

Multiple-Scale Systems Modeling: Three Cases Represented by a Computerized Platform

Gerardo Febres, PhD¹

¹ Universidad Simón Bolívar, Venezuela, gerardofebres@usb.ve

Abstract— *This paper presents some results of a platform for the modeling and visualization of complex systems. The platform has the capacity to represent different aspects of complex models at different observation scales simultaneously. This tool offers advantages in the sense of favoring the perception of the phenomenon of the emergence of information, associated with changes of scale. Criteria used for the construction of this modeling platform, as well as some data abstraction tools designed for this purpose are included: structure for recording data, internal language for data processing and management, capabilities for the operation with files and location of agents and their attributes, and graphic resource management. One of the techniques that has recently emerged for the analysis of complex systems is based on graphical representation. The ability of current computers has made viable the use of graphic resources such as shapes, colors and transparencies for the graphic modeling of systems made up of many elements. By visualizing diagrams, conveniently designed to highlight contrasts, these modeling platforms allow the recognition of patterns that drive our understanding of systems and their structure. Graphs that reflect the benefits of the tool in the visualization of systems at different scales of observation are presented to illustrate the application of the system.*

Keywords— *Complex systems modeling, systems architecture, system's model complexity, observation scale, visualization, agent-based systems.*

I. INTRODUCTION

The exponentially increasing capacity of computers has enabled the numerical modeling of systems that a few decades ago was beyond our practical reach. The explosion of forms and styles to undertake the analysis of these systems has led to the emergence of new ways of organizing research around names such as the Science of Complexity and Data Science. Whether or not this is really a 'new' Science, it reflects the different 'methods' of doing things today, when we have a rapidly increasing computational power, characterized not only by the capacity of the devices but also by the refinement of today's algorithms.

Some decades ago, when computers were still humble calculating machines, we used to understand phenomena by identifying their key aspects; the dominant factors of their behavior. To this end, science endeavored to synthesize the description of systems and reduce it to simple mathematical expressions. Thus, our understanding of the world was almost constrained by our possibilities for synthesis, leaving out complex and chaotic behavior. By the end of the 80's, when computers became a commonly used research tool, yet their use was practically limited to the repetitions of deterministic calculations, leaving aside considering the phenomena of information emergence which takes place when the representation of a system changes from one scale to another.

In spite of the initially algorithmic-centered, and afterwards object-oriented, programming techniques, it was already recognized that multiple scale systems modeling required more flexible paradigms of programming. Heylighen [1] for example, foresaw in 1991 the need for computerized systems with the ability to select different ways of viewing the object-system, evaluate some properties and thus, the emergence of complex system's computer modeling as we know it today. Nevertheless, there still were not fully capable computers to develop in practice Heylighen's emerging ideas about modeling.

The development of a set good practices and programming paradigms has been matter of discussion for the last three decades. During 1987 Geoffrion [2,3] presented a series of papers defining the so called *Structural Modeling Language* (SML). The SML relied on a modular structure aiming to organize the model's entities and to cope with its complexity. This approach, however, needed to fix a priori the coarsest and finest detail levels of the model, with its obvious disadvantages regarding the adaptation possibilities. In this line of development the *Computer Aided Software Engineering* (CASE) appeared in the early 90's as a formal methodology to establish the limits of the model, the internal entity relationships and to recognize the system's model major modules. With the increasing diversity of situations where computerized models were applied, more flexible conceptions of computerized systems and their design process appeared. In 2004, for example, Makowsky [4] offered the *Structural Modeling Technology* (SMT); a set of paradigms and good practices directed to cope with the defies imposed by complex systems. But the traditional structures used to represent the vast volume of data we now have access to, are still predominantly databases. They are structures of regular shapes and great simplicity, probably the simplest imaginable, which being able to organize the data in orthogonal grids, offer great advantages for data rapid location. However, traditional databases represent very different forms from those of the modeled system. Nature is not orthogonal. Perhaps because of the limitations of our 'mental languages' for processing, the systems models performed through databases take the same forms that we interpret about the modeled system, but do not allow the system itself to describe its form by means of the model. Databases are too rigid structures.

The traditional practices for the development of systems and the databases do not allow for an easy system's representation at different scales. Even more distant is the possibility for modeling the system at multiple scales within the same observation of the system. This paper presents four features that should be included as part of the internal structure

of programs for the modeling of complex systems, in order to increase their possibilities for adaptation to the changes of the real system, as well as to effectively represent the phenomenon of information emergence that occurs when the scale of observation changes. Those features are:

- Structure for recording data.
- Internal language for data processing and management.
- Capabilities for the operation with files and location of agents and their attributes.
- Graphic resource management.

II. ELEMENTS OF MULTIPLE-SCALE SYSTEMS MODELING

Due to their nature, modeling complex systems is an activity difficult to plan. Complexity itself resists being synthesized and essential or dominant aspects of the system modelled are hard to recognize. In fact, most of the complex systems models are justified as a tool to learn about the behavior and properties of the system. Therefore, the conventional paradigms of computer model design are prone to fail when the subject of the model is complex.

MoNET is the name of the platform used as basis in this work. *MoNET* has shown great capacity for modeling systems whose complexity seemed, before being approached with these methods, far from being dominated. There are four components in which we think these *MoNET* capabilities reside. This section depicts the aspects considered are essential for the success of any complex system analysis platform.

A. Structure for recording data

Whereas traditional data-structures, made up of tables, leave little freedom to adjust their form to the condition of the modeled system, the data organized in the form of a network offer the capacity to grow in a virtually indefinite form. This growth capacity is useful not only from the point of view of size, but also from the point of view of form. A typical barrier in systems with data recorded in conventional data bases is the construction of tables in which fields are assigned for the registration of properties of the entities to which each table is destined. This implies that the design of the system must advance to establish what properties the agents accurately describe the system, denying in this way, or in any case compromising, the possibilities that the system itself has to indicate in which aspect it is more convenient to grow or tune in to deeper levels of detail.

The structure for the proposed data record is in the form of a network. More specifically it is a file tree that can be shared among several data storage devices. Such a configuration can be considered as a Free-Scale structure that can grow with virtually no limits.

Three types of files have been sighted to organize the agents that make up the modeled system. The first of these types corresponds to files describing elementary entities. The elementary entities are considered indivisible and therefore contain the values of the properties that describe the element to which they correspond. The values of the properties of the elementary entities must be data entered into the system as

information. That is, they cannot be the result of calculations made on other sources of information. Due to their extreme condition in the data structure, we see these files as the data structure leaves. These files can be recognized by their .NPD extension.

The second file type describes a group of other elements. Conveniently, the elements included as part of the group share some properties that establish a certain affinity with each other. In fact, when the affinity between these elements, which can be considered the parts of the element that contains them, is very important, the container becomes an entity with its own identity. Without reaching these levels of affinity, container elements of this type can be used to impose hierarchical and classification rules among the elements of a complex system. To the files that play this role, we consider them as the branches within the general structure of data. The assigned extension is .NPM.

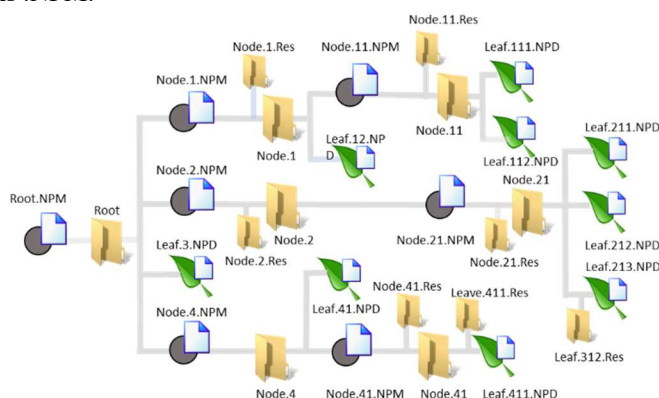


Fig. 1 Hypothetical model of file structure showing the relationship between the files and the hierarchical membership relationship.

A third type of file is used to record a selection of elements, branches or leaves. Once the elements of a sub-set of the system have been selected, they can be represented in the same graphical interface, sharing all the available tools of analysis and graphs for their study. This feature provides *MoNET* with the capacity to treat the model from different scales of observation simultaneously. The extension of these files is .NPS.

B. Internal language for data processing and management

For computerized systems operating over unstructured data—data not organized according to its position in a table of a database—, some intelligence is needed. The capacity of identification and location of agents is essential. In the absence of a database there are no database management-codes available. The handling of the information depends then, on pseudo-languages that must be elaborated by the constructor of the system. The purpose of this document is not to present complete documentation on the script language developed to serve *MoNET*. However I have considered it convenient to include here the description of some of its characteristics. Let's start by saying that we will use the name *Localizer* to refer to it. *Localizer* uses delimited tags with the '<' and '>' characters,

similar to those used by the *html* and *xml* languages. The file describing an agent consists of statements that, except for special cases, occupy a line in the text file. There are statements to specify the agent's name, the location of the file on the web, the agents directly related and the agents contained.

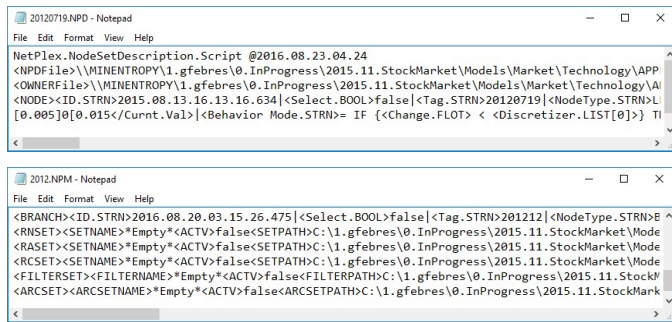


Fig. 2 Three examples of files associated with the agent description for the *MoNET* system. Above: start of the branch agent file (*NODE*). Below: file of a leaf type agent (*LEAF*).

The `<NODE>` and `</NODE>` tags are used to indicate the start and end of an agent description. Between these two tags is specified the path to locate the corresponding file in the space of the web where the system exists, and the parameters that describe each agent. These parameters, or properties of each contained agent, are recorded by a tag along with the value of the parameter and the expression for its calculation, when applied. The character `|` operates as a separator of the sentences referring to each parameter.

The name of the properties or attributes of the agents must include the specification of the data type. Thus, if an attribute is used to register the name of an agent, the attribute must be referred to as *Name.STRN*, which specifies that it is a string type. The data types included are: *.STRN*, *.INTG*, *.FLOT*, *.BOOL*, *.LINK*, *.LIST*, *.STRC* and *.EXEC*, corresponding to string, integer, floating, boolean, file link, element list, structure of elements and executable command respectively.

C. Capabilities for the operation with files and location of agents and their attributes

The replacement of the classical database with independent data files imposes the need to develop strategies for locating files according to criteria and filters. Commands that define the search addresses and other criteria for the location of the required information are essential for the proper functioning of a system with this architecture. The specific forms of these commands are many and grow as the simulation platform evolves. This document cites some cases. For example, in the agent localization routines within the system data structure, the characteristics of the wanted elements can be specified. Thus, the `<BRANCH>` or `<LEAF>` tags would indicate that the searched nodes are branches or leaves. If the tags were `<BRANCH.SUPRA>` (or `<LEAF.SUB>`), then they would be branches (or LEAFs) in the higher (or lower) hierarchy nodes with reference to the node from where the search starts.

The specification of subsets of agents within the complex system must be useful, not only to impose filters to allow the selection of information, but also for its use as parameters to condition the scope of the equations describing the interrelationships of the agents of the system and their behavior.

D. The Autonomous Data Representation

The complexity of a system lays in the amount of information required to describe it [5]. Considering also a system as the result of the overlapping the actions of many subsystems, each with its own structures, its description frequently becomes a difficult task. A way to cope with these difficulties is to extend the data types, so that a single data type –or somehow a special data type capable of adapting to the required form– can represent a multidimensional structure. This capacity should include not only the ability to represent arrays and trees, but also non-regular structures such as meshes. The *Autonomous Data Representation* was developed with the goal of meeting these requirements.

The *Autonomous Data Representation* is a logical syntactic representation serves to represent two classes of structure topologies. The first class include regular structures as orthogonal arrays of many dimensions, and scale-free structures as trees. The second class include networks which may not be seen as regular topologies; this is the most challenging application of this technique.

Figure 3 shows examples of structures of various dimensional shapes, represented according to the syntax proposed by the *Autonomous Data Representation*. The representation consists of separating the single values of the array by using a special splitter symbol. The splitter symbol itself indicates the dimensional substructures it is separating. The splitter symbol presents square brackets pointing outwards in both ends. Hence, if the structures being separated is an array of three dimensions, the splitter symbol `]0[` defines the 2-dimensional arrays comprised in the 3-dimensional structure, the splitter symbol `]1[` indicates the limits of the 1-dimensional arrays comprised in the 2-dimensional arrays and lastly, the symbol `]2[` indicates the 0-dimensional, elementary values comprising the 1-dimensional arrays.

Figure 4 shows examples of the representation some not regular networks. When the network's shape offers the possibility of being described with a noticeable characteristic, listing the node tags and this characteristic may suffice to form a precise description. Thus, the network a) in Fig. 4 can be seen as either a 3-element clique or a 3-element ring. Therefore it can be described as `<Cq>{A]0[B]0[C}` or `<Rn>{A]0[B]0[C}` where `<Cq>` and `<Rn>` are the corresponding net characteristic topology tags and *A*, *B* and *C* are the values representing some property at each node. Networks c) and d) are a 5-element ring and a 5-element star respectively. Hence their descriptions include the tags `<Rn>` and `<St>`. The net shown in e) can be seen as the superposition of the ring and the star of cases c) and d), therefore its description can be expressed by shifting the splitters' dimension indexes and using the dimensional index `]0[` to join them in a unique description. Similarly, in

case g) the dimensional index $J0I$ is used to join two networks thru elements D and K , and forming a description for the whole structure. The linking elements are indicated with the tag $\langle * \rangle$.

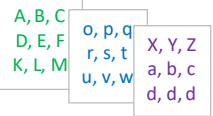
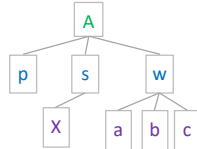
Multidimensional structure representation			
Struct. Name	Struct. Dims.	Structure Depiction	Autonomous Representation
Scalar	0	A	A
Tuple	1	A,B	A]0[B
Vector	1	G, F, D, S, A	G]0[F]0[D]0[S]0[A
Matrix	2	G, F, D, S, A 1, 2, 3, 4, 5 v, w, x, y, z	G]1[F]1[D]1[S]1[A]0[1]1[2]1[3]1[4]1[5]0[v]1[w]1[x]1[y]1[z
Matrix	3		A]2[B]2[C]1[D]2[E]2[F]1[K]2[L]2[M]0[o]2[p]2[q]1[r]2[s]2[t]1[u]2[v]2[w]0[X]2[Y]2[Z]1[a]2[b]2[c]1[d]2[d]2[d
Tree	>1 <2		A]0[p]1[s]2[X]1[w]2[a]2[b]2[c

Fig. 3 Examples of multidimensional structures according to the Autonomous Data Representation.


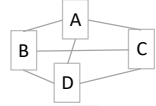
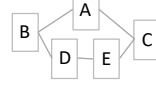
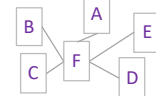
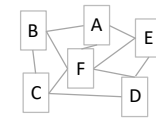
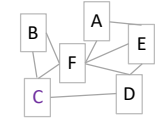
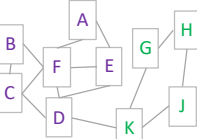
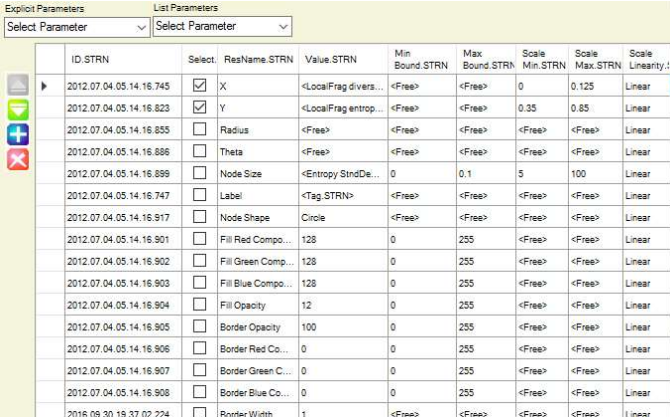
Network structure representation		
Network Name	Structure Depiction	Autonomous Representation
a) 3-Element Clique or 3-E Ring		$\langle Rn \rangle \{A\}0\{B\}0\{C\}$ or $\langle Cq \rangle \{A\}0\{B\}0\{C\}$
b) 4-Element Clique		$\langle Cq \rangle \{A\}0\{B\}0\{C\}0\{D\}$
c) 5-Element Ring		$\langle Rn \rangle \{A\}0\{B\}0\{D\}0\{E\}0\{C\}$ Notice the order has meaning; A is not in direct contact with D.
d) 5-Element F Centered Star		$\langle St \rangle \{F\}0\{A\}1\{B\}1\{C\}1\{D\}1\{E\}$ Notice F is in a hierarchical different position from other elements.
e) 5-Element F-Centered Star Plus a 5-E Ring		$\langle St \rangle \{F\}0\{A\}1\{B\}1\{C\}1\{D\}1\{E\} +$ $\langle Rn \rangle \{A\}0\{B\}0\{C\}0\{D\}0\{E\}$ or $\langle St \rangle \{F\}1\{A\}2\{B\}2\{C\}2\{D\}2\{E\}0\langle Rn \rangle \{A\}1\{B\}1\{C\}1\{D\}1\{E\}$
f) 5-Element F-Centered Star Plus an incomplete 5-E Ring		$\langle St \rangle \{F\}1\{A\}2\{B\}2\{C\}2\{D\}2\{E\}0\{A\}1\{E\}1\{D\}1\{C\}1\{B\}$
g) 5-Element F-Centered Star Plus an incomplete 5-E Ring plus a 4-Element Ring connected by elements D and K		$\langle St \rangle \{F\}2\{A\}3\{B\}3\{C\}3\{D\}3\{E\}1\{A\}2\{E\}2\langle * \rangle 2\{C\}2\{B\} \{0I\}$ $\langle Rn \rangle \{G\}1\{H\}1\{J\}1\langle * \rangle \{K\}$

Fig. 4 Examples of network synthetic representation with the Autonomous Data Representation.

E. Graphic resource management

Recently graphical representation of data has become a very active field of research. The capacity of current computers allows the development of techniques to represent two-dimensional graphs, showing information referring to phenomena that exist in more than two dimensions. Thus, using bubbles, instead of points, with diameters and variable colors, it is possible to go beyond the two dimensions in graphics that in the strict sense, remain 2D.

The graphic representation is a language in itself. The graphing capabilities should be able to adjust to the requirements of each particular situation to maximize the amount of information transferred to the observer. One way to equip the system with these possibilities is to allow the association of the properties of the agents with the graphic properties of the graphic elements used. As an example we can cite the diagrams of *Gapminder* [6] that use bubbles to represent agents or entities. The diameters of the bubbles are associated to some extensive property of the entity; population, volume and size of the group are typical cases. *MoNET* incorporates the use of graphical properties as a philosophy that manages those graphical resources. The intensive use of this philosophy allows the representation of many dimensions in 2D chart. The components of each primary color, the shape and thickness of the edge of the bubbles, the degree of fill opacity and the edge are some of the graphic properties that can be associated with the value of the attributes of each agent represented in the graph. Figure 5 shows one of the reticles dedicated to this aspect of the system.



ID	STRN	Select	ResName	STRN	Min Bound	Max Bound	Scale Min	Scale Max	Scale Linearity
	2012.07.04.05.14.16.745	<input checked="" type="checkbox"/>	X	<LocalFrag divers...	<Free>	<Free>	0	0.125	Linear
	2012.07.04.05.14.16.823	<input checked="" type="checkbox"/>	Y	<LocalFrag entrop...	<Free>	<Free>	0.35	0.85	Linear
	2012.07.04.05.14.16.855	<input type="checkbox"/>	Radius	<Free>	<Free>	<Free>	<Free>	<Free>	Linear
	2012.07.04.05.14.16.886	<input type="checkbox"/>	Theta	<Free>	<Free>	<Free>	<Free>	<Free>	Linear
	2012.07.04.05.14.16.899	<input type="checkbox"/>	Node Size	<Entropy StndDe...	0	0.1	5	100	Linear
	2012.07.04.05.14.16.747	<input type="checkbox"/>	Label	<Tag STRN>	<Free>	<Free>	<Free>	<Free>	Linear
	2012.07.04.05.14.16.917	<input type="checkbox"/>	Node Shape	Circle	<Free>	<Free>	<Free>	<Free>	Linear
	2012.07.04.05.14.16.901	<input type="checkbox"/>	Fill Red Compo...	128	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.902	<input type="checkbox"/>	Fill Green Compo...	128	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.903	<input type="checkbox"/>	Fill Blue Compo...	128	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.904	<input type="checkbox"/>	Fill Opacity	12	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.905	<input type="checkbox"/>	Border Opacity	100	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.906	<input type="checkbox"/>	Border Red Co...	0	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.907	<input type="checkbox"/>	Border Green C...	0	0	255	<Free>	<Free>	Linear
	2012.07.04.05.14.16.908	<input type="checkbox"/>	Border Blue Co...	0	0	255	<Free>	<Free>	Linear
	2016.08.30.19.37.02.224	<input type="checkbox"/>	Border Width	1	<Free>	<Free>	<Free>	<Free>	Linear

Fig. 5 *MoNET* Graphics Resource Management Panel.

III. APPLICATIONS AND RESULTS

The development of *MoNET* as a computer platform is about four years old. During this period *MoNET* has been used as the basis for the conformation of several experiments. After the conception of the idea and the elaboration of an initial model, the construction of the system has been guided by the answers to the needs that appear in the development of those experiments, always maintaining some basic rules of

programming such as using abstract representations of data that allow its universal application.

The representation of complex systems at various scales have allowed the realization of several studies in which this tool was used to measure the complexity of various natural and artificial languages[7], to evaluate writing qualities in the English and Spanish languages [8] and compare the entropy of languages of diverse nature such as natural languages and music [9]. This paper shows three examples of complex systems described by *MoNET*. The first refers to a model that represents the music seen according to its entropy. The second an evaluation of the variations of the prices of a stock in the Stock Exchange and the third the comparison of styles of writing in English and Spanish. Representative graphs of each of these systems are included in Appendixes A, B and C.

A. The effectiveness of the data structure

The organization of data in a hierarchical way in a tree-shaped structure offers advantages over its orthogonal counterpart such as tables in conventional databases. The tree structure organizes the agents, each formed by a data file, into nested directories according to the hierarchical order considered with dominant nature in the modeled system. It happens that for most recognizable systems, this hierarchical organization leads to the recognition of subclasses of agents that populate the model with numbers distributed in an approximate way to the logarithmic (or exponential depending on the direction it is seen).

Table 1 shows the number of nodes (directories and files) corresponding to each level of the data structure in three models mentioned. The classification of nodes is according to a certain hierarchy that organizes the agents with a criterion similar to the one used by the algorithm Quick Sort; classifying the elements according to a lower hierarchical criterion in several groups. This configuration of the data-structure represents in a more reliably fashion, the hierarchical structure of the components of the modeled complex system, if compared to its counterpart structure developed with an orthogonal database.

TABLE I
ELEMENTS OF SYSTEMS USED AS CASES OF APPLICATION

	Nature of Nodes. Hierarchical Classification	Number of nodes	Entropy of the Node Degree Distribution
Entropy of music	Periods, styles, genres	12	0.312
	Composers	69	
	Musical pieces	472	
	Fragments of pieces	3939	
	Total	4492	
Stock Market share price	Year	11	0.201
	Month	132	
	Day	2563	
	Total	2706	
Writing Style evaluation	Language (Spanish/English)	2	0.140
	Original / Translation	4	
	Speech / Novel-Story	8	
	Text	363	
	Total	377	

B. Capabilities of the multiple scale representation

The philosophy of managing graphics resources to increase the readability of two-dimensional graphics has allowed the representation of seven or even more dimensions in 2D graphics. The graphical resources used include the positions on the X and Y axes (angle and radius for polar coordinates) and various graphical properties of the bubbles that represent each agent within the system such as: diameter, shape, thickness of the edge line, Fill opacity, edge opacity, and component of each primary color. Appendixes A, B, and C show graphing examples where aspects of complex systems can be visualized. This philosophy of managing graphical resources is a key factor in the possibilities of representation at multiple scales of systems.

C. Pseudo-languages to handle and organize unstructured data

The script language developed that we have called *Localizer*, together with the autonomous representation of data has allowed the control of the complex data structure that serves each computer model. In order to understand the dimension of the difficulty that the program faces, the requirements of this program can be compared with those of a spreadsheet. In a spreadsheet the models are described by reference to the position of each element in a grid. These reticular structures can grow up to three dimensions, which make up the so-called 'workbooks'. In the present case, the data structure may have any shape; can be reticular, such as spreadsheets, or trees representing a certain hierarchy between data, or meshes without regularity with the capacity to represent even more complex situations. Logically, the flexibility of being able to represent any hierarchical structure or system of relations through the form of the network of data files, is paid to have to dispense with the possibility of locating the data using orthogonal coordinates, as would be the case in the Spreadsheets. A language must be available that allows the localization of data in that malleable structure that allows the natural form of any system to be associated with the form of the data structure.

IV. DISCUSSIONS

A. Flexibility vs. Data Structure

The construction of computer programs based on structured data has long been the commonly accepted way of approaching the problem of designing systems. The use of tables to represent object properties has become an effective vehicle for organizing objects represented on the computer and in general information. Techniques to represent relationships between different types of entities was a major advance in the modeling of complex systems during the 1990s. However, the platforms for computerized modeling of complex systems suffer from the constraints imposed by the rigidity of table-based architectures. The tables make it difficult to represent hierarchies and relations of belonging.

On the other hand, the hierarchically organized data structure, based on classification trees that store data in accordance with their levels of detail and observation scale,

effectively approaches the possibility of implementing data processing in parallel.

B. The true complexity of a computerized system

It is often attempted to measure the size and power of programs by specifying their number of routines or instructions. These dimensions refer more to the work and the cost of designing and coding a computerized program than to the performance of the final result. In fact, if I had to bet on the better of two programs, I would rely more on the smaller than on the heavier. There are more appropriate measures to evaluate the quality of software segments. Some of these measures are well known. One of them is the concept of Computational Complexity, which refers to the estimation of the resources required by an algorithm to achieve a result. The evaluated resources are typically time or memory space. The problem is that Computational Complexity evaluates the performance of an algorithm, while in today's most cases, a system consists of many 'coexisting' algorithms in an environment plenty of other components, and where the effectiveness of the algorithms does not necessarily define the effectiveness of the software.

As for the search and read times of the file associated with an agent, conventional databases certainly allow search times much lower than the crawling required for locating an agent in a file network. But the algorithms of search in tables require the implementation of indexes that 'hide' much fragility in the databases and that require important efforts of maintenance and security expenses.

In an environment of research and productivity where performance is more closely associated with the speed with which the computer platform conforms to the particular requirements of an experiment, it seems convenient to adopt a data structure capable of assimilating the change of the objects' nature and relationships. Having an own interpreted script-language, capable of incorporating new requirements, while keeping previously established criteria and syntax elements, or on the contrary incorporating new criteria and making the syntax to evolve, offers important advantages in this regard.

V. CONCLUSION

The presentation of three applications studied with the proposed system, are just a minimal expression of what can be done; But they are sufficient to affirm that this systems

architecture is viable and that it offers effective representations of the phenomenon of emergency of information that happens when changing the scale of observation.

The representation of complex systems based on independent file structures and without databases, seems to be the way that provides the necessary flexibility to model today's systems, whose structure changes with a dynamics that databases are unable to pursue.

When the perspective of software design is not dominated by a commercial character, the techniques that should be adopted seem to be those that offer possibilities of growth and evolution for its adaption to the increasingly frequent changes of the environment in which is applied. These results suggest that software treatment, as a language able to adapt to the requirements and evolve towards high levels of effectiveness, offers advantages in the medium term, compensating for the price of the slow start that characterizes this style of programming.

REFERENCES

- [1] F. Heylighen, "Modelling Emergence," *World Futur. J. Gen. Evol.*, vol. 31, pp. 89–104, 1991.
- [2] A. Geoffrion, "An Introduction To Structured Modeling," *Manage. Sci.*, vol. 33, no. 5, 1987.
- [3] A. Geoffrion, "1992. Geoffrion.SML.StructuredS The SML. Language for Structured Modeling: Levels 1 and 2," *Oper. Res.*, vol. 40, no. 1, pp. 38–57, 1992.
- [4] M. Makowski, "A structured modeling technology," *Eur. J. Oper. Res.*, vol. 166, no. 3, pp. 615–648, 2005.
- [5] A. D. Patel, "Language, music, and the brain: a resource-sharing framework," *Lang. Music as Cogn. Syst.*, pp. 204–223, 2012.
- [6] "Gapminder." [Online]. Available: <https://www.gapminder.org/>. [Accessed: 16-Oct-2016].
- [7] G. Febres, K. Jaffe, and C. Gershenson, "Complexity measurement of natural and artificial languages," *Complexity*, vol. 20, no. 6, pp. 429–453, 2015.
- [8] G. Febres and K. Jaffe, "Quantifying structure differences in literature using symbolic diversity and entropy criteria," *J. Quant. Linguist.*, vol. Early View, 2016.
- [9] G. Febres and K. Jaffe, "Calculating entropy at different scales among diverse communication systems," *Complexity*, vol. 0, no. 0, 2015.
- [10] G. Febres and K. Jaffe, "Music viewed by its Entropy content: A novel window for comparative analysis," *arxiv.org*, vol. arxiv:1510, 2016.

APPENDIX A

Classification of music according to its entropy and symbolic diversity at different scales of observation. Details of this research are available in [9], [10].

Figure A1 shows elements of music coded as MIDI music. An element can be a piece, a fraction of a piece or a group of pieces sharing some characteristic. Thus, for example there may be a bubble representing a movement of Beethoven's the Ninth Symphony, another bubble representing the four movements comprising the whole symphony, another bubble representing all Beethoven's pieces included in the model and another representing the set of composers considered as Classical —Mozart, Beethoven, Schubert, Paganini, Hayden, etc. —. Going in the opposite direction, the direction of smaller entities, there may be large symphony movements spitted on several fraction which in turn are shown a little bubbles. Including the spectrum of many composers and music genres, in is possible to build up a sort of map of music, showing several scales of observation at the same time.

The coordinates of the map are defined as the units assigned to the vertical and horizontal axis; in the case of Fig. 1 entropy and specific diversity respectively. The diameter of the bubbles are dimensioned proportional to the standard deviation of the entropy, so a large bubble expresses important differences among the elements comprising a set of certain class of music. For those bubbles corresponding to a set made of a single element, the standard deviation is not defined, thus a minimal diameter size has been assigned to these extreme observation scale case. The level of the scale is also represented by the thickness of the bubble border lines.

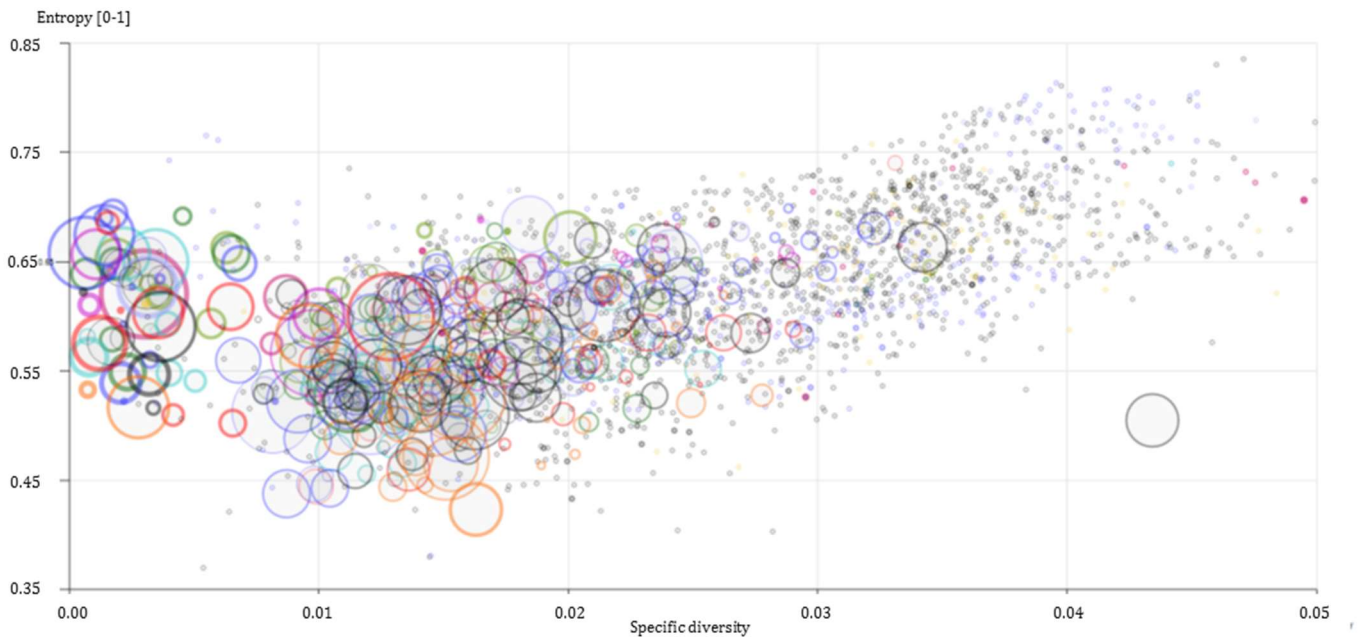


Fig. A1 Entropy vs. Diversity of music. Representation for more than 400 pieces of MIDI music. The chart shows various scales of observation: Periods or types of music, composers, pieces and fragments of pieces. The diameter of the bubbles is proportional to the standard deviation of the elements that make up each group. Thus, for example, the bubble representing romantic academic music has a diameter proportional to the standard deviation of the distribution of entropy characteristic of composers of that period. In turn, the bubble representing the music of a composer, for example Chopin, has a diameter that represents the standard deviation of the musical pieces of Chopin.

APPENDIX B

Variations of Apple Computer, Inc. stock market share value for different observed time periods. The graph shows the share price value percentage variation for three different time periods: days, months and years. The horizontal axis represents the total volume of shares changing of hands, measured in US dollars, for a specified time period. The vertical axis represents the change of the stock value, comparing the value at the end of the period with its corresponding at the beginning of the period.

Small bubble representing periods of a single day show a stock value limited by the low probability of encountering a large change in a single day. Medium sized bubbles show a larger scope in the stock value variation, but still remains restricted to a relatively small range of variations. For periods of a year, larger bubbles, the restrictions in the changes weakens and thus the range of variation increases considerably.

For each of the three time scale periods selected, there seems to be a relationship between dispersion of the stock value change and the volume of stocks changing of hands. But especially for the period of a single day, this relationship shows as a dominant factor in the dynamics of the stock market; once an important change occurs, the tendency to trigger an accelerated the purchase-sell process explains this relationship.

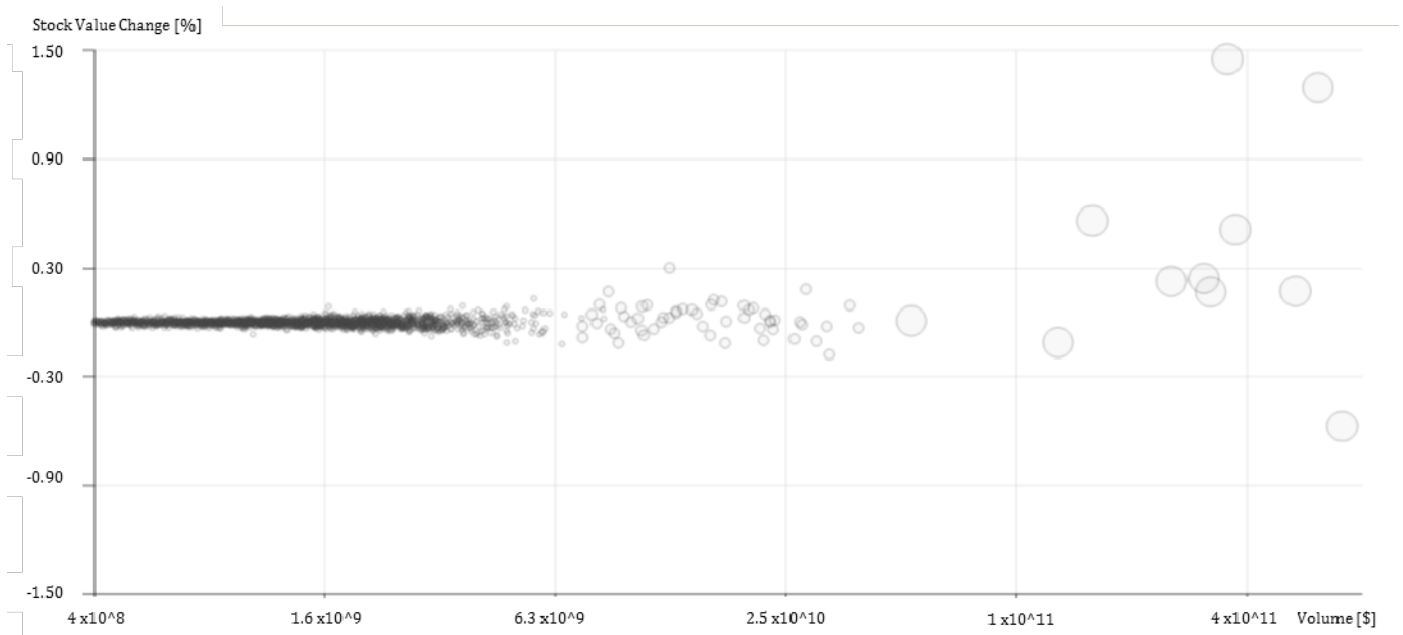


Fig. B1. Fraction of Change in the value of shares in the stock market vs. Volume of transactions. Chart made for Apple Computer, Inc. from January 2006 to July 2016. The values on the vertical axis represent percentage changes of the value for different periods: large bubbles represent years, medium months and small days.

APPENDIX C

Comparison of writing styles in English and Spanish. Details available at [8].

The comparison is done using the Writing Quality Scale (WQS); a scale which evaluates the text considers its relative entropy (the separation from the entropy which normally would correspond to a text of the same length), the relative specific diversity (the separation from the specific symbolic diversity which normally would correspond to a text of the same length), and the under the ordered symbol frequency profile J_{1D} .

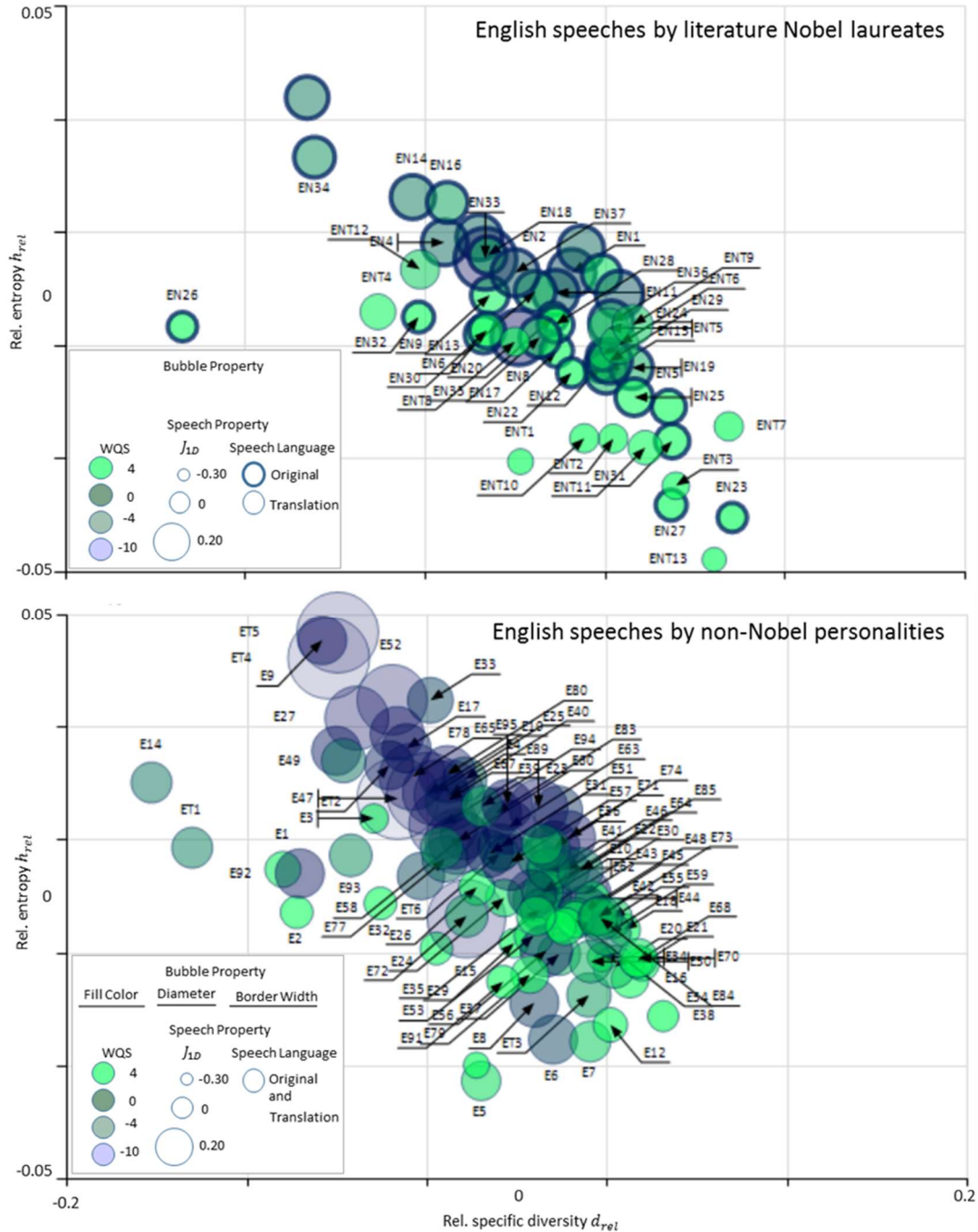


Fig. C1 Entropy vs. Diversity of speeches expressed in English. Representation of speeches by several authors. The chart above shows the speeches of authors who have won the Nobel Prize for literature. Other authors appear below. The color of the bubbles represents the score on the WQS scale.

Figures C1 and C2 show texts originally written in English and Spanish Languages. But there are also texts translated into these languages. When a bubble represent a Novel Laureate text expressed in its original language the bubble is shown with a thick border. The letter E in the tags means English, the S is for Spanish and the N for Nobel Laureate.

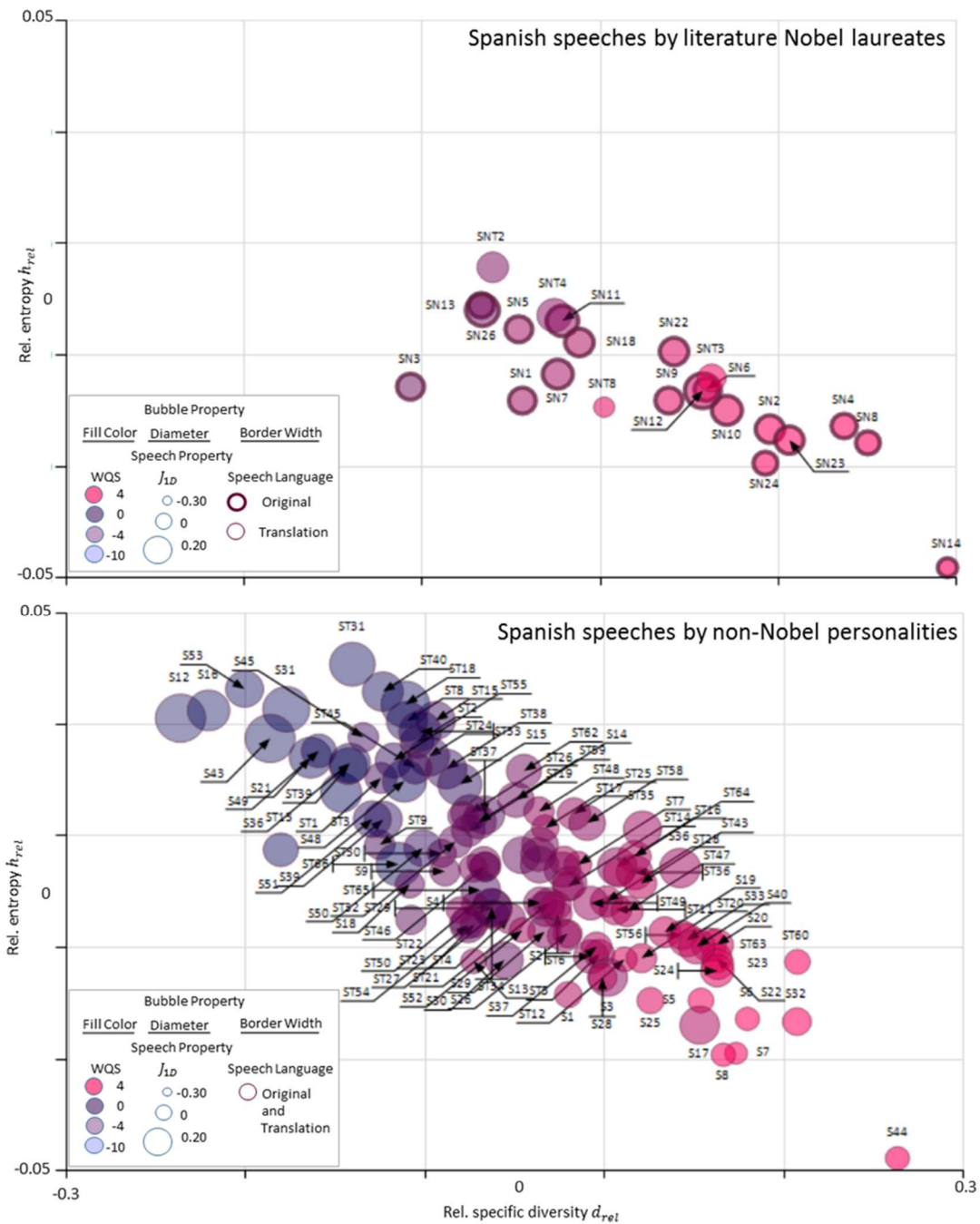


Fig. C2 Entropy vs. Diversity of speeches expressed in Spanish. Representation of speeches by several authors. The chart above shows the speeches of authors who have won the Nobel Prize for literature. Other authors appear below. The color of the bubbles represents the score on the WQS scale.