A Software Development Tool for Evolutionary Prototyping of Information Systems[1]

P. Paderewski, J. Parets-Llorca, A. Maya, M.J. Rodríguez, G. Sánchez, J. Torres, M.V. Hurtado
GEDES Research Group[2]
Departamento de Lenguajes y Sistemas Informáticos
Granada University
ETS Ingeniería Informática.
Avda. Andalucía, 38. 18071 –Granada
SPAIN

Abstract: Software Systems are executable models of complex Information Systems, which emerge from an intelligent process of interaction between the modeler and the system which he recognizes as being modeled. We are developing an approach to modeling Information Systems using evolutionary Software Systems because both should be able to evolve in unison. We have defined the method MEDES and are developing the tool HEDES, a prototype in which a Modeling System creates the structure of a Software System and makes changes to it (evolution) while the Software System is working. The principal features and the architecture of MEDES and HEDES will be presented in this paper.

Key-Words: Software System, Information System, Evolution, Prototyping, Software Development tool.
CSCC'99 Proceedings, pp. 7521-7526

1 Introduction

Software Systems are an important part of Information Systems, which try to reflect the activity and decision in organizations using processable data. However, most of these Software Systems are designed in such way, that adoptions to the new informational needs of the organizations can not be made.

As for as we are concerned, Software Systems are executable models of complex information systems which emerge from an intelligent process of interaction between the modeler and the system which he recognizes as being modeled.

This point of view requires that Software Systems and Information Systems should be able to evolve in unison and that this evolving process could be conceived and represented from the Development System point of view. Our approach is not to design Software Systems which can evolve by themselves recognizing the informational needs of the organization. We still consider that intelligence rests with the modeler and with the modeling process itself. Therefore our interest is centered around the research of this intelligent behavior, i.e. the evolution of the representations which the modeler makes for himself of the Information System that he is modeling.

This approach is gathered in our software development method (MEDES). A method based on the Theory of the General System [Le Moigne, 1977-90] and very different from the commercial ones, because its aim is to research the epistemological foundations of the evolution process by trying to incorporate models of

[1] This research is supported by a project financed by Spanish CICYT (TIC97-0593-C05-04) which is a subproject of the MENHIR project (TIC97-0593-C05).

[2] The author is the research header of GEDES (Group on Specification, Development and Evolution of Software). This paper presents the work which is being developed by the following researchers: Ana Anaya, Maria Visitación Hurtado, Patricia Paderewski, José Parets, Maria José Rodríguez, Germán Sánchez, Jesús Torres.

evolution from different disciplines.

This conception of the modeling process of Information Systems and its evolution through the modeling of Software Systems is being implemented in a software development tool which we call HEDES. The aim of this paper is to present the main features of the tool and the challenges that it will have to solve during its development.

The paper is organized as follows. In the first paragraph we will briefly summarize our approach in modeling Information Systems using evolutionary Software Systems. Secondly, we will present the main features of the method and tool, which we are developing. Subsequently, the architecture of the tool will be explained. Finally, we will show the main trends of research in the project.

2 MEDES: Evolutionary Modeling of Information Systems

An explanation of the objectives, hypothesis, foundations and characteristics of MEDES can be found in *Parets-Llorca [1995]*. A brief summary of the approach is presented in *Parets et al. [1994]* and *Parets &Torres [1996B]*.

2.1 Modeling system and modeled system

Systems, and therefore Software Systems, are artificial constructions made by a modeling system from an observation-conception of real phenomena.

The Information System (IS) is a complex structure which reflects the activities of the organization in which it is immersed and carries out the treatment of information with in the organization. The IS will serve as a support for taking decisions in the organization (some authors talk about Strategic Information Systems).

Software Systems (SS) are a model of a part of the Information System of which they form a subpart. Their incorporation into the IS will always produce changes which will oblige changes in the SS, the IS and, probably, the organization.

Our approach implies an evolutionary relationship between the Modeling System (MS) and the modeled system (the IS): the SS should evolve through the activity of the MS which adapts it to the new needs. For the moment we prefer not to say that SSs are intelligent because intelligence, apart from a representation of themselves, implies the capability of changing this representation in accordance with new finalities and changes in the environment. We consider that intelligence, the capacity of self-finalization, resides in the MS.

2.2 Function, structure and evolution

Traditionally the modeling process of Information and Software Systems implies the capture of two aspects of its interaction: function and structure. In fact, the majority of Information and Software Systems development methods and tools only emphasize these two points of view. Obviously both aspects are important, but they lose dynamical capacities without the possibilities of structural and functional changes through time.

In MEDES we represent Software Systems using the concepts of the Theory of the General System: processors, environments, systems, actions, events and decision. We begin with a basic structure of a software system integrated by these elements and by a historical representation of the functioning and of the structure of the system.

The **function** can be conceived using the concepts of action, event and state, assuming the temporal discontinuity with which the Modeling System recognizes the actions. An action is the possibility of carrying out a transformation. The consciousness of the system action, which the MS has, resides in the possibility of identifying a change in its state (production of an event). We conceive an *event* as information which represents the actions that have been carried out in the system.

Structurally a Software System will be constituted by autonomous processors capable of carrying out actions, an interface which allows the system to behave as a processor, and a memory of events (the history of the system). To allow flexibility in the relationships between processors and systems, MEDES includes hierarchical integration and the possibility of direct interaction between them, considering that each processor can be substituted by a more complex system. Because the structure of a system is a form stable enough to be recognized by the modeler, *[Le Moigne, 1977-90]* we can define the structure of a Software System as 'the patterns of behavior and decision in a moment of time, patterns stable enough to be modeled and to work during any given period of time'.

In this context, the notion of System History represents the trajectory of the functional and structural changes of the Software System through time adopting the view of the General System Theory.

The distinction between structure and function and their event representation allows us to introduce the concepts of Functional History of the System (SFH) and Structural History of the System (SSH). SFH stores the result of the actions that the processor/system carries out through time. SSH memorizes the information related to the changes in the structure. Both memories store information as events.

The **capacity of evolution**[3] is achieved by introducing the notions of evolution interface and Genetic Subsystem (GS) The first allows a Modeling System to take some actions in order to modify the structure of processors and the capacities of action and decision of the System which it is modeling. The second allows us to model evolution by processing the actions of the evolution interface. To model the work of the Modeling System we introduce the concept of METASYSTEM: a software system, which allows us to interact with the developed systems. In order to allow for evolution, the MetaSystem will be responsible for the actions which imply changes of structure in a system, storing the resulting events in the SSH. The interactions between both systems, the Software System and its MetaSystem, can be seen in Fig.1.

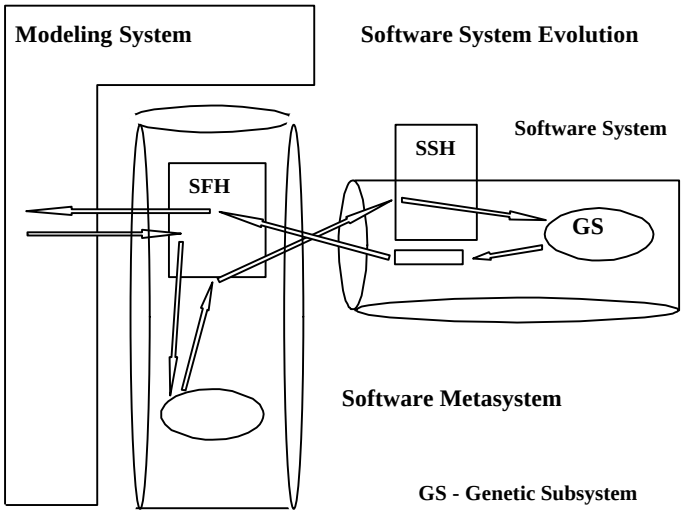

Fig.1

3 MEDES AND HEDES: features

In the previous paragraph the most important concepts used in MEDES have been explained. These concepts need a further refinement in order to have operational representations which, finally, could be implemented. In this paragraph we develop the previous notions using more concrete representations. These representations have been obtained in a process of interaction between the methodological level (MEDES) and the practical level (HEDES). This explains that the concepts are valid at both levels.

[3]To understand the notion of evolution Le Moigne introduces the concepts of CINEMATICS (functional changes in a system) and DYNAMICS (structural changes which allows new forms of functionality). We use the term dynamics in this sense.

3.1 System and processors

In MEDES and HEDES, the distinction between the notion of System and processor is only a question of level of abstraction. A System has an external environment and a processor works inside a System. Fig.2 shows the basic scheme of functioning of one of these processors. Because a System could be included inside another System, it can be used as processor.

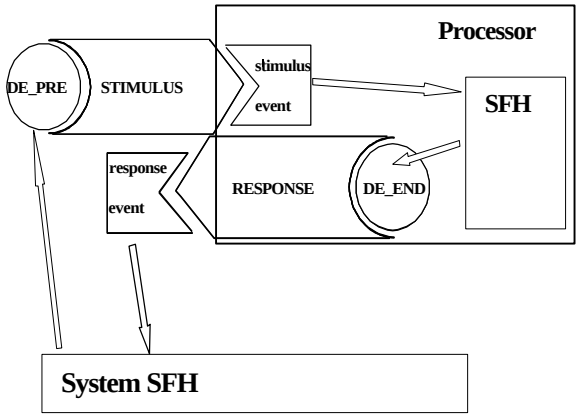

Fig.2

A pattern of activation of the actions of the processors based on a biological Stimulus-Elaboration-Response model is used. An important part of this functionality is the decision which we represent as PRE, DURING and POST conditions on the realization of the actions. This form of decision allows the processors to activate themselves through stimuli memorized in the SFH, being monitored during the execution of their actions.

3.2 Histories and time representation

The events stored in the SFH are related to the System functioning and represent actions carried out in the system (symbolization of the realization of an action, information about an external stimulus, which affects an action, etc.). On the other hand, the events stored in the SSH are related to actions carried out over the system structure: creation or removal of a processor, addition of preconditions to an action, and changes of the preconditions, etc.

Given that events occur through time, each event should be allocated in a temporal structure. One of the most interesting results of our research is that functional and structural events can be represented and managed in a homogeneous way using the same temporal structure and the same temporal inference machine. In fact we introduce a representation of time and we use predicate temporal logic *[Gabbay&Reynolds, 1995]* to model queries over the history of events *[Anaya et al.1996, 1997]*

3.3 Composability through collaboration

So far we have considered a HIERARCHICAL structure of a system, i.e. a processor is *a part* of a Software System and only has meaning inside a system. Yet we can conceive a COLLABORATIVE structure where a processor has a 'working for' relationship with the system. Because this conception of software systems allows autonomy and the possibility of a processor/systems working for more than one system (simultaneously), we introduce direct communication between the processor interfaces.

The use of collaborative actions is a powerful mechanism in modeling evolution because a software system can be derived from one processor using reproduction and collaboration. In addition it allows us to model autonomy and complex relationships between the Software System and its processors, its environment and its metasystem

4 HEDES: architectural design

The development of HEDES stems from the biological analogy: reproduction-differentiation-maturation in a cyclic and recursive way. We are developing the tool under an object-oriented language (Smalltalk) using and interpreter of the SDL (System Description Language, *in Parets&Torres [1996A]*) and a minimal software structure (the CORE). Fig.3 offers the approach. HEDES is the first and primitive MetaSystem with an interpreter of the SDL and an elemental system (ES) with the core capabilities. Using the SDL this elemental system can be evolved in order to obtain other Systems and MetaSystems.

From the point of view of the functionality of the tool we distinguish three types of features which are being implemented in different steps:

1) HEDES CORE

Includes the implementation of the memories and the functionality of the processors using events and conditions expressed in First Order Temporal Logic.

The Genetic subsystem which allows the basic changes in a system is also implemented.

A refinement of this core involves concurrency and collaboration which are non being implemented.

2) SYSTEM REESTRUCTURATION

Implies the possibility of advanced changes in the structure of a defined system and includes different possibilities (reclassification of processors, change propagation, event structure change, self-evolution and learning) that we are studying at a theoretical level (see next paragraph).

3) SYSTEM OPTIMIZATION

The growth of the histories of systems and processors implies a loss of functional efficiency. This historical representation is useful in modeling the evolution process but more efficient representations should be used. In order to manage this problem, we research the possibilities of transforming history in attributes by means of an operational representation of the activity based in CPN (Colored Petri Nets), Relational Data Bases and the OASIS Model *[OASIS 3.0, 1998]*. We also research the problem of forgetting part of the history. Although our results have not been implemented yet the theoretical results are promising.

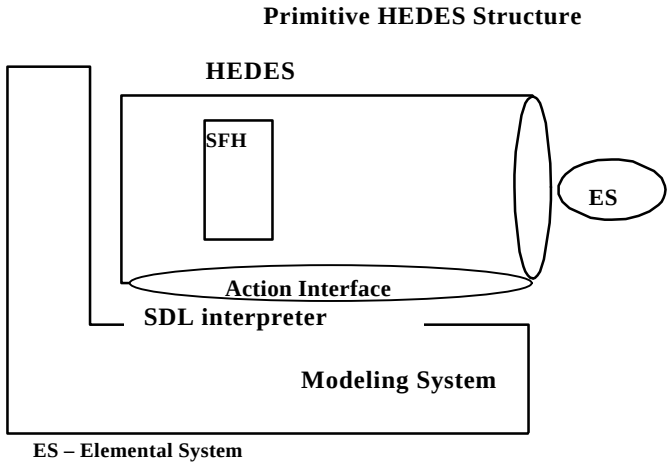

Fig.3

5 Trends of research

Although the notion of evolution is 150 years old *[Darwin, 1859]*, its application to different scientific disciplines has only begun to be carried out this century. At the end of the XIX century Engels applied the notion to the study and explanation of human societies *[Engels, 1884]*. In the first part of the XX century the concepts were developed in biology and were applied to

Epistemologic	Theory of the G.S.			Biological models Of evolution	Learning models	Classification models
Methodological (MEDES)	Functionality	Structure	Concurrence	Self-evolution and Reproduction		Evolutionary Classification
Practical (HEDES)	SFH Decision	SEH Meta-decision	Running monitor	Genetic subsystem (Reasoning system)		Dynamic classes
Instrumental: Representation	Temporal logic OASIS Model Coloured Petri Nets			Temporal logic Artificial intelligence		Delegation
Instrumental: Implement.	Object-orientation (Smalltalk/VisualWorks) OASIS					

the understanding of the origin of life *[Oparin, 1936*]. Physics, always apart from the irreversibility which implies the biological evolution, assumed evolution when the new cosmological theories and the thermodynamics of disequilibrium of Prigogin were formulated *[Prigogine, 1988]*. The selfsame development of science, which is conceived as an accumulative process, was 'adjusted' to the evolution paradigm by *Khun [1962]*. In the psychological field the work of *Piaget* introduces the concepts of accommodation and assimilation which are very fruitful in modeling the conception of software systems *[Torres&Parets, 1996]*.

In Computer Science the notion of software evolution is recent, has very different meanings and the models applied are quite 'endogamic'. A brief review of the main trends in this field can be found in *[Parets&Torres, 1996B]*. We believe that practical research on software evolution needs a continuous interaction with epistemological and theoretical research without loosing applicability. This implies that we need to adopt a transdiciplinary view of evolution and adapt models of other disciplines.

The practical approach presented in the previous paragraph is included in a more global vision of the problem of evolution, which implies different levels of research continuously interacting between them.

The table below shows these levels: epistemological, methodological, practical and instrumental and the topics/disciplines which we are interested in: General System Theory, Biological models and Classification models.

1) EPISTEMOLOGICAL LEVEL:

The study of System Theory was conducted in *Parets-Llorca [1995]* and allowed us to adopt the previous conception of systems and developments. Some topics need further research:

- Self-organization and self-evolution of systems.
- Alternative models of evolution in biology and cognitive psychology.
- A theoretical model of dynamic classification.

2) METHODOLOGICAL LEVEL

The conclusions obtained in 1) should be included in MEDES.

3) PRACTICAL AND INSTRUMENTAL LEVEL

We need representations which could be treated using formal tools (temporal logic, Petri Nets, OASIS Model ...) and could be implemented in Smalltalk. As stated in the previous paragraph, we are using representations of processors, history and functionality, which can be implemented. We will need to find these representations for the new concepts developed in the future at the epistemological and methodological levels.

6 Conclusions

We have presented in this paper a brief summary of our research work in modeling the evolution of Information Systems through Evolutionary Software Systems. Our

approach derives from the Theory of the General System and tries to model the activity of the Development System in constructing a Software System.

The main provisional conclusions of our work are:

1) Models of evolution from other disciplines are needed
2) The changes in a software system and the activity which produce these changes can be represented in an isomorphic way using some conceptual and formal tools.
3) The conception and modeling of the software development process should be improved including the metaprocess *[Conradi et al., 1993]*.

7 Bibliography

[1] A. Anaya, M.J. Rodríguez, P. Paderewski, J. Parets: Time in the evolution and functionality of information systems. *Jornadas de trabajo en Ingeniería del Software*. Sevilla 14/15 Noviembre, 1996.

[2]A. Anaya, M.J. Rodríguez, J. Parets: Representation and mangement of memory and decision in evolving software systems. In: F.Pichler, R. Moreno Díaz (eds.): *Computer Aided Systems Theory- EUROCAST'97 Lecture notes in Computer Science No.*1333 Berlin: Springer-Verlag 1997.

[3] R. Conradi; C. Fernström; A. Fuggetta (1993). A Conceptual Framework for Evolving Software Processes. *ACM SIGSOFT SEN*. Vol. 18 No.4 pp.26-34

[4] C.R. Darwin: *Of the origin of species by means of natural selection: Or the Preservation of Favoured Races in the Struggle for Life*. London, 1859.

[5] D.M. Gabbay M. Reynolds:Towards a computational treatment of time. In: D.M.Gabbay, C.J.Hogger: *Handbook of logic in Artificial Intelligence and Logic Programming.* Vol. 4: Epistemic and Temporal Reasoning. Oxford Science Publications. Oxford: Clarendon press 1995, pp.351-431

[6] T. Khun: *The Structure of Scientific Revolutions*. University Chicago Press, 1962 (Spanish translation: La estructura de las revoluciones científicas. FCE, México, 1978)

[7] J. L. Le Moigne: *La Théorie du systéme général. Thórie de la modelisation*. Paris: Presses Universitaires de France 1977-1983-1990.

[8] P. Letelier, I. Ramos, P. Sánchez, O. Pastor. *OASIS versión 3.0: Un enfoque formal para el modelado conceptual orientado al objeto.* Servicio de publicaciones Universidad Politécnica de Valencia. 1998.

[9] J. Parets, A. Anaya, M. J. Rodríguez, P.Parewski: A Representation of Software Systems Evolution Based on the Theory of the General System. In: F.Pichler, R. Moreno Díaz (eds.): *Computer Aided Systems Theory-EUROCAST'93. Lecture notes in Computer Science* No. 763. Berlin: Springer-Verlag 1994, pp.96-109.

[10] J. Parets-LLorca: *Reflexiones sobre el proceso de concepción de sistemas complejos: MEDES: un método de especificación, desarrollo y evolución de sistemas software*. Doctoral Thesis. Universidad de Granada 1995.

[11] J. Parets, J.C. Torres: A Language for Describing Complex-Evolutive Software System. In: F.Pichler, R. Moreno Díaz (eds.): *Computer Aided Systems Theory-EUROCAST'95. Lecture notes in Computer Science* No. 1030. Berlin: Springer-Verlag 1996A, pp.181-199.

[12] J. Parets, J.C. Torres: Software Maintenance versus Software Evolution: An Approach to Software System Evolution. IEEE *Conference and Workshop an Computer Based Systems*. Friedrichafen. March 1996B, pp.134-141.

[13] J. Parets-LLorca: The evolution of Software System: a framework for modelling organizational evolution. *Computational Engineering in Systems Applications*. Lille, July, 1996.

[14] I. Prigogine: *La nascita del tempo*. Edizioni Theoria. Roma, 1988.

[15] J.J. Torres; J. Parets: Biological Evolutive Models Applied to the Evolution of Software Systems. *Third European Congress on Systems Science*. Rome. October. 1996, pp. 705-710.