Organizing Thoughts into Sequences, Hierarchies, and Networks

Andrius Kulikauskas
Minciu Sodas Laboratory
Grudu 6, 2020 Vilnius, Lithuania
e-mail: ms@ms.lt

Saulius Maskeliunas
Institute of Mathematics and Informatics
Akademijos 4, 2600 Vilnius, Lithuania
e-mail:

ABSTRACT

Encapsulation allows one to distinguish external relations defined in terms of capsules from internal structure defined within a capsule.

External structuring builds complex structures with capsules as building blocks. Sequences, hierarchies, and networks are three basic external structuring modes. In practice, when a sequence, hierarchy, or network grows too large, an additional structuring must be introduced to get things back under control. Each restructuring makes for a qualitatively different visualization, giving six basic visualization types.

Internal structuring starts with an encapsulation of internal structure and unwinds it from the outside moving in, releasing meaning step by step. In practice the untangling of inner structure, decomposing a record into fields and sub-fields, occurs upon knowing or guessing the schema needed to parse this record.

As a result of this distinction, the need for a computer standard for the export and import of data sequences, hierarchies, and networks is revealed.

1. INTRODUCTION

This paper presents four criteria for defining sequences, hierarchies, and networks (SHN) in the way most appropriate for organizing thoughts. These criteria are the preference of writers, the accordance with introspective reflection, the applicability to human practice, and the ability to explain observed visualizations. Such criteria are important in designing software to support independent thinkers working on open ended projects involving thousands of notes. Arranging thoughts into sequences, hierarchies, or networks promotes radically different kinds of thinking. Mathematically, there is a sense in which these three structures are redundant, in that any one structure can encode the other two. Psychologically, these structures have very different effects on thinking, and therefore we want to define the structures in the way that isolates the effect of each (see: Section 2).

One criterion for defining SHN structures is the preference of writers who use different features of software to organize notes in different ways. Unfortunately, the effects of the structures are most apparent when working on an overwhelming body of notes, and few writers use software for open ended projects. The lack of a standard for the import and export of structured notes hinders the use of such software for large projects, and retards its development (Section 3). There is a need for alternative

criteria to define at least a preliminary format for the exchange of sequences, hierarchies, and networks of notes.

A second criterion is accordance with introspective reflection on personal experience. This offers a preliminary answer as to the effects we experience in organizing thoughts into sequences, hierarchies, and networks, and what aspects of these structures most heighten their effects (Section 4).

A third criterion is the applicability to human practice. Our survey of ways of organizing information is anthropological evidence for the way we humans use sequences, hierarchies, and networks. It includes the achievements of object technology, especially Unified Modeling Language methods (Section 5). The results of this survey indicate that an important theoretical distinction should be made between the three external structurings (sequence, hierarchy, network), and a fourth kind of structuring, the internal structure of records (Section 6).

By restricting our attention to external structurings, we are able to concentrate on visualizations of information, and observe that they result from restructurings of sequences, hierarchies, and networks. The main result of this paper is a classification of these restructurings into six visualizations, which helps to make sense of the role that the UML plays in bringing together different methods (Section 7). The classification applies to the mental interpretation of a diagram, rather than to the diagram itself. This is a philosophical result regarding the structuralization of human experience (Section 8). The success of this classification yields a fourth criterion for determining the most appropriate way to define a sequence, hierarchy, and network with regard to arranging thoughts. This criterion is that the definitions should offer a theoretical basis to explain the observed visualizations, as well as any future philosophical results suggested and confirmed by the anthropological data (Section 9). This paper concludes with an appeal for an SHN standard (Section 10).

2. THREE VERY DIFFERENT WAYS OF WRITING

Kestas Augutis sparked our interest in the consequences of organizing thoughts in various different ways. He had a wonderful vision for how computers should be used in school. He proposed the creation of software, 3 Knygos, with which every child could write three books: an encyclopedia, thesaurus, and chronicle [1]. A child could string together a letter to a friend by clicking on a button and walking through the encyclopedia of associations she had entered. She could generate the outline for a science report by selecting a branch from the hierarchy of words she had stored in her thesaurus. As she wrote more and more, she could look back over the progression of the work she had placed in her chronicle. She would add to her 3 Books throughout her entire education.

The different ways of writing promote different kinds of thinking. Writing in a sequence brings out the strong points and weak points in a chain of reasons. Writing in a hierarchy distinguishes the broad and narrow ideas. Writing in a network lets one use a private language and write down less, as it helps evade questions by referring them to footnotes that may never get written. One can choose a way of thinking by choosing a way of writing.

This all assumes that one knows how to write down an individual paragraph, a separate idea. A paragraph is a helpful unit to consider, in that it may have a nontrivial internal structure consisting of one or more sentences. A paragraph may get rewritten as several paragraphs, or several paragraphs may get collapsed into one. A paragraph, for our purposes, may consist of several words jotted down, or even no words at all, if the juxtaposition with other paragraphs is powerful enough. It is a form of mental punctuation, encapsulation that lets one separate the internal structure of a paragraph from its

various relations with other paragraphs.

The idea of employing all three structures on an equal basis for arranging thoughts can also be attributed to Roy Roebuck [2]. In principle, one should be able to find this idea in the literature of many fields, such as knowledge representation, conceptual modeling, structural anthropology, library science, sociology of business, etc. In practice, it is not so simple. For example, the psychology of knowledge representation seems to analyze the appropriateness of these structures in modeling psychological processing, rather than the psychological effects of employing these structures [3].

The first author of this paper can share some practical experiences from seventeen years organizing tens of thousands of short notes while working on original philosophy. It seemed impossible to write a "comprehensive theory of everything" in linear form. Questions would branch out in different tangents, and in practice could be reeled back in only through active engagement, much as in a Platonic dialogue. Concepts could often not be formally described, but illustrated only through an unbearably tedious abundance of examples. Attempts to write up these ideas always broke down, leaving stacks of short notes, scribbles, diagrams, often impenetrable, full of recurring, contradictory, and orthogonal ideas.

Hypertext editors presented a breakthrough. They made it possible to write about private words, vague ideas, and endless examples, by refering an imagined audience to explanatory footnotes that need never get written. At the same time, notes could be organized hierarchically. Given a note, such as "to love X is to create an environment where X can be sensitive, alive...", related notes could be deposited underneath. The author could review the accumulated notes, mull them over, and decide whether and how to group them further.

3. SOFTWARE FOR ARRANGING THOUGHTS

Hypertext editors make it possible to organize paragraphs in sequences, hierarchies, and networks. In practice, they are not intended for organizing thousands of paragraphs. HTML lacks a global hierarchy. Also, hypertext maps a word to a document rather than a paragraph to a paragraph. Other tools for organizing thoughts include NoteCards [4], developed at the Xerox PARC laboratory, Info Select [5], a product of Micro Logic, and The Brain [6], a product of Natrificial Software Technologies. The Brain is noteworthy because it places hierarchies and networks on an equal footing. A survey of such tools is an ongoing project of the Minciu Sodas laboratory [7].

4. DEFINING SEQUENCES, HIERARCHIES, AND NETWORKS

In April, 1998, the Minciu Sodas laboratory set out to design a prototype Imintis (Repository of Thoughts) for organizing paragraphs in sequences, hierarchies, and networks on an equal basis [8]. The idea was to impose equality by handling all three relationships with a single table.

Each record in the table would carry the information for a single paragraph and its external relations. Many questions ‘crept up’: Should the hierarchy be ordered or not ? Should a record be required to have a position within the hierarchy ? Should it be able to participate in more than one hierarchy, or more than one sequence ? Should the links in the network carry annotations ? Should multiple links from one record to another be allowed ? Should records be distinguished by primary keys, or should ambiguity be allowed ?

Ultimately, the design of the prototype became less important than the search for the answers to the

questions above. Certain answers were favored by aesthetic considerations, such as achieving parallelism among the three relationships, and storing all information within a single table. But all of the answers had to further the goal of balancing a total flexibility for writing in any manner with an absolute definiteness as to which manner was being used.

The difference between broad ideas and narrow ideas is heightened if each idea has at most one place within a global hierarchy. This does not rule out multiple inheritance, but rather treats it as a composite effect involving both the hierarchy and the network. For example, a description of a person can be linked by the network to a description of that person's role in a company, and also be linked by the network to that person's role in a political party. The hierarchy of roles in the company, and the hierarchy of roles in the political party, are different subhierarchies within the single global hierarchy.

The difference between broad ideas and narrow ideas is also emphasized if the hierarchy is unordered, rather than ordered.

Such arguments suggest the following description of a table with seven fields:

- One field is for unlimited text and holds the text of the paragraph. It may be null.
- Another field is for the hierarchical position. It may be null, but otherwise gives a unique position within a single well formed unordered hierarchy. There should be no restrictions on the depth of the hierarchy or number of branches. The prototype Imintis implemented this with strings by coding, for example, the 55th branch of the 120th branch of the root as z1zzz120zz55, where the place marker z is repeated once for each digit in the subsequent number. (Note that the hierarchy should be unordered, which means that, in theory, the records should get alphabetically grouped, but not sorted.)
- The next two fields are for the sequential position. Either may be null. The first field uniquely identifies the sequence, and might be coded with an integer. The second field identifies the position within the sequence, but need not be unique, and might be coded with a real number. This forces the writer to decide which sequence a paragraph will belong to, if any, but lets him assign two paragraphs to the same position, for example, the same date.
- The last three fields are for the network position. All three could be null, but otherwise could be coded with integers. The record describes a link within the network, and the text of the paragraph annotates that link. The text of the paragraph may be null. The first field is used to uniquely identify the record. The second field indicates the record it is linking from, and the third field indicates the record it is linking to. If the second and third field are both null, then the link is taken to be a node of the network. Note that there may be multiple links between paragraphs.

The above table description lets paragraphs be related to others in any or all of the three ways. It is a starting point for designing software for organizing thoughts. The use of such software should help clarify what each structuring can give to writing and thinking, and what each requires of the underlying table.

5. THOUGHTS AND OBJECTS

The table design described above is meant to store thoughts. It sets aside a field of unlimited text to write down each thought. A thought may be written down not only in terms of words, sentences, paragraphs, but also as data, images, formulas, macros, code, and so on. Which of these is the case

may be self evident, but it may also be impenetrable. This issue is separate from the fact that a thought may be organized with other thoughts into sequences, hierarchies, and networks. In this sense, a thought is a capsule of information. It can carry a meaning inside of itself, but it can also be related to other thoughts in a sequence, hierarchy, or network regardless of whether this meaning is understood correctly, if at all. A thought is a shell for distinguishing the meaning inside from the context outside.

If it makes sense to think of a thought as a capsule, then object technology should offer keen insights into the organization of thoughts. Object technology offers mature concepts that are the fruit of billions of hours spent organizing capsules of code. Also, it does an excellent job of employing sequences, hierarchies, and networks on an equal basis. For example, it allows execution to pass from code to code sequentially (within procedures), hierarchically (through inheritance), and referentially (via messages). This is not to say that object-oriented (OO) programming languages are the only ones to use all three. Practically every programming language has a conditional IF statement that lets execution branch through a hierarchy, and a 'GO TO' statement that lets it move through a network. (Note also that a loop can be thought of as starting with an 'IF' statement and ending with a 'GO TO' reference to the initial 'IF' statement). But object technology offers a balanced approach organized around the concept of encapsulation.

6. PARSING RECORDS AND UNWINDING MEANING

Mathematically, there are countless ways of organizing information. On what grounds, if any, might sequences, hierarchies, and networks be privileged ? The Minciu Sodas laboratory collects examples of ways that people actually do organize information [9]. This survey shows that a table of records is a fourth basic kind of structuring, which deals with the internal structure of a capsule, rather than the external structure.

A table is mentally organized very differently than a matrix , which can be thought of as an external structuring, a network of links from i to j. We think of a table as a set of records all having the same field structure. Each record can be thought of as a capsule, and these capsules may or may not be organized in a sequence. From this point of view, the field structure is inside the capsule, so that a table is a set of capsules that are understood to have the same internal structure.

Where does internal structure come from ? Suppose a writer has a thought L that links a thought A to a thought B. Information from two records A and B can be wound into, or coded into, one record L. Conversely, information from one record L can be unwound, or decoded, into two records A and B.

A writer can imagine A to contain specific data, and B to be a general template for that data, together giving an expected value for L. B might be a table structure, file type, command, formula, rule or program that operates on the given data A. L may or may not equal the expected output, but the writer can think of it as something that, in general, should.

What is the difference whether the information is given by two records A and B, or by one record L? When it is given by two records, then it is more explicit. The data A is separate from the template B. When it is given by one record L, then it is more implicit. The information from the data and the template makes the internal structure richer within L. It may take a clever guess of B to decode L and recover A. Or it may be necessary to guess A as well.

This way of thinking suggests that meaning is something wound up that we decode with the right schema, rather than something we build up from the right atoms. This means that one works from the

outside in, as in solving a math equation, where one starts with the outer parentheses, as if peeling an onion. Similarly, one can learn one's way around the World Wide Web, Java, and ASCII in that order, as each becomes relevant.

This also suggests why semantics is so difficult. It has to do with guessing the right schemas, the right templates. They are assumedly metaphysical, probably related to the various ways that hyperlinks can be used. They should accord with the kinds of questions that help one find one's bearings when one cannot rely on one's experience, such as "How does this seem to me?", "What else should I be doing ?", "Would it make any difference ?", "What do I have control over?", "Am I able to consider the question ?", "Is this the way things should be?", "Am I doing anything about this?"

The important point here for this paper is that the inner structure of tables, classes, and other templates should be thought of not as built up from capsules, but rather as defining the inner meaning of a capsule. Any such internal structure should be understood as bringing along additional internal meaning of its own, and thereby complicating things semantically. For example, aggregation imposes an internal structure on aggregates through a "part of" relationship that makes it more complicated semantically than the otherwise similar relationship of association. Other semantic concepts include collection classes, composite objects, and integrity constraints. Setting aside internal structure makes it possible to notice the remarkable but purely external visualizations that sequences, hierarchies, and networks can generate.

7. TO VISUALIZE IS TO RESTRUCTURE

A survey of the ways information gets organized supports the hypothesis that sequences, hierarchies, networks, and table structures are the natural ways in which people organize information. It also yields a surprise: sequences, hierarchies, and networks are practically always found in combination.

Imagine a scribe listing historical events in sequence. When the list is very short, it may seem trivial. As the list grows, the scribe is overwhelmed, and then introduces a hierarchy of eras and ages to get things back under control. Adding a hierarchy to a sequence yields a familiar way of visualizing information that may be called a chronicle.

This type of restructuring occurs in all six possible ways. Any one of the three external structurings gets used robustly, gets overwhelmed, and gets patched up with another of the external structurings. Each restructuring offers a different way of visualizing information. There are six visualizations in all, as shown in Table 1.

Visualization (S = sequence, H = hierarchy, N = network)	**Everyday Examples**	**Programming Examples**	**Modeling Uses (*Fowler [8])**
S to H, Chronicle Adds a hierarchy to a sequence. Chronological events get organized in eras and ages.	Table of contents, photo albums, infantry, decimal notation for whole numbers.		
H to S , Evolution Maps a sequence onto a hierarchy. Can show trees of possibilities change over time.	Origin of species, recipe book, decimal notation for real numbers, knockout tournament, genealogy, chess openings.	Activity diagrams*, Overriding, Propagation, Functional decomposition.	*Shows behavior with control structures. Can show many objects over many uses, many objects in a single use case, or implementation of method. Encourages parallel behavior.
H to N, Catalog Grafts references onto a hierarchy.	File manager, auto parts store, pantheon, codex, academia, thesaurus, bureaucracy.	Flow charts, Design by contract*	*Provides rigorous definition of operation's purpose and class's legal state. Encode these in class to enhance debugging.
N to H, Atlas Adds a hierarchy onto a network. This makes for global and local views.	Anatomy, web portal, political map, ecosystem, social network.	Package diagrams, CRC cards*, Refactoring, Extending Use Cases, Abstract classes.	*Helps get to essence of class's purpose. Good for exploring how to implement use case. Use if getting bogged down with details or if learning object approach to design.
S to N, Canon Adds a network onto a sequence, as when we have reuse.	Index of book, math proof, factory line, concordance, Scripture.	Sequence diagrams*, Patterns.	*Shows how several objects collaborate in single use case.
N to S, Tour Considers all possible walks within a network.	Correspondence, Markov process, conversation.	State diagrams*, Collaboration diagrams, Persistency, Scenarios.	*Shows how single object behaves across many use cases.

Table 1. Combinations of information external structuring modes.

These visualizations are best thought of as mental pictures. They classify a full range of examples taken from every day life, and also a wide variety of programming methods and concepts. In particular, they help classify OO methods, and explain the helpfulness of the UML in allowing for complementary views. However, the visualizations do not classify diagrams themselves, but rather the

associated mental pictures. For this very reason the visualizations can also classify OO concepts that involve mental pictures. Helpful sources of examples are books that offer a high dose of intuition, such as Fowler and Scott [10], and Taylor [11].

Suppose one shows the divergence of biological species with a tree, and then introduces a time sequence so that sub-trees can be selected. This visualization may be called an evolution. It is the mental picture offered by a typical Activity diagram. Such a diagram draws out the divergent possibilities of a complicated method, and then brings the threads back together with synchronization bars. The two diagrams may look quite different, but the mind ultimately reads them in the same way. The unrelated OO concept of Overriding calls up the same mental picture. It first presumes a hierarchy of classes, and then establishes a sequence of priority, one that happens to favor the method defined in an outer class over the method defined in an inner class when the two have the same name.

One may extend a hierarchy with links between members of the hierarchy, or perhaps even outside of the hierarchy, creating a catalog. This is the case with a file manager that allows for short cuts or aliases. Design by Contract is an OO method that establishes a hierarchy of pre-conditions and post-conditions and then uses them to debug the network of methods that need to check for them.

If a web is getting out of hand, then it is possible to organize on top of it a hierarchy of global and local views. This collection of views of different scope may be called an atlas. An example is a political atlas with maps of continents, countries, provinces and districts. CRC cards presume this same mental picture. These cards encourage programmers to stand back from the intricate network of methods relating classes and to identify for each class its chief responsibilities and the main collaborators it must interact with to fulfill them.

A canon is the weaving of a network of associations over a sequence. A factory line can include crucial machinery that finds itself used in a complicated network of activity. In Scripture, new associations can be found for key passages. Sequence diagrams are a third example. They are Interaction diagrams that make evident an object's lifeline and the network of methods that affect it.

A tour is the attempt to make sense of a network by walking through it. A conversation between good friends is a tour through the many topics of interest that they share. A State diagram is a tour that lets one follow an object as it passes from activity to activity.

8. SELECTING VISUALIZATIONS

By the classification above, we can consider whether two methods rely on fundamentally the same mental picture, or two radically different ones. This helps in deciding whether a method is superfluous, whether two methods complement each other, and whether one is working with a full set of methods.

Note that this classification does not apply directly to diagrams, but rather to the mental pictures they presume. However, it can be used to analyze diagrams by indicating which mental pictures are most likely. In this way it is possible to guess the nature of a diagram without knowing the meaning of its elements, which may be given in a foreign language. Similarly, concepts can be more quickly understood by learning the associated mental pictures. The classification suggests that a diagram can and often should emphasize a single mental picture, and diagrams should complement each other. It is unclear what advantages ambiguous diagrams may have.

This classification also predicts the ways a concept may develop. Introspection suggests that a

restructuring must occur for one to be able to visualize. For example, a use case is a network of relationships between actors, a small piece of the puzzle. One may grapple with this network in two ways, as given by the classification. One way is to track a sequence, which is a tour, what the UML calls a scenario. The other way is to extend the use case in various ways, which makes certain relationships secondary, and others primary, yielding an atlas. Introspection suggests that one can continue to restructure, but that the mind can focus on only one restructuring at a time. There may be composite restructurings, but there are no composite visualizations.

The ability to classify methods makes intuition, such as Fowler's, a very interesting object of study, as suggested by Table 1. Is it true, in general, that a canon "shows how several objects collaborate in a single use case", and a tour "shows how single object behaves across many use cases"?

The choice of visualization can have profound consequences. For every restructuring, the second structuring is always more unstable than the first. For example, as time goes on, historians keep adding new "modern" ages, and the hierarchy of ages becomes lopsided. When is this instability least painful, that is, when do changes in the first structuring make for the fewest changes in the second structuring ? The growth of a network and of a hierarchy affects more structure than that of a sequence. Therefore evolutions cause fewer changes than chronicles, and tours cause fewer changes than canons. A linear factory process will, in the long term, always hit up against a wall where changes in the line become prohibitively expensive because of their consequences on the overlying network structure. When a hierarchy has less structure than a network, then atlases cause less change than catalogs, and when a hierarchy has more structure, then it is the other way around.

Computers could be used broader to model painfully complex situations. The classification of visualizations can help us bear in mind the consequences of structural decisions we make, for example, in organizing companies and societies.

9. RELATING EXTERNAL AND INTERNAL STRUCTURE

The classification of visualizations only considers those that arise from external structurings: sequences, hierarchies, and networks. It is an open question to describe the visualizations that arise from internal structurings.

Several notable concepts relate the internal structuring of capsules, given by field structure, with the external structurings of capsules in terms of sequences, hierarchies, and networks. Relational database systems identify links to records with indexed fields inside of records. Polymorphism identifies a global method with separate implementations within each object. Class diagrams may also relate internal and external structure. They defy classification with their unusually evenhanded treatment of the hierarchy of generalization and the network of association. Perhaps this helps them focus attention on the internal constraints on each class.

10. CONCLUSIONS

It is very useful to deal independently with the internal and external structuring of capsules. Sequences, hierarchies, and networks are three external structurings that together account for visualizations in terms of chronicles, evolutions, catalogs, atlases, canons, and tours.

There should be a standard for the export and import of information into sequences, hierarchies, networks. E.g., one should be able to export a hierarchy of information from Lotus Notes to such a

standard, and then import it to Microsoft Word outline view, to HTML, to a menu driven device, or to a software development platform.

A standard would make possible the serious use of software for organizing thoughts by freeing users from any particular product that may get discontinued, such as Lotus Agenda or NetManage Ecco. Personal knowledge management might then become a significant part of knowledge management.

A standard would encourage the transformation of user interfaces from organizing documents to organizing thoughts. It might lead to interfaces that balance sequences, hierarchies, and networks and present a clearer choice of mental pictures.

A standard could help networks of devices accomodate humans, who excel at applying partial understanding to aggregates of information. Devices should not have to rely on a single unifying language, such as JAVA, but rather be able to act on sequences, hierarchies, and networks of information to the extent that they can identify and process the relevant portions. We presented this vision on April 12, 1999 to the Infrared Mobile Computing work group of the Infrared Data Association.

Finally, a standard would be a practical first step towards a Unified Knowledge Language [12] for representing and transmitting knowledge. With this intent we submitted the main ideas of this paper for discussion at the first standards meeting of the Knowledge Management Consortium International on January 29, 1999 in Washington, DC.

ACKNOWLEDGEMENTS

Our main ideas for this paper were inspired in conversations with Kestas Augutis, Raimundas Vaitkevicius, Edmundas Kulikauskas, and Roy Roebuck. We thank the Mathematics Department of the University of California at San Diego for making its computers available for writing this paper.

REFERENCES

[1] Kestas Augutis, Effective use of a computer at school, first place entry for the contest. Open Society Fund - Lithuania, Vilnius, 1997. (in Lithuanian).
[2] Roy Roebuck, A Unitary/Ecological Approach to Information and Information Technology, 1998. http://www.one-world-is.com
[3] Arthur Markman, Knowledge Representation. Lawrence Erlbaum Associates, Inc., 1998.
[4] M. Mont, T.P. Moran, Study of authoring process in NoteCards. ACM SIGCHI Bulletin, 18(2), 1986, pp. 59-60.
[5] Micro Logic Corp., Info Select 5.0, 1999. http://www.miclog.com
[6] Natrificial Software Technologies, TheBrain, 1999. http://www.thebrain.com
[7] Minciu Sodas Laboratory, Tools for Thinking, 1999. http://www.ms.lt
[8] Andrius Kulikauskas, Will computers help us concentrate ? InfoBalt Laikas, Nr. 3, InfoBalt Association, Vilnius, Oct. 1998, pp. 38-40. (in Lithuanian).
[9] Minciu Sodas Laboratory, Structures for Thinking, 1999. http://www.ms.lt
[10] Martin Fowler, Kendall Scott, UML Distilled, Applying the Standard Object Modeling Language. Addison-Wesley, 1998.
[11] David A. Taylor, Object Technology: A Manager's Guide. Addison-Wesley, Second Edition. 1997.
[12] Knowledge Management Consortium International, Project 3: Standard on UKL (Unified Knowledge Language) Artificial, 1999. http://www.km.org/Standards/ukl.htm