Hypertext as a Tool for Information Gardening for Legal Applications

MARC NANARD, JOCELYNE NANARD

1. Assisting Information Gardening

In the legal domain, activities deal with huge and growing amounts of information which are mainly handled as documents. But unlike technical documents, legal documents contain subtle, informal, non-explicit aspects of information that cannot be neglected. So many legal tasks are still difficult to be computer-aided.

Currently a lot of techniques are used for legal applications, especially in fields where a preliminary modeling of information is possible. On the one hand, databases and information retrieval are concerned with storing and retrieving large amounts of legal documents. For instance, automatic transformation of SGML documents [Wilson 1990] as well as indexing and clustering techniques are used to produce hypertext databases allowing querying as well as browsing. On the other hand, a lot of works take advantage of A.I. techniques for building specific legal expert systems and take advantage of computer assisted reasoning. All of these techniques suppose that a sound knowledge acquisition process has been achieved in order to build a domain model and an application model [Breuker 1991].

Conversely, a lot of intellectual work has to be done without the help of preexisting models and, as a consequence, do not benefit of efficient computer assistance. Barristers, for instance, need often to build up possibly different interpretations of document materials, to study their interdependencies, to compare them, gather and classify information, produce specific documents from existing ones. Drafting tools and ideas managers are sometimes used to help such activities but they are independent tools more dedicated to document preparation than to knowledge elicitation. Hypertext systems, especially those integrating an interactive authoring environment, can assist such tasks in a more complete manner, bridging the gap between source documents and the production of new information because of their ability to incrementally handle knowledge elicitation.
The purpose of this paper is not to describe or propose yet another hypertext tool for the law domain but to examine at a general level why and how hypertext tools can assist the kind of intellectual work identified as information gardening which radically differs from information mining and manufacturing [Bernstein 1993]. We show that the classical hypertext paradigm must be enlarged and replaced by a more functional view in order to support this kind of activity. The different metaphors of mining, manufacturing and gardening/farming are situated in this new framework. We then examine how gardening/farming is supported by current systems. Finally, we show, as illustrated by the MacWeb system we have developed, how hypertext tools should contribute to make interpretation structure emerge from documents thus providing means to exploit information from these documents.

2. The Hypertext Paradigm Revisited

2.1. What is the essence of hypertexts?

Beyond the initial idea from Bush [Bush 1945], the hypertext notion has largely emerged from very different hypertext applications or systems. Thus, whereas a lot of computer applications as different as an interactive encyclopedia on CD-ROM or a writing environment are considered as hypertext applications, a unanimously accepted formal definition of the notion of hypertext is not yet really available. Hypertext is a paradigm.

From a naive and external point of view, hypertext systems are considered as document displayers that offer a point and click user-interfaces for navigating in a document space. Browsing from one document to another has become a commonplace. But this is not sufficient to characterize hypertexts.

From an abstract point of view, the link-node paradigm is generally proposed as a basis for characterizing hypertexts. However, links and nodes are complex notions that may correspond to very different entities. Elements as different as Hypercard cards, Guide expandable documents [Brown 1989], KSM pages [Akscyn 1988], composites [Grønbaek 1994], Sepia activity spaces [Streitz 1992], MacWeb virtual documents [Nanard 1993] are kinds of nodes.

1 The study does not concern a specific category of professional of legal domain – such as for instance barristers, attorneys or solicitors-, but any people whose intellectual activity requires structure emergence. Thus, we use the word lawyer with a generic meaning to refer to all of them.
Similarly, hard-wired links, computable links, queries, slots in frames are considered as links. Other approaches based on sets suggest alternative to linking [Parunak 1991]. Furthermore, an evolution towards sophisticated models is observed. Using types on nodes and links and even frame-based approaches [Marshall 1992, op. cit.] are more and more frequently used to handle structure. This complexification makes the hypertext notion difficult to formally characterize.

Tuned in 1990 and re-edited in 1994 [Halasz 1994], the DEXTER\(^2\) hypertext reference model is now the most accepted model. It results from a cooperative effort to unify and generalize the work done by pioneers developers of hypertext systems. It is an attempt to capture both formally and informally the important abstractions found in a wide range of existing and future hypertext systems. Beyond clarifying the distinction between the storage and the run-time aspect of hypermedia systems as well as the connection with external applications handling documents, DEXTER supersedes the traditional node/link model by the notion of components. It emphasizes an implementation independence allowing various data models.

However DEXTER only characterizes the structural aspects of hypertexts and their consequences on hypertext architecture. Whereas it suggests a good architecture for robust implementation as illustrated in the Devise system [Grønbaek 1994, op. cit.] DEXTER only aims at observing hypertexts from the inside. It describes how hypertexts internally work but not the function they play in terms of human-computer interaction. Hypertext are described from a static point of view. The dynamic aspects of structure evolution during a knowledge acquisition process are not the concern of this model.

Thereby, an important question is “what is the role of linking or more generally of establishing relationships between pieces of information presented as documents? What can we do with it?”. Hypertexts need to be observed from a functional point of view.

2.2. Metaphors shaping information management in hypertext systems

Whereas hypertext systems technically rely on handling or using relationships between information pieces, they deeply differ depending on the kind of use of relationships and the style of application which is addressed.

\(^2\)DEXTER is not a researcher but the place where the model was elaborated.
Three main categories of activities can be observed when a human is dealing with information:

- Extracting information from large repositories,
- Organizing existing information for presenting and better accessing it (and even selling it),
- Making information and structure emerge.

This diversity is stressed by Mark Bernstein [Bernstein 1993, \textit{op. cit.}] who proposed a set of three relevant metaphors: \textit{information mining}, \textit{information manufacturing} and \textit{information farming}. These metaphors account for the attitudes and the interests of users when concerned with information. They make clear what kinds of features are suitable in hypertext with respect to the addressed applications.

2.2.1. \textit{Information mining}

Information miners are interested in providing relevant information pieces found within large repositories more than in adding value to information or in creating it. Information is like nuggets extracted for subsequent use. Such approaches are mostly concerned by the cost-efficiency of extraction. They are very close to information retrieval. So, recall and precision are important criteria. Hypertext features improves the efficiency of retrieval: links are used for instance to record explicit relationships asserted by the user on retrieved data, in order to improve the efficiency of later searches [Croft 1989]. \textit{Information mining} is a frequent need in legal applications where, for instance, case law leads to retrieve documents about similar situations.

2.2.2. \textit{Information manufacturing}

Contrary to information mining which is more concerned with the relevance of the extracted nuggets, information manufacturing mainly focuses on ease of access to information by end-users. Information manufacturers refine raw information into high added value products specifically adapted to some end-users need. Hypertexts are well suited tools for presenting refined information. Relationships mapped onto information make closer information pieces and draw highways for accessing them more efficiently in context.

Legal texts are highly complex and benefit from information manufacturing. Relevant information is often disseminated in many documents. Building a rich structure for efficiently accessing them obviously provides
an obvious commercial added value. Hypertext on for instance fiscal texts, laws or decrees are very helpful and boost the efficiency of their users. Relationships between items are made explicit and semantically connect pieces of information, thus allow a reader to get all relevant information in a given context by a few interactions.

All the efforts in information manufacturing consist in designing and building consistent structures dedicated to the intended use of information. In addition to techniques that are suitable to take into account the specificity of non-linear reading, notions such as schema and some design techniques which have proved their efficiency for databases are helpfully adapted to hypertext [Garzotto 1993] and serve to producing large hypertexts. Like in industrial manufacturing, standards are important to preserve the investment for the long term. SGML [ISO 1986] and Hytime [ISO 1991] make it possible to keep the structure (that is mapped onto documents and provides the added value) in a form which is independent of any system and preserve it for the future.

2.2.3. Information gardening/farming

The third metaphor, Information gardening, seems less obvious but is also extremely pertinent. It addresses evolution of information understanding and of structure elicitation. Unlike mining and manufacturing which are interested in information as a product, Information gardening is interested in the evolution - in the growth, in the life - of information and knowledge. Information gardening is fundamentally a mental activity.

2.2.3.1. Understanding the metaphor

Cultivation of information aims at eliciting knowledge implicitly embedded in document sets. Information gardeners make new information and structure grow from existing documents like gardeners make flowers and vegetables grow on soil. They are the real producers of information and knowledge.

Let us detail the metaphor on several points:

- **Information generation**: The gardener starts working on an unconnected document set, as flat as prepared soil. Discovering and expliciting relationships between information makes hidden structures emerge from documents. Links grow on documents like plants on soil. Links, as relationships expression, represent knowledge. But it is necessary to push the metaphor further. Like plants, links have fruits; links help
make information closer; it is well known that making closer two pieces of information may generate a new meaning which is not explicitly present in any of them. Establishing relationships among information generates new information or at least makes appear hidden information. The gardener seeds them again and goes on working on them. This is the fundamental semiotic aspect of hypertext. Thereby, the information farmer typically is an information producer.

- **Maturation of structures**: Like plants need long care before they mature, it is necessary to support the evolution of knowledge structures through inconsistent transient states before they become well thought out. Information gardening addresses the handling and the care of the transient states which are often looked down or at least ignored in many approaches. In this sense, information gardening is directly concerned with the actual mental processes of designers [Na-nard 1994].

- **Enactment in gardening**: Whereas information manufacturers and miners coolly consider information as a material which is to be extracted and transformed without any subjectivity nor intuition, the gardener is more directly concerned with the information life. All actions which make structure change must be clearly reflected and made visible to the user.

Thus, information gardening is concerned primarily by the cognitive aspects of the human relations to structure emergence, mainly that which involve intuition and capture of raising structures.

Some very slight differences exist between Bernstein’s *information farming* and Marshall’s *information gardening* metaphors though there is some continuum between the two activities which share the property of making structure emerge. Information farming may refer to a larger scale work. It may also involve some more collaborative work. The information gardening metaphor relies on similar ideas but emphasizes the user’s individual relation to information and more private enactment in information producing.

### 2.2.3.2. Information gardening in practice

Let us observe some cases of information gardening in legal applications. Many tasks require to work on large amounts of information distributed in numerous documents. The expert has to build up new documents based on existing data. For instance a barrister has to prepare a speech for defense from the various pieces of a brief. The advocate has to build up in his mind a network of relationships between facts, information, events and so on.
He has to make emerge an argumentation structure by eliciting causal relationships between pieces of information and then to map a rhetoric structure for planning the ordering of enunciation of arguments. A simple analysis of the task itself makes it obvious that the essence of such task is the emergence of structure on the initial document set. This activity (often very long) is typically information gardening and needs to be computer-aided.

The problem is to bridge the gap between means for representing information and knowledge and means for dealing with their evolution (Fig. 1).

**Fig. 1**

Information gardening is a computer-supported human activity which leads to organizing information and eliciting knowledge. Thus, two inter-dependent axes are to be considered: knowledge representation – on the formal side – and evolution handling – on the human side –. The major issue concerns the support to the incremental evolution of information and knowledge all along the mental process of organization.

Information gardening must not be reduced to modal logic. Evolution cannot be considered simply as a sequence of stable modal descriptions. Evolution precisely is the set of bridges between descriptions. The addressed problem is not to describe various states, but to help handle their transitions, on the human side.

Both aspects representation and evolution are considered in the present paper.

Information gardening as an intellectual activity, whereas it is sometimes hard, rarely benefits from effective computer aid. Organizing information, producing structure and eliciting knowledge are truly creative activities where delicacy, subtlety, feedback and backtracking are important. Such activities need specific tools and techniques which take great advantage of recent progress in human-computer interaction, knowledge acquisition and representation as well as hypertext.

Currently, drafting tools and ideas managers are helpfully used like electronic scrapbooks but usually are disconnected from the information sources. Thus they only help handling the structure of documents under
writing but do not address the emergence of structure on existing document bases. On the contrary, hypertext technology may bridge the gap.

2.3. A functional view of the hypertext paradigm

In the activity of information gardening, the hypertext network should be used as a rather evolutionary structure that captures the process of thought and collaboration [Bush 1945, op. cit.] and not only the final state of some information structure. As we observed in section 2.1, the DEXTER model abstracts the architectural and logical view of the node/link network that is supposed to be the essence of hypertext. Stepping beyond the simple link-node model was one of Halasz suggestions as new directions that future hypertext systems should support [Halasz 1991]. Thus, we suggest to consider hypertext from a functional point of view as a paradigm:

- A hypertext helps establishing, handling and exploiting complex relationships among information.
- Handled relationships are meaningful for people when interacting with that information.

Consequently, a hypertext may be considered as an observation structure that maps the human-oriented semantic space of documents – possibly not formally explicit – onto a semiotic space suitable to handle them, thus providing a kind of user interface to interact more directly with informal data within documents.

The important point for information gardening is to provide flexible, powerful and direct means for expressing both informal and, when needed, formal relationships which implicitly exist between pieces of information in order to make possible new information grow from existing ones. Documents are the vehicle of rather informal semantics. The aim of hypertext is to deal with informal, changing and incomplete semantics of documents. The true nature of hypertext consists in anchoring some knowledge.

The term “anchoring” is used for denoting the attachment of information to either end of a link. The anchoring metaphor is very relevant. It suggests the flexibility of linking in hypertext: one may drop the anchor of a link anywhere in a document. Anchoring does not need to build a harbour in the document before (i.e. does not need preliminary modeling of the structure nor defining a database schema to handle it)!

The link network does not alter documents. It floats over them and is just anchored in some places. This leads to distinguishing between three kinds of elements: documents which are dedicated to users for reading and understanding their informal content, the hypertext network which is a computer structure that helps formal work on documents (for instance browsing, but also computing) and the anchorage which bridges informal structures to
structure whether informal or formal, computable or not, onto the informal semantic structure of documents (Fig. 2).

![Hypertext diagram](Image)

**FIG. 2**

From a functional point of view, a knowledge-based hypertext results from the anchoring of a knowledge-oriented formal structure (helpful for computation), onto a human-oriented informal semantic space of documents. Relationships between pieces of information are either direct or computable (e.g., gray line) according to knowledge represented by the formal structure. The duality of hypertext allows to use them for browsing from the document side as well as for working on the formal structure side. The duality makes it possible to mix the advantages of both formal and informal aspects. Information gardening aims at helping such structure incrementally grow on documents.

2.4. Interest of the functional view of hypertext for information gardening

Hypertexts do not only provide static structures for presenting information. Hypertext systems may be flexible working environments for supporting the dynamic aspects of intellectual activities. However, an important and often neglected aspect of hypertext is to deal with structure evolution.

formal structures. As a consequence several hypertext networks can be anchored on a same set of documents, like Intermedia's Webs [Haan 1992].

A good metaphor to understand the real nature of hypertext is to imagine a paper document pined on a wall, a transparent sheet of plastic protecting it and a set of colored lines drawn on the plastic for stressing relationships between part of the document. One may draw and write at will on the plastic sheet to enhance relationships between document parts or attach information to them. This does not alter the initial document but makes it interpretation easier. The anchoring is the superposition of information on the plastic and the information in the documents. This double layered organization supports the growing of a structure on the plastic film and its emergence from the underlying documents.

This metaphor is directly implemented in some recent systems such as Dolphin [Streitz 1994] which takes advantage of Liveboards instead of screens [Elrod 1992].

Thus, hypertext should not be considered in terms of data structure but in a more functional manner as a mapping of a formal structure onto an informal one.
The interest of the functional view of a hypertext as a mapping of a knowledge structure onto documents and pieces of information is twofold:

- Explicit the knowledge structure separately from information: this provides means for reasoning both formally and informally about information;
- Manipulate the structure itself: the structure being separated from documents and enacted may support simply its own evolution.

Thus hypertexts appear as convenient tools for recording emerging structure and use it for flexible and intelligent browsing. Direct observation of the structure as well as formal or informal manipulation make appear – or make possible to deduce by computation – facts that are not obviously visible in the informal structure of document contents.

3. The State of Information Gardening with Some Hypertext Systems

As pointed out in the previous sections, information gardening requires to incrementally make knowledge structures emerge and to represent and to anchor them onto documents.

But, it is necessary to observe the management of knowledge simultaneously from two points of view. One is static and is concerned with the power of expression of the hypertext data model. It concerns the representation of knowledge itself. The other one is dynamic. It concerns the human mental process which leads to knowledge elicitation, and its implications on the user interface of hypertext systems. Both aspects are strongly dependent. However, many theoretical works on artificial intelligence have neglected the importance of human factors in knowledge elicitation.

The current section envisions knowledge structure emergence according to the two points of view. It analyses some hypertext systems that may be a posteriori considered as dedicated to information gardening in order to characterize key features for information gardening.

3.1. Information gardening as knowledge elicitation

3.1.1. Help information structuring is not enough

The interest of hypertext for knowledge organization has been stressed very early. Engelbart’s NLS project initially had a very ambitious name ‘Augment’, since it aimed at ‘augmenting human’s intellect’. The first tool
truly used at large scale, KMS – Knowledge Management System – [Akscyn 1988, op. cit.], explicitly announced it within the system name. But, in all of the first approaches, the word knowledge has not the meaning that recent A.I. works use to give it. Knowledge has an anthropocentric meaning. It is more the concern of authors and readers than a computable structure. So, its management is achieved as side effects of consistent information presentation. The proposed mechanisms are driven from data side more than from the structure side. The interaction is centered on information management and presentation, and expressed at the level of the nodes contents. For instance, structuring with KMS is achieved by breaking information into small nodes organized in a hierarchical structure, with some added untyped transversal links.

But such systems aim at organizing data for retrieving it easily, more than at truly representing and handling knowledge associated to data. The duality between information and structure should be explicit, even it is represented informally.

3.1.2. Represent informal structures but also support encapsulation or processing

During the same time, works in A.I. stress on the computability of knowledge and propose techniques for representing knowledge in order to handle it. A similar trend is observable in the hypertext domain. Many systems even among the earliest ones stress on the need for typed links. Notecards links are labeled. So, many people use it for a wide variety of structuring tasks. Notecards graphical browser makes it possible to visually observe the link-node network, making the structure explicit for the user. Several works concerned with representing the structure of argumentation took advantage of this feature.

gIBIS [Conklin 1988], thanks to the graphical representation and edition of structure, is typically dedicated to knowledge elicitation tasks in the argumentation domain based on the IBIS model. At the opposite of KMS, gIBIS is oriented towards the explicit representation of relationships between pieces of information. It provides the user with a graphical view of the resulting structure. But gIBIS structure, dedicated to argumentation representation, is rigid and too restrictive. Furthermore it is mainly dedicated to description and does not provide any possibility of computation on the structure nor evaluation.

Free labeled links, and the flexibility of its Lisp based implementation, make Notecards [Halasz 1987] a flexible environment for handling any
knowledge representations. IDE [Jordan 1989], built on Notecards, takes advantage of added Lisp procedures which empowered it for directly computing some aspects of structure. Nevertheless, such treatments concern more structure generation than reasoning.

Using Notecards for representing argumentation structure according to Toulmin’s model is experienced by Marshall [Marshall 1989]. The micro-structure of Toulmin’s model is easily implemented with typed nodes but the recursive construction of a large scaled argumentation is more difficult to express unless the hypertext system enables to handle sub-nets as first class objects and to interconnect micro-structures. Furthermore, a great friendliness of the user interface is necessary or a least suitable for working efficiently on structure emergence tasks.

3.1.3. Handle complex but also computable and freely specified knowledge structures

Taking advantage of previous experiences with Notecards, Aquanet [Marshall 1992, op. cit.] is specifically developed to address knowledge structuring tasks. For instance, exhibiting an actual argumentation structure on a set of information is an important task for helping decision processes. Such an activity is very similar to those that lawyers are concerned with. The main originality in terms of data structure is to clearly introduce the notion of objects, which are frame-like entities. In Aquanet, basic objects (nodes), and relations (links) are first class objects (i.e. are handled as such and support linking). Composites (sub-structures) are handled as a composition relationship between objects. This important feature allows to easily build complex structures such as in Fig. 3. The expressive power of Aquanet model is similar to those of frame based languages which are used in A.I. approaches for representing knowledge.

Another system, MacWeb, uses an object-oriented approach to represent knowledge. The main originality of this system is to allow an interactive specification of types and even their evolution within a session. This system is presented in details in section 4.

As a conclusion, hypertext fundamental is to map structure on information. Whereas many systems only handle poor or weak structures, some systems take advantage of formalisms very close to those used in A.I. systems for representing knowledge. But very few systems really offer possibilities to run complex computation on such knowledge. In most of systems, the structure, according to the hypertext paradigm, is only used as an added information for the reader and help access information. Nevertheless, some
systems like viki [Marshall 1994] and MacWeb specifically address the problem of freely specifying and representing complex knowledge structures.

In A.I. approaches, knowledge representation is mainly operated by the machine, with the explicit purpose of computing. Conversely, in hypertext, elicited structures are understandable for the user, with the explicit purpose of making new information appear in the reader's mind thanks to the elicited structure. Both features are useful for gardening.

3.2. The role of the user interface in information gardening

In this section, we present some implications on user interface due to cognitive constraints of structure elicitation and describe how they are addressed in systems which are concerned with information gardening.

Representing information is necessary not sufficient for information gardening: the incremental acquisition and reorganizing is the key of the problem. The right structure never rises once; like a plant, it takes time to grow and need to be cared while growing. Eliciting and organizing knowledge always results from a long and iterative work based on feedback and including a lot of evaluations and backtracking. Expecting to straighten such activity and driving the expert like on rails towards a solution is a pure nonsense. Most of failures in computer aided intellectual activities are due to tools rigidity. Supporting transient states of structuring is far more suitable than enforcing consistency unconditionally! The key to the success is not to suggest (or worst: to constrain to use) a pre-defined method, but to take into account the reality of human behavior and to cope with it.

Knowledge elicitation relies on a loop in which data are evaluated, deductions operated and new information generated or extracted. Basically an induction, deduction, abduction cycle is observable in almost knowledge structuring tasks, even when machine learning is used. In all of them a human agent interacts with a representation agent. In the poorest case the knowledge representation agent is a purely passive one like a scrapbook. In hypertext based knowledge acquisition environment, the user interacts with an active agent which facilitates each of these steps, but should let him or her the initiative.

When an expert studies a set of documents, the most frequent sub-tasks are comparing elements, discovering relationships between pieces, deducing consequences of hypothesis, checking them against facts and noting deductions. This activity implies backtracking. Keeping track of the work in a machine supported form empowers the expert’s work. Relationships which are set by the expert keep related information in a vicinity which
facilitates later handling. Assuming the system allows to set and delete relationships at will and to compute on them, the expert operates with an active scrapbook.

To illustrate these different points, the user-interfaces of two major systems Aquanet and Sepia are used as examples.

Aquanet use a spatial metaphor to represent and handle structure (see Fig. 3). Each Aquanet object is a frame which is denoted by its visual

FIG. 3

organizes pieces of information, moves them within the information space like it would be done on a desktop. Their relationships, even the most global ones are caught at a glance due to their placement, assumed the expert is aware of the meaning given to each specific placement pattern. As a consequence, no explicit naming of slots is necessary.

structure. Each slot is assigned a relative location within the frame. Relationships between information are made explicit only by their relative placement. As a consequence, the user 'makes closer' two information to set a relationship between them. The model is recursive since any slot can be an Aquanet object which visually exhibits its structure with the same convention. Aquanet interaction model is strongly unified. Both the expression of a structure and its observation rely on the same paradigm: information proximity and placement.

But, as Marshall explains it in [Marshall 1994, op. cit.] the drawback of Aquanet is its lack of run time flexibility. Aquanet uses a two stages mechanism which separates frames specification from hypertext edition. As a consequence, it is rather difficult to change the types of objects and relationships within a session once they have been used. Like in databases, schema evolution is a real problem. The placement rules have to be previously defined externally to the system. Thus, the user could only combine a
set of pre-defined placement rules associated to already defined schema but is not allowed to define new schema on the fly. This is of little importance when well defined structures - such as Toulmin's - are repeatedly used as basic patterns, but it is a significant drawback to handle complex knowledge modeling.

For information gardening, premature definition of schema should not be enforced. Rather, informal representation of structure is preferable first as well as smooth shift from informal to more formal structure.

Furthermore, Aquanet is primarily designed to address the task of information structuring, mainly its human-computer interaction aspects, but it provides no or little computation on the elicited structures. This has lead to stress on the extendibility of knowledge representation model in hypertexts. A recent work VIKI, developed also at Xerox PARC, takes advantage of Aquanet experience and aims at providing users with a more flexible and extendible knowledge representation environment. It provides the user with a set of basic constructs which makes possible to express complex structures as and when they emerge without needing the preexistence of a schema. VIKI may help discover some constructs.

Another system, SEPIA, [Streitz 1989; Streitz 1992] developed at GMD IPSI aims at helping writing. It does not promote any given method for writing, but supports the writing mental process like it actually occurs in a writer's mind. Any given piece of information can be concurrently observed and handled in several activity spaces which reflects aspects of the author's activity. Each activity space let appear a specific structure according to the specific problem solved in this space. For instance, the argumentative structure among pieces of information in the rhetoric space or the planning space are projected in the argumentation space. Each activity space is displayed in a separate window. Structure is represented like a graph where relationships are drawn like edges connecting pieces of information. The user creates the structure incrementally, creating nodes and drawing links between them according to design objects and operations specific to each activity space. Unlike Aquanet, SEPIA proposes a free placement of information. As a consequence, each relationship is explicitly labeled. The main originality of SEPIA is the separation of interdependent activity spaces which reduce the complexity of the task, allowing the expert to focus only on specific aspects at a given time. The activities spaces are interconnected, thus any changes done in one are automatically reflected in others.

MacWeb also provides users with a user friendly interface which allows both to take advantage of a visual metaphor for observing and editing the structure. This point is detailed in section 4.2. In the three systems, the
interest of using a spatial metaphor has been demonstrated as well as the possibility of using several point of view on information.

3.3. Key points for information gardening

As a conclusion, helping structures that emerge pre-suppose several conditions:

1) Directness of interaction. Directness is a key issue, since no expert would use a system if it is more time consuming than doing the same work without help.

2) Duality information/structure. To support structure emergence, structure must be handled as such, not only as a side effect of organizing information. Many hypertext support this important aspect and allow to directly edit the structure itself, independently of node contents.

3) Perceptibility of structure via spatial organization. Structure is an abstraction, but providing a visual representation helps working on it. The visual metaphor is a strong help to reasoning which is widely developed in humankind. Advocates are used to handle argumentation in speeches. But this is only the final stage, the result, the elaborated form of the work. During the preparation phase of a speech, how many pages of scrapbooks have been covered with geometrically organized information, with arrow connecting information chunks, with diagrams? Information gardening systems aim at providing with simple to use tools for making structure visible, in order both to make it better understood, and to make emerge new parts as consequences of already elicited parts.

4) Reversibility of actions. Backtracking is a natural feature of human’s work and must be supported by computer systems. No expert would blame himself when checking mentally several hypothesis and backtracking when suitable. So, why to refuse it in a computer aided environment? As a consequence, it is necessary to allow late definition of the structures useful in a given situation. Nevertheless, since a lawyer’s work involves repeated situations many helpful structures can be stored and reused at will.

5) From informal to formal. The aim of information gardening is to take advantage of the ‘man in the loop’ to bridge informal to formal. Man is able to intuitively take the best of subtle information often implicit in documents. Exhibiting these relationships make them usable in a machine, which at its turn helps the user to step on.

6) Not only representational means but also computational means. Representing knowledge in a hypertext is a hard job. It is not done only for fun. Making structure explicit allow to compute on it. Going on working
on document sets on which an elicited structure is growing benefits of the initial efforts. The user may run computations on the elicited parts of the structure to check hypothesis, to study possible connections between loosely related elements, to stress structural regularities, and so on. Thus information gardening becomes easier and easier as and when the emerging structure is used as a bootstrap for helping further emergence.

4. MacWeb and Information Gardening

We illustrate now with MacWeb how and why knowledge-based hypertexts are helpful in information gardening tasks.

Although it is basically a test-bed for studying research hypothesis – and thus is used for experimenting on a wide variety of problems –, MacWeb has mainly be designed to allow a free elaboration of hypertext and help structure emergence. This work has been incrementally tuned up over several years by taking into account the feedback of users and trying to better match their actual behavior. As a consequence, many helpful features are implemented for information gardening because their need has been experimentally observed and their usefulness checked. Detailed descriptions of principles and applications can be found in [Nanard 1991, Nanard 1993, Nanard 1994]. The present description focuses on dependencies between design issues and information gardening needs.

MacWeb enables to interactively create and update hypertext networks with typed links and typed nodes. The main design choices and constraints which address the problem of information gardening are classified in three categories: documents, structure and user interface:

Documents issues

- As an authoring environment, MacWeb provides the authors with an integrated editor to create or update documents. It handles structured multimedia documents, with texts, graphics, images, video and sound.
- Documents are also created as result of computations. MacWeb makes it possible to specify how to select parts from existing documents and re-organize them into a new document structure which provides the reader with a new view of information [Nanard 1988].

Structure issues

- MacWeb supports transient states of structure elaboration.
- The structure modeling features and the system are designed to allow extendibility and run-time evolution of the hypertext structure.
The underlying hypertext formalism has a sufficient power of expression to enable knowledge representation.

**Interaction issues**

- The human-computer interaction is very direct, in order to reduce the cognitive load on the user and to speed up system use.
- Hypertext is handled as easily from the information side than from the structure side.
- The structure is not handled as an abstract notion. It is made a real and perceptible object for the user as an editable graph.

4.1. **Overview of MacWeb hypertext engine**

The idea which gave birth to MacWeb is to mix hypertext and A.I. approaches in order to enable computation on the elicited structure. Typed hypertext networks have similarities with semantic networks. Since a hypertext results from the mapping of a structure onto documents sets, why not to take advantage of efficient structures already used in A.I. to represent knowledge?

MacWeb manages *webs* of typed nodes and typed links. Unlike many hypertext systems where a link type is simply a label, MacWeb enables authors to specify the semantics of types. As a consequence, types carry explicit meaning useful both for the reader and for computation. A reflexively bootstrapped approach is used to define types semantics. It gives the expressive power of an object-oriented formalism to MacWeb webs. The foundation is simple: the typed hypertext network is able to represent relationships among information; thus, assuming types be represented by nodes too, relationships between types are described as a typed network. The hypertext network formalism is used twice, once for describing the types structures and once for describing the information structure. This reflexive construct transforms the hypertext network into a knowledge representation structure. The bootstrap is very simple and only relies on three pre-defined node types which have a specific built-in behavior (concepts, relationships and scripts) and on one pre-defined relationship (inheritance). It allows to describe types as classes in the sense of object-orientation, (with classes and instances, slots and methods, multiple inheritance and overriding). Methods code are programs, which source are kept in 'script' nodes. They enable expression of complex semantics when needed.
MacWeb hypertext engine is especially designed to deal with all the transient states of incremental structure elaboration, which by nature are characterized by incompleteness and inconsistency. MacWeb software has been designed to resist to most of contradictions which are natural during incremental structure elaboration steps. It accepts incomplete and inconsistent structures, allows to use undefined types, resists to cycling definitions and so on. All the design relies on permissiveness in order to preserve authors’ initiative.

4.2. Concepts and interaction for structure elicitation

The current section describes the internal mechanism used for representing knowledge in MacWeb.

4.2.1. Factual description of information

The two elements of webs are nodes and links. Due to the inherent duality of hypertext structure, nodes are handled both as structural elements and as information ‘containers’. Each node has a name and a type. Node names as well as node types are user-defined. Naming enables direct access to information; so, important nodes are directly accessible without browsing. Node types define classes of nodes (see § 4.2.2).

Similarly to current systems, most of nodes contain multimedia data. This is useful for legal applications. The expert works directly on documents: for instance scanned images of archive documents, as well as texts, private annotations or drawings, or even audio records. Anchoring is possible on any part of node contents. MacWeb can be used like a basic hypermedia authoring environment.

Some nodes may contain a set of nodes. Such composite nodes are called groups. Any given node may simultaneously belong to several groups, thus allowing non-hierarchical organizations. Groups may be considered as a mechanism for modularizing information as well as for expressing some contexts. This is particularly useful to handle complex sets of information material where a given item has to be accessed in several contexts. For instance an important document can be accessed directly from several groups: the unicity of information is preserved but its access is made easier. Escaping the tyranny of hierarchical arrangement is important for information gardening.

An originality of MacWeb is that a node may identify an abstraction as well as some information material. In the first case, the node content may
be an informal description of the abstraction – or even nothing –. In both cases nodes are handled in the same manner. This makes it easy to link an information to an abstraction (Fig. 4). Let us remark that such hypertext constructs have the same power of expression as facts in an expert system. Thus a user who wants to explicit a fact in a text has just to draw a link from the anchor to an associated abstract node. The use of this feature to represent knowledge in technical documents is detailed in [Nanard 1993, op. cit.].

Fig. 4

A simplified example. How to represent facts with MacWeb. Nodes typed as “Articles” contain the texts of the legal status of our laboratory. The node “Departement” stands for this abstraction. Typed relationships are used to denote facts such as “Article 18 defines what is a Departement”. Such links and nodes are useful to improve browsing, for instance when navigating from an article to another one related by a shared abstraction as shown on the right side. Pointing on an anchor in the text makes a pop-up menu appear, which indicates the link type and the reached node.

Links are explicit relationships between portions of nodes considered as anchors. MacWeb links are oriented, but can be reversibly crossed. Links have always anchors which are any item of a node contents or even the entire node. Unlike nodes, MacWeb links have no names, they only have types. Types define equivalence classes of links and aim at representing the
semantics of the denoted relationship. Unlike many systems which only propose a restricted set of pre-defined link types (e.g. annotation, reference...), MacWeb link-types are freely user defined. For instance, one may define at will a new type 'contradiction' to express this semantic relationship between two documents.

4.2.2. From an informal to a formal structure

The aim of object-oriented specification of types is to enable representing and handling knowledge structures similar to those handled in knowledge representation systems. The interest of structure elicited by information gardening is strongly augmented when computing is possible.

To easily handle structural abstractions in a hypertext environment, it is necessary to provide them with a handy representation similarly to those of any other information handled in the system, in order to unify their manipulation by the user. 'Concept' and 'Relationship' nodes allow to represent and handle abstract notions as and when they emerge. The representation formalism is object-oriented. MacWeb allows to define both classes of node types and of link types: it is important to enable the user to describe the relationships which exist between the relationships, as well as those between concepts.

The node types named 'Concept' and 'Relationship' are pre-defined in the system and have a built-in set of reflexes triggered by the hypertext network editing operations. A 'Concept' node is considered as the representation of the model of the class which has its name. Nodes with this type are considered as instances of this class. Similarly 'Relationship' nodes are used to model link types. Using nodes to represent types makes it possible to use links between them to represent the relationships between types (Fig. 5).

This feature supports the late structure criteria, since instanciation is not considered as an action but as a property which is evaluated at run time: A node type is considered as a class only if there exist in the web a 'Concept' node with the name of that type. Otherwise it is simply considered as a label. Thereby, the user may establish or break it at will the instanciation relationship. Late structure is especially important for information gardening, since the right structure never rises once, but results from an incremental work which is important to support. So, it helps stepping from informal to formal. Relationships between information can be observed informally, in an almost intuitive way by an expert. Their expression as links between nodes start representing them. As and when the expert’s
Defining graphically a concept with MacWeb: inheritance and composition graph. MacWeb uses the hypertext metaphor to describe concepts denoted by nodes. Relationships between concepts are described like links. Specifications are graphically editable. The free placement of concept nodes in the web makes the structure easy to perceive at a glance. The inspector window displays in textual form the current specification of any given class. Observe how is described the attachment of the method “check validity” to the class “a quo avec .clause”.

work enforces the meaning of the elicited structure, the typing can be specified and associated concepts formalized. The absence of pre-defined schema for structuring avoid disturbing the expert natural way of working.

4.2.3. Specifying structure dynamic semantics

Since ‘Concept’ and ‘Relationship’ nodes aim at representing classes, it is suitable to allow specifying the dynamic semantics of classes. This is done by attaching methods to nodes as scripts.

The ‘script’ type has a built-in behavior. It allows the textual part of node contents to be interpreted as a program. The script language ‘Webtalk’ allows two categories of actions, those oriented towards automatic document
production and those oriented towards hypertext network edition. They are detailed in section 4.4.

Scripts nodes may be either independent nodes – they represent named global routines or are linked to nodes. In this case, the associated link name is considered as a ‘method’ name, in the sense of the object-oriented paradigm. When a script is linked to a ‘Concept’ node, this script is directly visible from any node of the class described by the concept node.

Scripts are also used to define reflexes triggered by system events. For instance, this allows to attach a specific behavior to a given node or node class as reaction to events such as node creation, linking or renaming and so on. Thus types are no longer labels but classes with specific behaviors which encapsulate the semantics described in the methods.

Scripts provide the mechanisms which enable any computing on the hypertext network. This point, very important for helping information gardening is discussed in section 4.4.

4.2.4. Reuse by inheritance and overriding

Specifying types as classes would be of little interest if it was not possible to take advantage of inheritance. The link type named ‘inherits’ is pre-defined and has a built-in behavior. It represents the inheritance relationship of the object oriented formalism. So, when a node is selected, the visible methods are those which are either directly owned by itself, or by its actual class, or by any of the ancestors of its class. Multiple inheritance is supported, and the order of method selection is a deep-first one, based on the links priority set on each node.

4.2.5. Conclusion

The design choices of the MacWeb engine provides the user with a flexible mechanism which has the power of object-oriented formalism, but can be handled in a unified manner, as simply as a hypertext web. Interpreting instantiation like a structural property makes possible to handle late structuring and to support incompleteness of definitions and their possible run-time evolution.

Although we focus on flexibility and openness, another important aspect of the object – oriented architecture of MacWeb is classes reuse. From a cognitive point of view, classes and relationships that have been abstracted during the structure emergence process and proved useful, can be reused, thereby keeping track of users’ mental structure.
4.3. The role of the spatial metaphor for interaction directness and structure perceptibility

In this section we present one important aspect of MacWeb user interface with respect to information gardening: the role of the spatial metaphor as a tool to help structure emergence.

MacWeb enables the user to observe and interactively work on the hypertext from the content side as well as from the structure side. The network of relationships between information is given an interactive representation which supports structure editing. Thus, any operation can be expressed either on documents or on the structure view.

To be easily handled, structure must be clearly perceived by the user. The spatial metaphor which provides the user with a geometrical representation of structure, turns an abstract structure into a visible object. The role of geometrically organized patterns as an help to structuring is well known.

MacWeb ‘Web View’ displays the hypertext network as a graph. The nodes locations are defined by the user who may change them at will. From the structure side, expressing a structure is just drawing a graph. A lot of features which address the directness issue, allows to draw and edit complex graphs in a very intuitive way. This feature is important from a cognitive point of view. The user-defined spatial organization helps building a clear mental scheme of the hypertext structure.

Although MacWeb also proposes an automatic placement, most of users prefer the manual placement. Two reasons argue for that: automatic placement are intuition-less and one remembers more easily a personal placement. Automatic placements can only take care of explicit (such as interconnection) or of abstract properties (such as planarity). They cannot take advantage of intuitive perception of non elicited properties relationships which naturally push users towards informal but ‘clever’ placements. Users arrange some unconnected nodes close one to another because of implicit similarities that they feel more than they could argue.

Information gardening aims at supporting informal aspects. Intuition is one of them. Allowing one to express his or her intuition just by a locating pieces of information in an information space is very subtle. It has no direct consequences on the actual structure like a placement in a spatially organized frame base structure would have. It helps enable or prepare later structuring. It just memorize an intuition, no more. Systems which address information gardening must take care of respecting the user’s way of working, but also keep tack of any elements leading to structure emergence.

Free placement of nodes supports structure emergence. Geometrical
symmetries are often used to make apparent similitude or oppositions. Horizontal levels sometimes express hierarchies, and so on. There are no general rules, but the experiments makes clear that users quickly take great advantage of the visual metaphor to organize the information space. The regularities in the placement of nodes playing similar roles makes patterns appear. Such visual patterns are easily recognizable. They allows the user to locate information more efficiently. Similarly, the perception of the geometry of an already built structure helps structural regularities and exceptions emerge.

In order to generalize the spatial metaphor to large webs, MacWeb uses a 'Sky View' which looks like a stared night sky (Fig. 6), in which each node is displayed like a pixel. It allows the book thickness metaphor for accessing large webs. Selections in the SkyView make the web view directly scroll to the selected area. Only large scale patterns are recognizable in the sky view like constellations in a sky. But the global placement of the patterns within the view makes it possible to take advantage of the spatial metaphor even with networks of thousands of nodes. Even when patterns are not clearly discernible, a user at least remembers in which area is located some kind of information, assumed that the spatial organization of the web is consistent. 'Sky View' has become one of the most useful features for direct access to information.

MacWeb displays groups nodes in the same style as the entire web. Groups are specific sets of nodes. Their web views only represent the subgraphs build on these nodes. This feature provides the user with specialized points of view which make it easier to focus on some important aspect of the structure. Thus, building groups is an efficient work technique for

**Fig. 6**

The sky view. It implements the book thickness metaphor for accessing large webs. Selections in the SkyView make the web view directly scroll to the selected area. Only large scale patterns are recognizable in the sky view like constellations in a sky. The name of the document under the cursor is displayed at the bottom. So both global and precise selection are possible. The frame around the cursor sketches the future position of the screen.
scaling to human size large and complex structures. MacWeb also allows to build partial graphs and partial sub graphs of the main web view. Each link type and node type has a visibility flag which can be toggled at will to hide or show significant structures.

We have observed users at work while designing structures. The interactivity and the direct visual feedback of actions allows very efficient trials and errors strategies to make emerge geometrical structures. A given formal structure can be displayed with many different geometry. Each of them enhances some aspects of the structure which become observable at a glance. So, many users are naturally attracted towards expressive placements putting the stress on regularities in the structures. Such placements often put in evidence some irregularities in the structure and make obvious missing relationships. Such interactive work helps structure emerge.

4.4. Computation on the hypertext network

One of the major objectives of MacWeb development was to make computation possible on the hypertext network. The Webtalk language provides basic control-structures usually present in programming languages, procedures and recursivity. As a tool integrated in MacWeb, it allows to evaluate structural properties of the web especially those defined at run time on types. MacWeb scripts are used for two very different purposes which reflect the duality of hypertext: actions on the hypertext network and document production.

4.4.1. Actions on the hypertext network

Any edition command that a end-user can interactively give to the system can be placed in a script. Scripts makes it possible to specify complex and conditional edition operations. For instance automatic generation of specific sub-structures in the web, as well as consistency checks can be specified in scripts.

The use of scripts for reflex triggered computing is very important in information gardening since it allow the user to incrementally tune the balance between without constraints approaches and model driven approaches. When and where strong constraints are suitable on some classes of information, methods are attached as reflexes to the classes in order to automatically trigger the relevant actions, check and enforce security on these information or on structural elements. For instance a script can express how the creation of a given class of links automatically updates other parts
of the structure. Conversely, free edition remains possible on others elements. This approach does not constrain any pre-defined strategy for handling security problems. It provides the user with flexible mechanism for tailoring the environment at will.

Actions on the hypertext network are used for a lot of applications useful when information gardening. For instance, selecting which types of nodes and of links are visible allows to define at run time some observation points of the structure. Scripts language contains queries which can be used for instance to build new substructures from existing ones.

Though the current interpreted version of Webtalk is far from being fast enough to efficiently support complex reasoning programming, it represents typically the kind of computing which should be addressed by knowledge-based hypertexts.

4.4.2. Actions dedicated to document production

Producing documents is the most original feature of MacWeb. New documents are built on the fly as result of script evaluations. They are produced by organizing into a document structure pieces of information selected in nodes according to queries. A script contains a generic model of the intended document structure, defined in a form similar to the SGML markup. The document model specifies, if any, the conditions of omission and of repetition of document elements in terms of properties of the web structure. The content of each element of the document structure is defined as a mix of static parts defined as texts and of variable parts specified by queries. Queries select information in nodes, either entire node contents, either the anchor of a link. Selection is achieved according to a navigational model based on links, taking advantage of links classes and node classes. The result of the script evaluation is the unfolding of the document model structure and its filling with actual values computed from the web state.

As a consequence, the resulting documents are virtual. The actual content depends upon changes operated in the structure or in the node contents. Such scripts are frequently used to gather into a single document pieces of information which are distributed in several documents but that the user should enjoy to see as a whole. A very simple and classical example concerns the text of amendments of a law, which are disseminated in a lot of documents. A script expresses how to reorganize them taking into account their inter-references expressed by links.

Unlike ad hoc applications, this mechanism does not directly provides the user with off the shelf solutions but with a generic tool which enables
the production of tailored sets of helpful documents which reflects points of views of the user on the information space. Collections of scripts often are attached to classes as methods which encapsulate the origin of information. Thus the user only functionally knows how to get the information he wishes. Such virtual documents make the work on the information space easier and easier as and when elicited structure is made stronger. Computing and producing documents taking advantage of the emerging structure is particularly important for information gardening.

Nevertheless, Webtalk, as an interpreted and not optimized language, must be considered only as a prototyping tool developed in the context of research for evaluating the interest of knowledge-based hypertext approaches. Anyhow, scripts are programs which can run any suitable computation. The last version of MacWeb includes hooks which allows to link compiled procedures to MacWeb code and to call them from scripts to improve the efficiency (and mainly the speed) of computing on knowledge, at the cost of some loss of flexibility.

As a conclusion, MacWeb is typically an open authoring environment. It does not enforce any strategy, but provides users with a set of simple basic features which enable to tailor the environment. The choice of an interpreted approach for structure handling allows a great flexibility and makes run-time structural changes easy. Three complementary aspects are important for information gardening:

- Ease of expression of the emerging structure due to the simplicity of the model and to the directness of interaction;
- Efficiency of the evaluation of the structure, due to the spatial metaphor;
- Efficiency of information observation, that the document generation feature boosts.

Conclusion

Computer applications are usually dedicated to very precise tasks. The information gardening metaphor does not characterize a specific task but a class of activities, a style of intellectual work. Designing software for helping not a task but a style of work is a challenge.

The purpose of the software supporting information gardening is no longer to compute a specified result, nor to drive a user along the steps of a method, nor to achieve a given task. It is to delicately help a human at work and to try to disturb him or she as little as possible from his or her
own expertise. Computer-aided information gardening must rely on an open architecture. The software must be as transparent as possible and a flexible and customizable framework that the user dynamically adapts to his needs. The best piece of software would be one that is so transparent for the user that no one remarks it when at work.

This implies radical changes in developer's mentality: most of computer science works address problems which are formally complex but rarely wide and subtle; they are more concerned with rigor than with flexibility. To address information gardening, it is fundamental to understand that the aim is not directly a result, but a human activity. It is necessary to take care of cognitive aspects of intellectual work, in which intuition, bottom-up approaches, backtracking play an important role. Thus, the application must not constrain the user to follow rigid rules which only make applications easier to develop. It is necessary to accept the reality of human's behavior, and to cope with transient states, with inconsistency, with evolution. It is suitable to display on demand information which are more 'views' of information than planned results. Improving the efficiency of humans at work starts with accepting their actual behavior and giving them the ability to design by themselves, in the context of their work, what kind of help they need.

Currently, hypertext authoring environments are very flexible tools which are well suited to handle this kind of intellectual work on legal documents. They bridge the gap from documents to structure, allowing the user to directly and incrementally express the observed or discovered relationships between information, to define the semantics of the emerging structures, and to use them for stepping in the work. Exhibiting relationships is the goal of information gardening. Expressing relationships as a hypertext network boosts the efficiency of the work since the already elicited relationships make subsequent work easier, allowing to better access information, to observe and change at will structures, to compute the consequences of relationships, to make closer distant information within virtual documents.

Computer aided information gardening does not aim at solving one given problem but to help humans take freely and efficiently advantage of their own expertise for problem solving. It is a work booster.

References


