

---

# XER - Extensible Entity Relationship Modeling

Arijit Sengupta  
Sriram Mohan  
Rahul Doshi

## Abstract

XML is one of the premier formats for data representation and interchange. Many organizations are starting to store data in XML and using XML as an intermediate format for publication and use of these documents. Most database systems have support for storing data in XML or internally representing XML data for storage. However, XML does not have a suitable mechanism for intuitively creating a conceptual model for the data and cannot automatically or semi-automatically generate the schema for the actual data storage. The area of designing conceptual modeling techniques for XML is still not adequately explored in literature. In this paper we describe the XER (Extensible Entity-Relationship) model, a conceptual modeling approach that can describe XML document structures in a simple visual form reminiscent of the ER model, and has the capability to automatically generate XML document type definitions and schema from such structures.

## Table of Contents

1. Motivation .....	1
2. State of the Art .....	2
3. XER — Extensible Entity Relationship Model .....	2
3.1. Modeling Nuances in XML .....	2
3.2. XER Constructs .....	3
4. Translation Between the Model and Implementation .....	7
4.1. UP-Translation from DTD .....	7
4.2. UP-Translation from schema .....	8
5. Down-Translation from XER .....	8
6. Implementation of XER .....	8
7. Conclusion and Future Work .....	9
Bibliography .....	9

## 1. Motivation

Design is one of the most important steps in the software development process [SDLC]. In most software engineering literature, the process of design always comes before implementation, and carries a major weight in determining the success of a project. The process of conceptual design is the phase of design that is independent of the final platform and the medium of implementation, and is usually in a form that is understandable and usable by managers and other personnel who may not be familiar with the low level implementation details, yet have major influence in the development process. Most of the development methods utilized in today's software development include such a conceptual design phase. For instance, in relational database development, the conceptual design is presented using Entity Relationship (ER) diagrams [ER], and in software model development, the conceptual model is presented using data flow diagrams (DFD's), and in Object-Oriented design, the conceptual model is presented using the Unified Modeling Language (UML), and so forth. Unfortunately, the area of conceptual design with XML has not been explored significantly in literature or in practice. Most XML design processes start by directly marking up data in XML, and the metadata is typically designed at the time of encoding the documents.

In order to apply the lessons learned from standard Software Development Life Cycle (SDLC) process, a method for incorporating conceptual modeling in the design and development of XML applications is highly desirable. In this respect we present XER (pronounced “Sher”), an extension of the well-known ER model [ER] , which is capable of handling all the nuances of XML in a highly presentable graphical form.

## 2. State of the Art

Conceptual modeling of XML data is an emerging area of research. Literature shows several potential modeling applications for XML. Mello and Heuser [Rule] use a rule-based conversion technique to convert a DTD into a canonical conceptual model using a semi-automated rule-based process. Conrad, Scheffner and Freytag [XML/UML] utilize a subset of UML for DTD based XML conceptual modeling. Psaila [ERX] introduces ERX (Entity Relationship for XML) as a conceptual model based on the entity relationship model that copes with the features of XML. ERX however does not support multiple features of XML such as Mixed Content etc., and does not describe how complex types with their various nuances can be modeled into the system. Several commercial tools provide support for graphical manipulation of XML structures. However, such tools have their own proprietary methods for graphically designing XML structures. For instance, XML Authority [XML Authority] from Tibco Solutions as well as XML Spy from Altova [XMLSpy] provides a visual representation of a DTD or a XML schema using two different views, a tree representation and a tabular representation listing the various elements and attributes of the schema or a DTD. The Near and Far Designer [NFD] is a DTD modeling tool to create a DTD without prior knowledge of the DTD syntax, and it provides the user with a simple easy to use tree representation to create a DTD. Microsoft’s Visual Studio .NET product [.NET] includes a graphical XML schema editor that uses connected rectangular blocks to present an overview of a schema, although most of the details are hidden in dialog boxes.

Although some research has been performed on conceptual modeling for XML, and XML tools provide support for graphical editing of XML structures, none of the tools or models have the versatility of the ER model. In this paper, we demonstrate that the ER model can be adequately expanded to handle the nuances of XML, and the resulting models are as presentable as the original ER model for relational databases.

## 3. XER — Extensible Entity Relationship Model

We present XER, an extension to the popular ER model [ER] as a conceptual model for XML. To avoid the minute details on all the various features of XML, we are going to assume a canonical view of XML called ENF (Element Normal Form) [ENF], which is a representation of XML documents without using any attributes. Technically, an XML document is in ENF if it does not have any attributes. It is trivial to show that any XML document with attributes can be transformed into ENF by converting the attributes to elements with a special naming convention (e.g., prefixed with a symbol like ‘@’). This transformation can be performed using XSLT [XSLT], and the original document can be obtained back from an ENF representation using XSLT as well. In the rest of the paper, we assume that all the documents in our data set are in ENF. We can now simplify our problem of describing a conceptual model for XML documents to a conceptual model for XML documents in ENF, i.e., with documents with no attributes.

### 3.1. Modeling Nuances in XML

At first glance, the Entity-Relationship (ER) model seems to be an appropriate modeling tool for XML. ER is very successful in the relational domain, with a set of simple graphical modeling constructs like rectangles and diamonds to describe the data objects and the relationships between them. However, XML structures are typically not designed with a data-relationship view, but they are designed more from a document perspective. The main complicating factors that make the standard ER model unsuitable for XML are as follows:

1. XML objects are inherently ordered i.e., there is a specific ordering between different elements as well as different instances of the same element.
2. XML does not have a direct way to support many-to-many relationships, since the structure is essentially hierarchical.

3. XML structures often involve heterogeneous types, a concept in which different instances of an element may have different structures.
4. Individual element structures can be complex, involving structured groups of elements, as well as optional, required and multi-valued elements.
5. An element in XML may have mixed content – with atomic values as well as non-atomic values at the same time.
6. Last but not the least, XML supports many related concepts such as namespaces that would make the task of creating conceptual model non-trivial.

## 3.2. XER Constructs

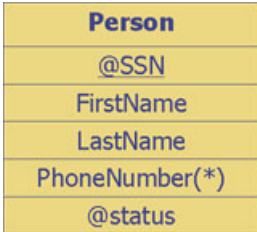
With the above viewpoints in mind, we now present XER (pronounced “Sher”), an extension to the Entity-Relationship model and describe various components of the model. Note that because of the overlap between the concepts in XML and ER, the appropriate terms in the rest of this paper will refer to XER concepts. For example, an ‘entity’ will refer to the concept of Entities in the ER model and not the concept of general or parameter entities in XML. Also, XER attributes will refer to attributes of entities and not XML attributes of elements (we are assuming ENF, thereby eliminating the need for attributes).

The XER model includes all the basic constructs of the ER model, and some new constructs of its own. The primary constructs in XER are described below.

**XER Entity:** The XER entity is the basic conceptual object in XER. A XER entity is represented using a rectangle with a title area showing the name of the entity and the body showing the attributes. XER attributes are properties of entities that are usually atomic, potentially optional or multi-valued. Attributes are shown in the model by placing the names of the attributes in the body of the entity. Attributes are ordered by default, and the ordering in the diagram is top-to-bottom. Multi-valued attributes are allowed, showing the multiplicity in parentheses.

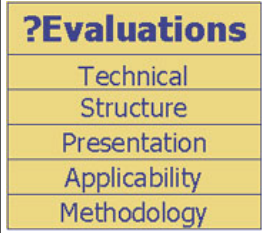
Depending on the type of the schematic element being modeled there are subtle changes in the representation of the XER entity. A XER entity can be of the following types

**A. Ordered Entity:** XER entities are ordered by default. An ordered entity indicates that the attributes in the entity must appear in the same order they are presented in the diagram. XML Attributes (translated using ENF) are shown in the entity prefixed with the symbol @, and the key attribute is underlined. An ordered entity is represented using a tiled rectangle as shown below, the order is from top to bottom:

XML schema	XER
<pre> &lt;xs:element name="Person"&gt;   &lt;xs:complexType&gt;     &lt;xs:sequence&gt;       &lt;xs:element name="FirstName"/&gt;       &lt;xs:element name="LastName"/&gt;       &lt;xs:element name="PhoneNumber"         minOccurs="0"         maxOccurs="unbounded"/&gt;     &lt;/xs:sequence&gt;     &lt;xs:attribute name="SSN"       type="xs:ID"/&gt;     &lt;xs:attribute name="status"/&gt;   &lt;/xs:complexType&gt; &lt;/xs:element&gt; </pre>	


**Table 1. An ordered XER Entity and the corresponding XML schema**

**B. Unordered Entity:** XER also has support for unordered entities, in which all attributes are required but may appear in a document in any order. The unordered entities (as in the <all> tag in XML schema) have the same representation as ordered entities, although the name of the entity is preceded with a question mark (?).

XML schema	XER
<pre> &lt;xs:element name="Evaluations"&gt;   &lt;xs:complexType&gt;     &lt;xs:all&gt;       &lt;xs:element name="Technical"/&gt;       &lt;xs:element name="Structure"/&gt;       &lt;xs:element name="Presentation"/&gt;       &lt;xs:element name="Applicability"/&gt;       &lt;xs:element name="Methodology"/&gt;     &lt;/xs:all&gt;   &lt;/xs:complexType&gt; &lt;/xs:element&gt; </pre>	

**Table 2. An Unordered XER Entity and the corresponding XML schema**

**C. Mixed Entity:** XER supports the mixed entity, in which both text as well as elements are allowed. The mixed entity (as in the “mixed” attribute in the XML schema) is represented in XER using a solid rounded outer rectangle as shown below. In the case of mixed entity with a choice content model, the outer rectangle of the choice entity itself is rounded.

XML schema	XER
<pre> &lt;xs:element name="Para"&gt;   &lt;xs:complexType mixed="true"&gt;     &lt;xs:sequence&gt;       &lt;xs:element name="Bold"/&gt;       &lt;xs:element name="Italics"/&gt;       &lt;xs:element name="Footnote"/&gt;     &lt;/xs:sequence&gt;   &lt;/xs:complexType&gt; &lt;/xs:element&gt; </pre>	

**Table 3. A Mixed XER Entity and the corresponding XML schema**

**XER Relationships:** Relationships, which denote a connection between two or more entities, are introduced in XER when a complex entity contains a complex element as one of its sub-elements. Relationships can be one-to-one, one-to-many or many-to-many. The cardinality of relationships is equivalent to the minOccurs and maxOccurs tags present in the XML schema. In a XER diagram, a relationship is shown with a diamond as in the ER model. Relationships may or may not be named, and labels along the connectors indicate participation constraints for a relationship and the connecting entity. An example of a XER relationship is shown below:

XML schema	XER
<pre>&lt;xs:element name="BOOK"&gt;   &lt;xs:complexType&gt;     &lt;xs:sequence&gt;       &lt;xs:element name="title" /&gt;       &lt;xs:element name="author"         minOccurs="0"         maxOccurs="unbounded" /&gt;       &lt;xs:element name="Chapter"         minOccurs="1"         maxOccurs="unbounded"&gt;         &lt;xs:complexType&gt;           &lt;xs:sequence&gt;             &lt;xs:element               name="title"/&gt;             &lt;xs:element               name="abstract"/&gt;             &lt;xs:element               name="section"               minOccurs="0"               maxOccurs="unbounded" /&gt;           &lt;/xs:sequence&gt;         &lt;/xs:complexType&gt;       &lt;/xs:element&gt;     &lt;/xs:sequence&gt;     &lt;xs:attribute name="chapno"       type="xs:ID" /&gt;   &lt;/xs:complexType&gt; &lt;/xs:element&gt; &lt;xs:attribute   name="isbn"   type="xs:ID" /&gt; &lt;/xs:complexType&gt; &lt;/xs:element&gt;</pre>	<p>The XER diagram illustrates a 1:M relationship between the <b>BOOK</b> and <b>Chapter</b> entities. The relationship is represented by a diamond labeled "has". The <b>BOOK</b> entity (left) has attributes: <u>title</u>, <u>author(*)</u>, <u>@isbn</u>, and <u>chapter</u>. The <b>Chapter</b> entity (right) has attributes: <u>@chapno</u>, <u>title</u>, <u>abstract</u>, and <u>section (*)</u>. The cardinalities are 1,M for the BOOK side and 1,1 for the Chapter side.</p>

Table 4. A XER Relationship and the corresponding XML schema

**XER Generalizations:** The term “generalization” refers to the concept of having an entity that can have different sub-entities (with an IS A relationship). In XER, a generalization is represented using a covering rectangle containing the specialized XER entities as shown below:

XML schema	XER
<pre>&lt;xs:element name="ITEM"&gt;   &lt;xs:complexType&gt;     &lt;xs:choice&gt;       &lt;xs:element name="BOOK"&gt;         &lt;xs:complexType&gt;           &lt;xs:sequence&gt;             &lt;xs:element name="pages" /&gt;             &lt;xs:element name="author"               minOccurs="0"               maxOccurs="unbounded" /&gt;           &lt;/xs:sequence&gt;             &lt;xs:attribute name="isbn"               type="xs:ID" /&gt;           &lt;/xs:complexType&gt;         &lt;/xs:element&gt;         &lt;xs:element name="VIDEO"&gt;           &lt;xs:complexType&gt;             &lt;xs:sequence&gt;               &lt;xs:element name="title" /&gt;               &lt;xs:element name="actor"                 minOccurs="0"                 maxOccurs="unbounded" /&gt;             &lt;/xs:sequence&gt;           &lt;/xs:complexType&gt;         &lt;/xs:element&gt;       &lt;/xs:choice&gt;       &lt;xs:attribute name="itemno"         type="xs:ID" /&gt;     &lt;/xs:complexType&gt;   &lt;/xs:element&gt;</pre>	

Table 5. XER Generalization and the corresponding XML schema

**Other XER Concepts:** Like the ER model, XER can also have weak entities, ternary relationships, and aggregations having similar semantics. Due to space constraints, these have been omitted.

The following figure shows a complete XER diagram combining all of the constructs described here.

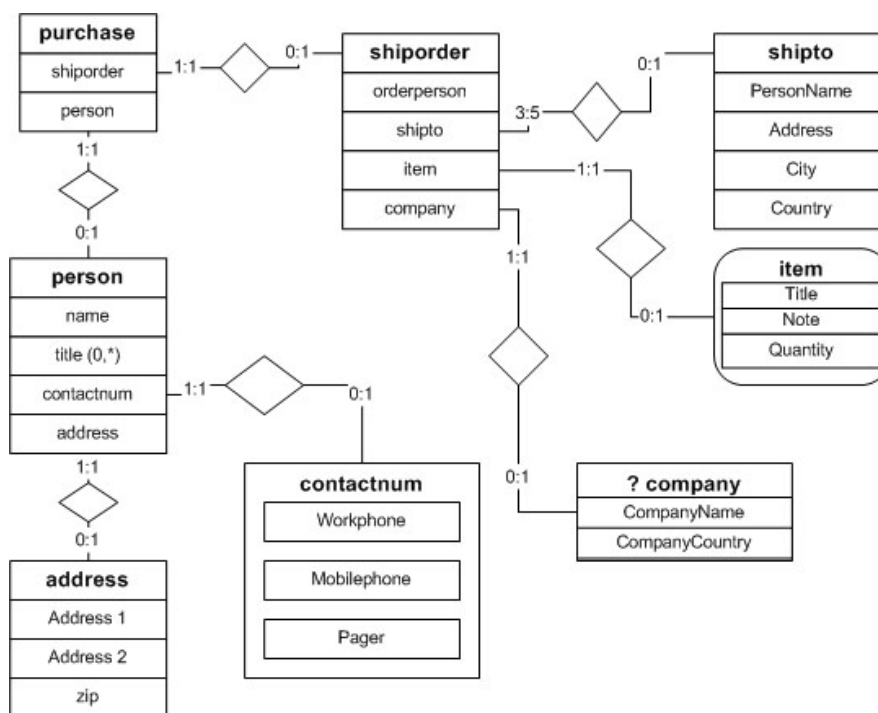


Figure 1. A Complete XER Diagram

## 4. Translation Between the Model and Implementation

Given any XML schema or DTD, an equivalent XER can be constructed, and vice versa. In this section, we are going to describe the basic strategies for translation in both directions. For simplicity's sake, the translation from a DTD or schema to XER would be considered a “reverse translation” or “up-translation”, and from XER to a DTD or schema would be considered a “forward translation” or “down-translation”.

### 4.1. UP-Translation from DTD

Both DTDs and schema can be translated to XER diagrams. The translation process from a DTD to an equivalent XER diagram involves the following rules:

1. All parameter entities are expanded.
2. Every element that is not a #PCDATA (i.e., every composite element) is translated into a XER entity.
3. #PCDATA elements are translated to attributes of the XER entities corresponding to elements that they belong to.
4. A union type is translated into a generalization; an integrated union type may result in an unnamed generalization.
5. Any repeated sub-element becomes a multi-valued attribute if the sub-element is a #PCDATA element. If the sub-element is composite, it is translated into a 1-M relationship. Participation constraints are appropriately assigned depending on whether the element is required or not.
6. Mixed content is handled by making use of a special kind of entity (represented using a solid outer rectangle). An EMPTY content group is translated into an empty (i.e., no attributes) XER entity.



The above rules can translate a DTD into an equivalent XER. Further inspection of ID references can be used to create many-to-many relationships in this translation mechanism as well. We purposely do not handle the ANY content group, since ANY is arguably not a good design principle.

## 4.2. UP-Translation from schema

For the ease of convenience, we only consider schema constructs that are translated into XER constructs. Some of the specialized schema elements are not represented in a XER construct, but kept as properties of the diagram for down-translation. For example, the `<schema>` element includes information about the namespace from which the elements and the data types used in the schema are derived from. This information is kept in the property sheet of the diagram, and can be shown as a footnote in the diagram. The following steps constitute the procedure for an up-translation from an XML schema into a corresponding XER diagram:

1. A simple element becomes an attribute of the XER entity if it's a part of a sequence within a complex element. If the simple element is a standalone element then it's modeled as the attribute of the root element. Data types and other special properties associated with the simple element (such as fixed and default) are built into the model.
2. Since the schema is converted into ENF, all XML attributes are converted to simple elements and the rules for the simple elements will apply while converting the schema into a XER diagram.
3. XML schema supports a lot of restrictions that can be used to constrict the values that the elements of the schema can accept. XER supports all restrictions that are legal in XML schema. Restrictions are handled by specifying them in a separate dialog box called the attributes and restrictions template. Data types are not displayed in XER and are handled similar to restrictions. Cardinalities (min and max Occurrence fields in the schema) are also included as properties of the XER construct.
4. All Complex elements are modeled as entities. A complex element can be of one of the following three types: a) Empty, b) Elements only, and c) Mixed content. The rules for converting mixed content and elements only types are similar to the XER Entity described in the previous section. The empty element is modeled as an empty XER entity.
5. Element groups are handled by modeling the group as an entity according to the rules defined above and then creating relationships between this group and the elements which refer to it. Complex type indicators such as `<choice>` and `<all>` are handled by using generalizations and unordered entities as described in the previous section.

## 5. Down-Translation from XER

The translation from XER to an equivalent DTD essentially reverses the process of the up-translation. In a down-translation, one of the XER entities is designated as the "root" element of the schema, and the down-translation proceeds by performing a depth-first search following the relationships and generating the appropriate schema constructs represented within the XER components. Since all the XER components are unambiguously generated, the reverse translation is also unambiguous. The forward translation incorporates any properties specified in the entity or attribute property sheets, and generates appropriate constraints in the XML schema.

## 6. Implementation of XER

We implemented a prototype of the XER creator using an embedded VBA (Visual Basic for Applications) application with Microsoft<sup>TM</sup> Visio 2002. The VBA implementation of this tool is capable of performing up-translations, conversion of XML schema to XER diagrams and down-translations, converting it back to XML schema. It can also create new XER diagrams and down-translate them into equivalent XML schema.

This tool provides a menu with options for up and down-translations. For up-translation a dialog box pops up and the user can select the schema for converting to a XER diagram. The up-translation process starts from the root element in the XML schema and proceeds in a depth-first fashion. The root element in the XML schema is also



marked as the root entity in the XER diagram. The process of down-translation starts from the root entity and also proceeds in the depth-first fashion while creating equivalent XML schema constructs as described earlier.

This tool also provides an option for adding and displaying properties like minOccurs and maxOccurs to XER attributes. The user is presented with a menu having the option of adding/displaying properties if he/she right-clicks on the corresponding XER attribute. The changes made in the properties window are reflected in the XER diagram by updating the cardinalities in the XER diagram. Initial experiments with this tool have shown the XER tool to be quite robust in handling most types of schema nuances.

## 7. Conclusion and Future Work

XER provides a novel method for providing a conceptual presentation for XML. Typically XML document structures are represented using DTDs or XML schema, both heavily textual formats and often fairly complex to understand from the textual representation. We presented XER, an extension to the ER model to represent all the nuances of XML, and provided mechanisms for forward and reverse translations for going back and forth between the visual and textual views. XER also provides a means for quality assessment of XML data designs. Our prototype implementation is capable of performing all the transformations presented in this paper, and has shown its robustness by preserving the schema structure through multiple up and down-translations. An XML schema, reverse translated into XER, and then forward translated to a new schema generates an exact equivalent of the original.

Designing of a model is only a first step towards the creation of the model. For future research, we are planning experiments that would compare XER with other modeling tools discussed in this paper. We also need to incorporate more intricate XML concepts such as namespaces into consideration in order to be fully XML compliant. However, we are confident that XER can be a highly useful tool in the presentation of XML data models to developers and users alike. In addition, we are working on incorporating support for relationships of arbitrary arity.

## Bibliography

- [ER] Peter Chen, The Entity Relationship Model — Towards a Unified View of Data, ACM Transactions on Database Systems, 1976.
- [Rule] Ronaldo Mello, Carlos Hueser, A Rule based Conversion of a DTD to a Conceptual schema, Lecture Notes in Computer Science 2001.
- [XML/UML] Conrad, Scheffner, Freytag, XML Conceptual Modeling using schema, 19<sup>th</sup> International Conference on Conceptual Modeling, 2000
- [ERX] Giuseppe Psaila, ERX — A Conceptual Model for XML Documents, SAC 2000.
- [XML Authority] XML Authority, Tibco Solutions, <http://www.tibco.com/solutions/products/extensibility/>.
- [XMLSpy] XMLSpy, Altova, <http://www.xmlspy.com>.
- [NFD] Near and Far Designer, Microstar Software, <http://www.xml.com/pub/p/61>.
- [.NET] Visual Studio.NET, Microsoft, <http://www.msdn.microsoft.com/vstudio>.
- [ENF] Andrew Layman, Element Normal Form for Serializing Graphs of Data in XML, 1999.
- [XSLT] James Clark, XSL Transformations XSLT Version 1.0, <http://www.w3.org/TR/xslt>, 1999. W3C Recommendation.
- [SDLC] B.T. Mynatt, Software Engineering with Student Project Guidance, Prentice Hall 1990.

## Biography

### Arijit **Sengupta**

Indiana University, Kelley School of Business  
Department of Information Systems  
Bloomington  
United States of America

Dr. Sengupta is Assistant Professor of Information Systems at the Kelley School of Business and Adjunct Assistant Professor in the Department of Computer Science at Indiana University, Bloomington. Before joining IU, he was Assistant Professor of Computer Information Systems at Georgia State University, Atlanta. He has a Ph.D. in Computer Science and has published numerous papers in the area of database issues of XML, including query languages, database implementation and conceptual modeling.

### Sriram **Mohan**

Indiana University, Department of Computer Science  
United States of America

Sriram Mohan is a Ph.D. candidate in the Department of Computer Science.

### Rahul **Doshi**

Indiana University, Computer Science  
United States of America

Rahul Doshi is a Graduate Student in the Department of Computer Science.