. USA. 1997.
4. Curtis B., Kellner M and Over J. Process Modelling. Communications of the ACM. Vol. 35. No. 9. 75-89. 1992.
5. McLaughlin, Brett. Java and XML. O’Reilly & Associates. Inc. USA. 2000.
6. Ould, Martyn A. Business Processes: Modelling and Analysis for Reengineering and Improvement”. John Wiley and Sons Inc. England. 1995.
7. Davenport T. H. Process Innovation: Reengineering Work Through Information Technology. Harvard Business Press, Boston. 1993.
8. Hunt V. Daniel. Process Mapping. How to reengineer your business processes. John Wiley and Sons Inc. USA. 1996.
9. McLaughlin, Brett. Java and XML. O’Reilly & Associates. Inc. USA. 2000.
10. Miers, D. Use of Tools and Technology Within a BPR Initiative. In Business Processes Reengineering. 1996 edited by Coulson-Thomas Collin, Kogan.142-165. 1996.
11. Weiss Mark A. Estructuras de datos en Java. Addison Wesley. España. 2000
12. Geary David M. Graphic Java 2. Sun Microsystems Press. USA. 1999.

Acknowledgments

This work was partially supported by CONACYT under grant I33046-A and scholarship 143001 provided to the first author. We would also like to thank Dr. Fernando Rojas from CCMC-UNAM for helpful discussions and comments. We thank the staff of the Family Medical Unit 32 in Ensenada, Baja California, Mexico for their participation during the development of the case study.