Ma$\mathcal{X}$: An Access Control System for Digital Libraries and the Web*

Elisa Bertino[a] Elena Ferrari[b] Andrea Perego[a]

[a] Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
Via Comelico 39/41, 20135 Milano, Italy
E-mail: {bertino,perego}@dsi.unimi.it

[b] Dipartimento di Scienze Chimiche, Fisiche e Matematiche
Università degli Studi dell'Insubria
Via Valleggio 11, 22100 Como, Italy
E-mail: elena.ferrari@uninsubria.it

Abstract

Digital Libraries (DLs) introduce several challenging requirements with respect to the formulation, specification and enforcement of adequate access control policies. Unlike conventional database environments, a DL typically is characterised by a dynamic subject population, often making accesses from remote locations, and by an extraordinarily large amount of information, stored in a variety of formats. Additionally, protecting a DL does not only means protecting documents that reside at the DL site, but also protecting accesses that the DL subscribers made to external Web documents.

In this paper we present Ma$\mathcal{X}$, a comprehensive system for enforcing access control, specifically tailored to both DL and Web environments. Key features of Ma$\mathcal{X}$ are the support for credential and content-based access control to DL and Web documents, and its full integration with standard Internet rating systems.

*The work is partially supported by the European Community under the EUFORBIA project (IAP 26505).

1. Introduction

Digital Libraries (DLs) introduce several challenging requirements with respect to the formulation, specification and enforcement of adequate access control policies. Unlike conventional database environments, a DL typically is characterised by a dynamic user population, often making accesses from remote locations, and by an extraordinarily large amount of information, stored in a variety of formats. Moreover, in a DL access control policies are often based on user qualifications and characteristics, rather than user identity (for example, a user can be given access to a document only if he or she is older than 18 years). Another crucial requirement is the support for content-dependent access control policies (for example, all documents containing discussions on how to operate guns must be made available only to users who are 18 or older).

Conventional access control models developed for DBMS systems are not adequate to meet the protection requirements of DLs. In traditional environments, access control is performed against a set of access control policies stated by Security Administrators (SAs) or users [9]. Typically, a policy is specified as a triple $\langle s, o, p \rangle$, which states that a subject s is authorised to exercise privilege p on object o. To allow the possibility of specifying more expressive access control policies, we have proposed in [6] an access control model (referred to as the *Milano Model* in what follows) specifically tailored to the protection of DL documents. Among the key features of the Milano Model is the possibility of specifying fine-grained access control policies based on subject credentials and objects content. Additionally, the Milano Model supports policy propagation and exception management as a means to reduce as much as possible the number of policies that need to be specified.

However, it is often the case the DL is distributed and that references exist from the DL to internal or remote documents. Therefore, the problem of access control cannot be confined to the protection of documents local to the DL (called *in-site* documents in what follows). For instance,

consider a DL of a school which is accessed by all the school students. The students, once authenticated to the DL service, can either browse the DL content or request to access a Web page outside the DL (called *external* or *Web* document in what follows). It is thus necessary that Web accesses made through the DL service are regulated by proper access control policies. For instance, students must be prevented to access Web pages with improper content. For this reason, in this paper we propose an extension of the Milano Model able to regulate access also to documents that do not reside at the DL site. The key features of our proposal are that the model provides an integrated protection of both in-site and external documents, and it is almost independent from the type of metadata describing their content. Additionally, the model is compliant with standard Internet rating systems. More precisely, the model we propose exploits the labelling mechanism provided by the PICS standard [4, 8].

In this paper, besides giving the detailed description of our access control model, we describe an implementation of this model, called Ma$\mathcal{X}$ (Milano Access Control System), developed in the context of the EUFORBIA IAP project [1].

The remainder of this paper is organised as follows. Section 2 introduces the Milano Model. Section 3 discusses the implementation of the Milano Model. Section 4 describes the main components of the Ma$\mathcal{X}$ prototype system. Finally, Section 5 concludes the paper.

2. Overview of the Milano Model

In this section, we briefly illustrate the Milano Model with respect to its main characteristics. We firstly describe how objects (that is, any DL or Web document) and users are represented; then we show how these components are used in the specification of access control policies.

We formally illustrate only the object specification, since it is the main innovative part of the model with respect to the one reported in [6]. For a detailed description of the other components, we refer the reader to [6].

2.1. Objects

The Milano Model provides support for both content-dependent and content-independent access control. The content of an object can be expressed in terms of concepts hierarchically organised or keywords belonging to a flat set. In particular, in the Milano Model content-based access control is supported into two different and orthogonal ways:

1. By pre-processing the objects using a document management mechanism (such as, for instance, [7]), which is capable of extracting concepts from documents and eventually organising them into a hierarchy, referred to as *conceptual hierarchy*. In the following, we denote with $\mathcal{CP}$ the set of concepts, and with $\prec_{\mathcal{CP}}$ the conceptual hierarchy. Given two concepts $cp_1, cp_2 \in \mathcal{CP}$, we say that cp_1 is a more specific concept than cp_2 if and only if $cp_1 \prec_{\mathcal{CP}} cp_2$. As we will see in the following, the conceptual hierarchy can be used to exploit a notion of *policy propagation*.

2. By using the labelling mechanism provided by the PICS standard. A basic idea of the PICS standard is to interpose a selection software between the recipient and the online documents, and to label the documents with respect to their content. PICS provides a common format for labels, so that any PICS-compliant selection software can process any PICS-compliant label. PICS content labels can be regarded as a set of pairs $\langle category, value \rangle$, where $category$ is a concept describing the content of the document, and $value$ is a numeric value which quantifies the 'degree' of $category$.[1] The set of categories is usually a flat domain.

Access control policies can be given either on specific objects, by listing their IDs (that is, the URL of a Web document or the path of an in-site document), or on all the objects with a particular content. Moreover, a finer-grained access control can be enforced by authorising users to access only specific portions and/or links within an object.

All these possibilities are summarised by the notion of *entity specification*. Before formally describing the notion of entity specification, we need to introduce the definitions of *conceptual expression* and *label condition*.

The set $\mathcal{COEX}$ of conceptual expressions is defined as follows: 1) each element of $\mathcal{CP}$ is a conceptual expression; 2) if ce_1 and ce_2 are conceptual expressions, then $(ce_1 \wedge ce_2)$ and $(ce_1 \vee ce_2)$ are conceptual expressions.

The set $\mathcal{LC}$ of label conditions is defined as follows: 1) if $cat \in \mathcal{RC}$, where $\mathcal{RC}$ is the set of rating categories as defined in the PICS standard, $v \in \mathcal{RV}$, where $\mathcal{RV}$ is the set of rating values as defined in the PICS standard, and OP $\in \{>, <, \geq, \leq, \neq, =\}$, then cat OP v is a label condition; 2) if lc_1 and lc_2 are label conditions, then $(lc_1 \wedge lc_2)$ and $(lc_1 \vee lc_2)$ are label conditions.

Thus, an entity specification has one of the following two forms:

1. {`ids` | `co-spec` | `label-spec`}.`slot-spec`, where `ids` is a set, possibly empty, of object

[1] For example, the RSAC*i* PICS-based rating system [3] – that is fully integrated in the Microsoft Internet Explorer and the Netscape Navigator – includes four content categories: `Nudity`, `Sex`, `Language` and `Violence`; each of them can be associated with numeric values, from `0` to `4`, which correspond to particular 'degrees' according to which the category describes the document content.

identifiers, `co-spec` is a conceptual expression in $\mathcal{COEX}$, `label-spec` is a label condition in $\mathcal{LC}$, and `slot-spec` is a set, possibly empty, of slot names, or

2. `link-spec`, where `link-spec` is a set, possibly empty, of link identifiers.

2.2. Credentials

To better take into account user profiles in the formulation of access control policies, we associate each user with one or more *credentials*. A credential is a set of properties concerning a user that are relevant for security purposes (example of properties can be the user name, age, sex or nationality). To make the task of credential specification easier, credentials with similar structures are grouped into *credential-types*, organised into a *credential-type hierarchy*.

Figure 1 shows an example of a two-level credential-type hierarchy, suitable for a high school context, where `PERSON` is the root credential-type, whereas `TEACHER`, `STUDENT` and `ADMINISTRATIVE STAFF` are its children.

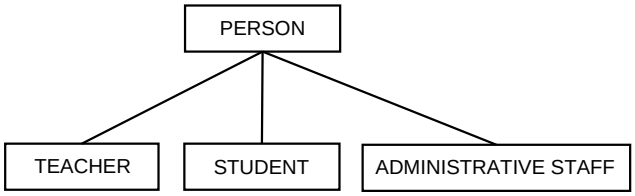

Figure 1. An example of credential-type hierarchy

The Milano Model provides a formal language to express conditions on user credentials (referred to as *credential expressions*) into the access control policy specification. For instance, by means of this language it is possible to specify a policy that applies to all the users of a given nationality, or to all the users whose age belongs to a given range. This allows a flexible specification of policies based on the qualifications and characteristics of users, in addition to the user identity. In the Milano Model, access control policies can be given either explicitly to users, by specifying their identifiers, or implicitly by imposing a set of conditions that the user must satisfy.

2.3. Privileges and Access Control Policies

The Milano Model supports *browsing* and *authoring* privileges with various subtypes within each privilege type. These privilege types subsume the conventional privileges such as read and write. Additionally, we provide support for the specification of negative as well as positive policies, where a positive policy expresses a privilege, whereas a negative policy expresses a denial.

Thus, in the Milano Model, access control policies are represented by a tuple (`crd-spec`, `ent-spec`, `priv`, `sign`) where `crd-spec` is a credential specification denoting the users to whom the policy applies, `ent-spec` is an entity specification denoting the contents, objects, and/or the slots or links to which the policy refers, `priv` is the privilege for which the policy is granted, and `sign` $\in \{+, -\}$ indicates whether the policy is positive ($+$) or negative ($-$).

For example, the policy specified by the tuple (`STUDENT`(X), (`sexual activity` $\vee$ `sexual violence`), `view-all`, `-`) prevents all the users associated with a credential of type `STUDENT` to access objects whose content is related to the concept `sexual activity` or the concept `sexual violence`, whereas the policy specified by the tuple ((`PERSON`$(X) \wedge X$.`age` $\geq$ `18`), (`activity` $\vee$ `violence`), `view-all`, `+`) authorises all the users associated with a credential of type `PERSON`, whose age is greater than or equal to 18, to access objects whose content is related to the concept `activity` or the concept `violence`.

The fact that concepts and credential-types may be hierarchically structured is exploited to support a notion of policy propagation which provides a means to reduce as much as possible the number of policies that need to be specified. More precisely, policies propagate down in the hierarchies in that a policy given to all the users with a credential of a given type propagates to all the users whose credentials are of a subtype of the credential-type which appears in the policy, and a policy which applies on the objects containing a given concept cp implies an analogous policy on all the objects containing a concept more specific than cp.

Policy propagation is very useful since it is a means to concisely express a set of related policies. However, the Milano Model allows one to specify exceptions with respect to how policies propagate along the hierarchies by providing support for both positive and negative policies.

Although negative policies greatly augment the expressive power of the model, they have the drawback of introducing potential conflicts, since a user may simultaneously hold both a negative and a positive policy for the same privilege on the same object. To deal with these conflicts, we have defined a conflict resolution policy which determines the prevailing policy among a set of conflicting policies. The conflict resolution policy keeps into account both the conceptual and the credential-type hierarchies, and is based on the concept of *strongest policy*. The idea is that policies specified on lower level elements of the hierarchies prevail

over policies specified on upper level elements. When conflicts are not solved by the conceptual and credential-type hierarchies, negative policies are considered as prevailing.

3. Implementation

The first reference application domain of the Milano Model was the Global Legal Information Network (GLIN) [2, 6], a project originally undertaken by the Law Library of Congress (LLoC). The goal of GLIN is to create a knowledge base of international laws and to make this knowledge base available to member countries from around the world.

Subsequently, in the context of the IAP project EUFORBIA we have developed an implementation of the extended version of the Milano Model we propose in this paper.

The EUFORBIA project is supported by the European Community under the *Safer Internet Action Plan* (IAP) [5]. The main goal of EUFORBIA is to contribute to the production and use of new generations of Internet filtering systems, more powerful and flexible than the existing ones, and easier to adapt to the cultural, political or religious differences. Therefore, the goal is to develop systems supporting a computer-effective description of the semantic content (the 'meaning') of Web documents that could simultaneously be a) very precise and complete in the description of the content of a given document, and b) neutral as much as possible with respect to any specific doctrine, ideology or value system. Moreover, these systems must provide users – both the individual consumers (parents, teachers, media experts, journalists, etc.) and institutional users (private users such as publishers or public users such as NGOs or parental associations) – with software tools able to make use directly of the neutral descriptions to set up filtering policies and filtering schemata according to the most different cultural, political, religious, etc. options.

In EUFORBIA, a complete and neutral description of the content of Web documents is obtained by attaching to them a *conceptual annotation*. Conceptual annotations are standardised descriptions of both a) the objects, notions, characters, events, etc. that are mentioned in the document, and b) the logical and semantic relationships among all these entities.

Conceptual annotations are expressed according to the syntax and semantics of the NKRL (Narrative Knowledge Representation Language) formal language [11], and are built by using an ontology (referred to as *EUFORBIA Conceptual Hierarchy*) which does not rely only on concepts belonging to the content domains usually considered liable to be filtered (e.g., sex, violence, racism). NKRL conceptual annotations (referred to as *NKRL EUFORBIA labels*) are saved in a text file associated with the corrispondent Web documents, and can be retrieved to evaluate the content of a Web document whenever an access to this document is requested.

In the framework of the EUFORBIA project, the Milano Model has been further extended to support NKRL labels, in addition to concepts and PICS labels. This extended model has been implemented in a prototype system, referred to as Ma$\mathcal{X}$, which provides a user management system suitable for both individual and institutional users, and enforces content-based access control to both Web and in-site documents.

4. The Ma$\mathcal{X}$ Prototype System

Figure 2 depicts the architecture of Ma$\mathcal{X}$, which is a Java-based system, built on top of the Oracle DBMS.

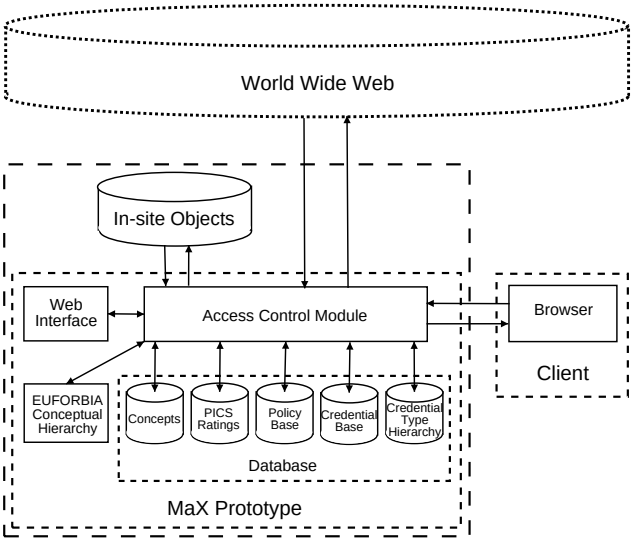

Figure 2. Ma$\mathcal{X}$ architecture

Ma$\mathcal{X}$ is structured into three main components:

1. The *Access Control Module*, which intercepts each access request submitted by a user and verifies whether the access request can be granted or not according to the access control policies specified by the SA and stored in the Ma$\mathcal{X}$ Database.

2. The *Database*, which stores all security information needed by the Access Control Module to determine the answer to an access request, and is organised into five main components: a) the *Credential Base*, which stores user credentials; b) the *Policy Base*, which stores the access control policies specified for both users and credential-types; c) the credential-type hierarchy; d) the concepts associated with in-site documents; e) the ratings of the supported PICS-based rating systems.

3. The *Web Interface*, which provides a graphical environment to perform user authentication and manage the access control system.

In Ma$\mathcal{X}$ access control policies can be specified on objects with respect to:

- the object identifier, that is, the URL of a Web document or the path of an in-site document;
- the sets of concepts associated with in-site documents;
- the NKRL EUFORBIA labels associated with Web documents;
- the PICS content labels embedded in the HTML code of Web documents. In particular, the current version of Ma$\mathcal{X}$ can manage PICS content labels generated by both the ICRA and the RSAC*i* rating services [3].

Ma$\mathcal{X}$ works as a proxy server, and can be used both in a LAN context (installing it on a server of the local network) and by home users (by configuring the Internet connection to use as a proxy a remote server running Ma$\mathcal{X}$).

In the following, we briefly discuss the key features of the Ma$\mathcal{X}$ Access Control Module and Web Interface.

4.1. The Ma$\mathcal{X}$ Access Control Module

The Access Control Module (ACM) is the core of the system. When started, the ACM stands listening to a specific port, not used by other services. Whenever a client configured to use Ma$\mathcal{X}$ starts the browser, the ACM sends the Web Interface login page.

After authentication, each access request submitted by a user is processed by the ACM, which returns the requested object or an access denial message, according to the access control policies specified for the requesting user.

In order to simplify the specification of access control policies, we introduced in Ma$\mathcal{X}$ two 'general' access control settings, `access-all` and `access-nothing`, which are associated with users who, respectively, can access all objects or no objects.

Ma$\mathcal{X}$ does not apply access control policies directly on the NKRL EUFORBIA labels, but on the set of concepts extracted by a software tool from the NKRL EUFORBIA labels, and stored in a text file associated with the Web document. The reference to this file is embedded in the HTML code of the Web document. We use to this purpose the `META` HTML element, assigning the name `euforbia.concepts` to the text file storing the concepts extracted from the NKRL EUFORBIA label.

4.2. The Ma$\mathcal{X}$ Web Interface

Ma$\mathcal{X}$ provides a Web Interface for inserting, editing and/or deleting data stored in the database. The Web Interface is organised into three main components:

- the *Configuration Interface*, by which the SA can build a 'profile', that is, build a credential-type hierarchy and specify attributes associated with credential-types;
- the *Administration Interface*, for the management of the access control system;
- the *User Interface*, which deals with user authentication.

The Web Interface is composed by HTML files, embedding JavaScript code to specify the features of the browser window displaying the interface. The interaction with the database is managed by Java applets loaded by the client browser as objects of the HTML files. This solution satisfies both the principles of cross-platform and usability, since plain HTML files can be managed by Ma$\mathcal{X}$ without any further processing, whereas the JavaScript methods used in the embedded code are 'primitive' and compatible with respect to all the browsers supporting JavaScript. Finally, Java applets work as a client application, thus it is possible to provide an interface whose configuration and management procedures recall those more familiar to end users.

Each page of the interface consists of a title bar, which displays the name of the current page, a menu, which allows the user to navigate through the interface, and the body of the page, which displays the content of the current page.

Assistance is granted by message and alert windows, which open both to prevent users from unintentional errors and to simplify the configuration and management of the access control system; for further and detailed information about the use of the interface, an online help can be opened from the menu.

Figure 3 shows the screenshot of the page of the Administration Interface dealing with the specification of access control policies.

This page is organised into *browsing* and *authoring* components. By using the boxes in the upper part of the form, the SA can browse the credential-type hierarchy (displayed in the *Credential-type Hierarchy* box), and view the users associated with a credential-type and its children. By selecting a credential-type or a user among those displayed in the *Users* box, the associated access control policies are displayed in the *Filtering Policies* box. Additionally, the SA can choose to display a) only the policies explicitly specified for the selected user or credential-type, b) only the policies inherited from upper credential-types, c) both.

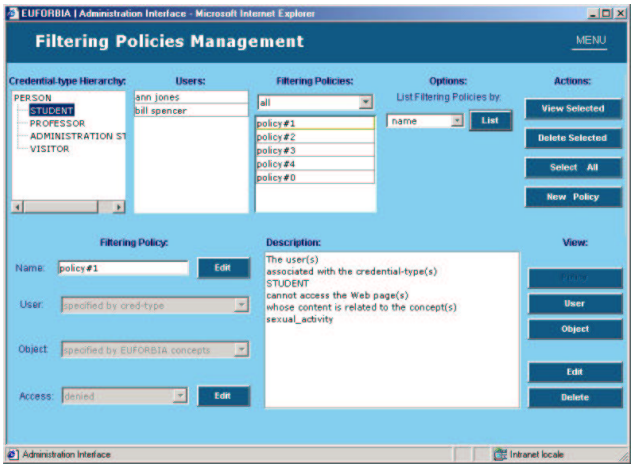

Figure 3. Policy specification page

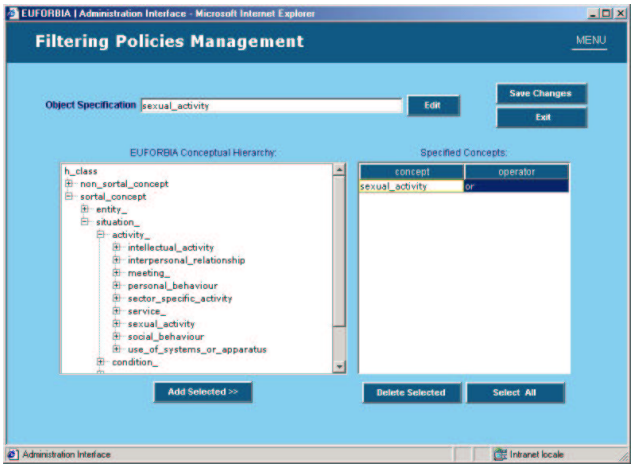

Figure 4. Conceptual expression specification page

The buttons in the upper part of the page activate the main operation that the SA can apply to access control policies, that is: view the components of the selected policy, delete the selected policies, specify a new policy.

By using the *View Selected* button, the components of the selected policy are displayed in the elements of the lower part of the form. The *Name* field displays the policy identifier. The *User* and *Object* fields display, respectively, how users and objects to which the policy applies are specified (that is: IDs, credential-types, conceptual expressions, label conditions). The *Access* field displays the sign of the policy, that is, whether the privilege is granted or denied. Both the identifier and the sign of the policy can be modified by using the *Edit* buttons associated with, respectively, the *Name* and *Access* fields. The *Description* box displays a NL description of the policy. The buttons on the right allow the SA to select the part of the filtering policy which he or she wish to view, delete or modify.

By using the *New Policy* button, the SA can start the policy specification procedure, which is structured in a series of steps. Firstly, the SA must assign an identifier to the policy. The second step is to choose the credential specification type. Then, the SA must choose the entity specification type, and, finally, he or she states whether the policy is negative or positive.

During this procedure, the page dinamically changes in order to supply a different form for both the credential and entity specifications. Figure 4 shows the screenshot of the form provided for the specification of conceptual expressions. The box on the left displays the graphical representation of the EUFORBIA Conceptual Hierarchy.

5. Conclusion

In this paper we have presented Ma$\mathcal{X}$, a system supporting the specification and enforcement of access control policies for both DL documents and Web pages. Key features of Ma$\mathcal{X}$ are the support for both content-based and credential-based access control, and the full integration of standard rating systems for Web pages.

In the paper, besides giving a description of the access control model on which Ma$\mathcal{X}$ relies, we have illustrated the architecture and the implementation of Ma$\mathcal{X}$.

Future work includes the extension of the model to provide content-based access control to multimedia objects (such as, for instance, images, videos, and so on) and the development of a plug-in version of the system. This version will be coinceived for a domestic use and will exploit XML [10] for representing the information on users and access control policies.

References

[1] The EUFORBIA project: `http://www.saferinternet.org/filtering/euforbia.asp`.

[2] Global Legal Information Network (GLIN): `http://www.loc.gov/law/glin/glinv1/glin.html`.

[3] Internet Content Rating Association (ICRA): `http://www.icra.org/`.

[4] Platform for Internet Content Selection (PICS): `http://www.w3.org/pics/`.

[5] Safer Internet Action Plan (IAP): `http://www.saferinternet.org/`.

[6] N. Adam, V. Atluri, E. Bertino, and E. Ferrari. A content-based authorization model for digital libraries. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):296–315, 2002.

[7] R. D. Holowczac. *Extractors for Digital Library Objects*. PhD thesis, Rutgers University, Department of MS/CIS, 1997.

[8] P. Resnick and J. Miller. PICS: Internet access controls without censorship. *Communications of the ACM*, 39(10):87–93, 1996.

[9] B. Thuraisingham. A tutorial in secure database systems. Technical report, MITRE, 1992.

[10] World Wide Web Consortium. *Extensible Markup Language (XML) 1.0 (Second Edition)*, W3C Recommendation 6 October 2000. Available at: `http://www.w3.org/TR/REC-xml`.

[11] G. P. Zarri. NKRL, a knowledge representation tool for encoding the 'meaning' of complex narrative texts. *Natural Language Engineering - Special Issue on Knowledge Representation for Natural Language Processing in Implemented Systems*, 3:231–253, 1997.