IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITB8081

Chapter XV

Information Models for Document Engineering

James A. Thom
RMIT University, Australia

INTRODUCTION

Software engineers develop an information model in the systems analysis and design process to represent the concepts, specification or implementation design of a software system (Fowler and Scott, 1997). This information model is designed using a modeling language such as the Unified Modeling Language (UML) defined by Rumbaugh, Jacobson, and Booch (1999). The software is implemented by translating the information model into code. Similarly, data engineers develop an information model in the database design process to represented the types of data to be stored in a database. This conceptual information model is typically defined using one of the semantic data modeling languages (Hull and King, 1987) such as Entity-Relationship diagrams (Chen, 1976), or NIAM conceptual schemas (Leung and Nijssen, 1988). The database is implemented by translating the information model into a database schema (defined using an implementation data model such as the relational data model or an object-oriented data model). Likewise, document engineers will develop an information model when designing the structure of a collection of documents. This information model will be implemented by translating it into a document schema.

Traditional database information modeling has dealt with structured data such as that found in relational databases. However, much of the information produced using and stored in computers involves documents that do not contain data with a fixed structure - rather it is referred to as semi-structured data. The need for better modeling of documents is no more apparent than in the rapid and chaotic development over the last few years of the World Wide Web. In response to this need, various information models have been proposed to model the semi-structured data found in documents.

Information models useful in document engineering, must be able to represent documents as they move through the different stages in the lifecycle of a document as described by Wilkinson et al. (1998). Document *production* includes the conversion of existing documents in different formats, the creation of new documents, and the editing of existing documents. Document *publication* involves building the access paths that people need to use to find documents; this may require methods for registering, indexing, versioning and storing of documents. Document *discovery* is the process whereby users' information needs are satisfied by finding documents that have been published. Document *delivery* includes the mechanisms for delivery of the document to the user in a form they are able to use. Document *removal* includes both deletion and possible archiving of documents. Document *control* is required to manage the whole lifecycle of documents and includes business processes, workflow, and central document management.

Wilkinson et al. (1998) view documents as essentially messages, their chief characteristics being that they have content, structure and meta-data. To represent documents, a document description language must be chosen that supports these characteristics of documents. Many document description languages have been used over the past couple of decades including simple text (ASCII and Unicode), proprietary languages (such as RTF, OLE, PostScript and PDF), as well as open standards for markup (such as SGML, XML and HTML). XML (Bray et al., 1998) is a document description language that easily represents the content, structure and meta-data associated with documents and can be used throughout the lifecycle of a document. XML is becoming a very widely adopted document description language as document computing enters the new millennium. Indeed, XML has wide applicability beyond ordinary documents; XML is being used in many different contexts for interchange of data.

This chapter describes how information models can be used as a basis for designing the structure of XML documents. This approach can be extended to other information models. Several of these models are briefly reviewed with respect to their capacity to represent the characteristics of a document (content, structure and meta-data) in order to handle different aspects of the lifecycle of documents.

DOCUMENT MODELING USING ENTITY-RELATIONSHIP DIAGRAMS

We use the following simplified example of publishing information about a university department. We assume that for each subject the department would want to publish, the subject number, the subject title, the description of the subject that may comprise several paragraphs and the list of prerequisites. For each lecturer we assume that the department would want to publish their name, phone number, office location and the list of subjects they are currently teaching. We assume that lecturers are either academics or graduate students, and that academics may supervise several graduate students.

A traditional relational database approach would be to begin with an Entity-Relationship diagram or schema (Chen, 1976) as shown in the figure below (using the notation and extensions described by Elmasri and Navathe, 1994).

Challenges of Data Management in Always-On Enterprise Information Systems

Mladen Varga (2010). *Information Resources Management: Concepts, Methodologies, Tools and Applications (pp. 443-462).*
www.irma-international.org/chapter/challenges-data-management-always-enterprise/54494