Original Article

PhactaManager: A Clinical Trial Management System Incorporating an XML Layer as a Database-Independent Processing Platform

Okgu Kim[1,2], Yu Rang Park[2], Young Shin Kim[3], Seon Ha Kim[3,4], Hwa Jung Kim[3,4], Ju Han Kim[1,2*], Byung Joo Park[3,4*]

Interdisciplinary Program in Bioinformatics, Seoul National Univ. College of Natural Sciences[1]
Seoul National Univ. Biomedical Informatics (SNUBI), Seoul National Univ. College of Medicine[2]
Seoul National Univ. Hospital Clinical Trial Center[3]
Dept. of Preventive Medicine, Seoul National Univ. College of Medicine[4]

Abstract

Objective: Clinical trials are the most time-consuming and expensive part of the drug development process. Clinical Trial Management Systems (CTMSs) help sponsors of clinical trials manage all aspects of planning, performance, and reporting. Most conventional systems provide data processing functions using database management system (DBMS) procedures, which cause DBMS dependency problems. Thus, it is hard to handle the system by researchers who are unfamiliar with database. It is also difficult to share Electronic Case Report Forms (eCRFs) between institutions because conventional systems rely on specific software. **Methods:** PhactaManager was developed for solving these problems by introducing an XML Layer in the application tier using an Entity-Attribute-Value model in the database tier. **Results:** PhactaManager is a three-tier clinical trial management system that has an XML layer. The XML Layer provides a common DBMS independent eCRF document processing platform. Also we developed XML based eCRF Grammar to describe eCRF documents. The XML data elements described by eCRF grammar was constitute to eCRF by PhactaDesigner which an eCRF document design program. **Conclusion**: We achieved DBMS independency by implementing the XML Layer in PhactaManager. The Development of the eCRF Grammar enables the standardization of eCRF design, data correction and data sharing in multicenter clinical trial. ***(Journal of Korean Society of Medical Informatics 13-2, 99-113, 2007)***

Key words: Clinical Trial, XML, Case Report Form, Clinical Trial Management System, Entity-Attribute-Value Model

Received for review: May 14, 2007; **Accepted for publication:** June 26, 2007

Corresponding Author: Byung Joo Park, MD, MPH, PhD Professor, Department of Preventive Medicine Seoul National University College of Medicine 28 Yongon-Dong, Jongno-Gu Seoul, 110-799, Republic of Korea
Tel: +82-2-740-8325, **Fax:** 82-2-747-4830
E-mail: bjpark@snu.ac.kr

Corresponding Author: Ju Han Kim, MD, PhD, MS Associate Professor of Biomedical Informatics Depts. of Medicine and Psychiatry Seoul National University College of Medicine 28 Yongon-Dong, Chongno-Gu Seoul, 110-799, Korea
Tel: +82-2-740-8320, **Fax:** 82-2-742-5947,
E-mail: juhan@snu.ac.kr

*This research is part of the National Research Laboratory Project M10302000093-03J0000-04710 funded by Korean Ministry of Science and Technology and supported by a grant from Ministry of Health & Welfare, Korea (0412-MI01-0416-0002).

Ⅰ. Introduction

Clinical trials are the most time consuming aspect of drug development prior to market approval. Moreover, the complexity of the trial regimens required for new therapies is increasing. For those who become involved in the clinical trials, it is hard to imagine performing multicenter clinical trials and recruiting hundreds of patients without the help of information technology, and especially of computer systems and the Internet. Clinical trial management systems (CTMSs), which provide a common user interface for entering, updating, managing, and reporting clinical trial data, played an essential role in reducing errors and speeding up processes involving paper-based clinical trials[1-3].

Currently clinical trials involving partial computer support for data entry and management offer much scope for improvement. Some aspects that require special consideration are as follows. Paper CRF design and its CTMS eCRF d esign repeatedly consume time as modifications are made during trial execution, and certain modifications of eCRF designs require dramatic database schema changes. Once clinical trial data have been entered into a CTMS, programming of database procedures is required to prevent logical errors. Moreover, in a multicenter clinical trial situation it is difficult to share a designed eCRF with other clinical trial centers, if they do not use the same CTMS. Most of all, too much time is spent on the education of staff who lack DBMS related knowledge for eCRF design and on programming of validation procedure, even with the help of dedicated DBMS technicians.

Clinical trials are an area where information technologies are actively applied. Information technology approaches can be categorized into two parts. One is the clinical trial process management approach and the other is the clinical trial data management approach. The former approach involves creation of a clinical trial registry[4], subject recruitment[5], subject eligibility testing[6], randomized enrollment[7], generation of reminder[8], adverse event reporting[9], and protocol registration and distribution[10][11], whereas the latter involves research on data entry interface design[12], remote data entry[13][14], data management[15-17], and data retrieval[18]. Improvement in the data management aspect can provide significant benefits to us, because most time is spent in this area. Therefore here, we limit ourselves to the improvement of data management processes.

The following functionalities are needed for data management. Remote data entry is necessary, though distribution and maintenance costs of the client program should be as low as possible. eCRF design must be capable of minimizing the costs of repeated modifications and minimize resulting changes in database schema. This means a clean separation of data entry and data storage is required. Knowledge of the DBMS should be restricted to computer-oriented users.

A number of commercial CTMSs have been introduced and are being used in pharmaceutical companies. In spite of the undoubted importance of CTMS, several drawbacks associated with CTMS have been reported. In most CTMSs, creating new clinical trial requires careful database design in order to capture every detail of the study electronic Case Report System (eCRF), the definition and implementation of data validation rules and consistency checks, the design of data-entry forms and the testing of the database[19]. It makes a dependency between eCRF and DBMS. Strong relationship between eCRF and DBMS makes several difficulties in clinical trial. For instance, modifying or deleting eCRFs, which frequently happens in clinical trials, is difficult due to this relationship. Also users must learn about the DBMS to use data processing procedures or to create their own procedures in commercial CTMS. Due to these characteristics of commercial CTMSs, their reusability, flexibility and interoperability are limited.

As the number of clinical trial using various commercial CTMSs increases, there are increasing demands for comparability and interoperability among clinical trial data from these CTMSs. For solving these problems between CTMSs and providing standard acquisition format of clinical trial data, the Clinical Data Interchange

Standards Consortium (CDISC) offers a set of standards including Study Data Tabulation Model (SDTM), Operational Data Model (ODM), Analysis Dataset Model (ADaM), and Standard for Exchange of Non-clinical Data (SEND)[20]. The Common Data Element (CDE) emanating from the caCORE project defines a set of uniformly data element of clinical trial across institutions and studies funded by NCI, while CDISC provides exchanging standards across CTMSs[21]. Recently, CDISC and Association of Clinical Research Organization (ACRO) initiate the Clinical Data Acquisition Standards Harmonization (CDASH) project to develop a set of "content standard" for a basic set of global data collection fields that will support clinical research studies[22]. The CDASH standard is not available yet. Kuchinke et al reported that CDISC standards are predominantly used for the submission of study data to authorities or the integration of patient data from different sources but not to improve academic clinical research[23]. Also these standards are complex to use by non-computer-oriented user. In this study, we are focused on the development of easily used CTMS.

We currently allocate significant time, cost, and space to the management of paper CRFs in data management centers and multicenter clinical trial coordination centers. We make use of Clintrial 4.2 as a CTMS, which supports client-server communication using Oracle DBMS libraries[17]. While remote data entry is possible using the Clintrial 4.2 client program, its availability is limited because of distribution and maintenance problems. Every data entry site, whether it is local or distant, has to install the Clintrial 4.2 client program and Oracle DBMS libraries, which requires Oracle DBMS technicians at each site. Web-based data entry systems can solve client-side maintenance problems. Inform® supports an integrated environment for developing a web-based remote data entry interface for Clintrial 4.2[13], but still demands much work in the Oracle DBMS because the Oracle DBMS schema has a tight relationship with the data entry interface. The implementation of three-tier architecture for medical information accessing has been described[24], and the implementation of this architecture allowed explicit separation between data entry and data storage. The other commercial systems, i.e. Oracle Clinical, have same problems such as the system strongly dependent on DBMS and hard to manage the system by researchers who are relatively unfamiliar with database management.

The Entity-Attribute-Value (EAV) model, also called row modeling, have been widely implemented in management systems for heterogeneous data sets such as the HELP system[25][26], the Data Repository at Columbian Presbyterians Medical Center[27], OpenSDE[28], and WebEAV[16]. The EAV model stores the value of an attribute as a row with the name of its attribute in another column. This contrasts with the orthodox model where value of an attribute is stored in a column prefixed for the attribute. The EAV model has advantages, i.e., because attributes are not fixed as columns of tables, modifying attributes does not result in physical database schema changes. Therefore, database schema designs can be simplified[15][16][28][29]. Due to the characteristics of the EAV model, several CTMSs was implemented using the EAV model, such as COATI[30], TrialDB[15] and OpenSDE[28]. However, though the systems discussed above address issues concerning generic web interface to database schema problems, they do not present a structured common platform for intermediate clinical data processing independent of database schema. We need a structure that enables eCRF and clinical trial data handling that maintains an independence from DBMS and database schema type.

In the present study, we designed a new CTMS applying XML Layer, which we call PhactaManager. The introduction of the XML Layer provides a DBMS independent common platform. In the XML Layer, clinical trial data are contained in XML-based eCRF document templates before manipulation. Moreover, the XML Layer additionally supports Interface-Lead-Schema-Follow (ILSF) model, where database schema is automatically generated from an eCRF.

PhactaManager is a three-tier CTMS, which is currently undergoing pilot testing. PhactaManager consists of a web

browser (the interface tier), a web server (the application tier), and a relational DBMS (the database tier). Here, we describe PhactaManager. Its overall architecture incorporating an XML Layer for clinical trial data processing is presented first, then eCRF Grammar featuring atomic element and composite element concepts is explained. Next, PhactaDesigner, which is an eCRF document template design program based on the eCRF Grammar, is described. And finally, the database schema required for clinical trial data storing is described with the ILSF model.

II. Materials and Methods

For solving the DBMS dependency problems in CTMS, we adopted the Model-View-Controller (MVC) architecture, three-tier architecture, and document management system with XML technology[31)32)]. The MVC architecture consists of three tiers; an interface tier, an application tier, and a database tier. The interface tier provides a user interface, whereas most of business logic occurs in the application tier, and the database tier stores data. Thus, the MVC architecture separates the presentation of data from the storage of data. A XML document consists of document template and data, both of which come from the database tier. Turau presented a framework using the MVC architecture incorporating XML document technology[30)]. The benefits of a document management system are improved productivity and better access and use of information[32)]. Another paper focused on XML document presentation to mediate heterogeneous data sources[33)]. Assembly of the three architectures introduces a common platform called the XML Layer. Data related operations such as validation and vocabulary coding are performed on XML documents of Model in the application tier.

The designed CTMS system was devised to:

- To implement three-tier architecture where the MVC architecture is located in the application tier. A web browser was the preferred interface tier to minimize distribution and maintenance costs on the client side.
- To devise a mechanism where database schema changes are automatic or minimized when necessary eCRF modifications are made so that non-computer-oriented users do not have to spend considerable time familiarizing themselves with DBMS related knowledge.
- To implement a document management system rather than a data management system so that data processing modules can be applied on documents.
- To devise an XML based eCRF grammar specified by XML Schema to describe documents of Model of the MVC architecture. Moreover, this XML grammar must support reusability, so that parts of the eCRF can be reused in other documents.
- To implement the EAV model so that a large number of attributes can be stored in the relational DBMS without the limited column number problem of the orthodox model.

The metrics used for evaluating PhactaManager were two aspects of the time spent, i.e., the time spent on eCRF design and database schema modification following eCRF design updates. The more reusable and the more independent of the database schema the eCRFs become, the quicker eCRF design and management is concluded. The other time aspect was the amount of time spent on data cleaning. As data processing part, i.e., the XML Layer, becomes more independent of the database schema, validation procedures also become more independent. Therefore, non-computer-oriented users will become familiar with the new system faster than with the existing CTMSs, which demand considerable DBMS knowledge.

III. Results

1. Overall Structure

A conceptual diagram of the overall system structure is presented in Fig. 1. It implements a three-tier architecture, with a web browser as the interface tier, an Apache Web Server 2.0.52 as the application tier, and a MySQL 4.1 relational DBMS as the database tier. The system runs on Linux and Microsoft Windows XP. The Apache Web Server uses a PHP engine for server-side interpreter programming language. For security, the whole system

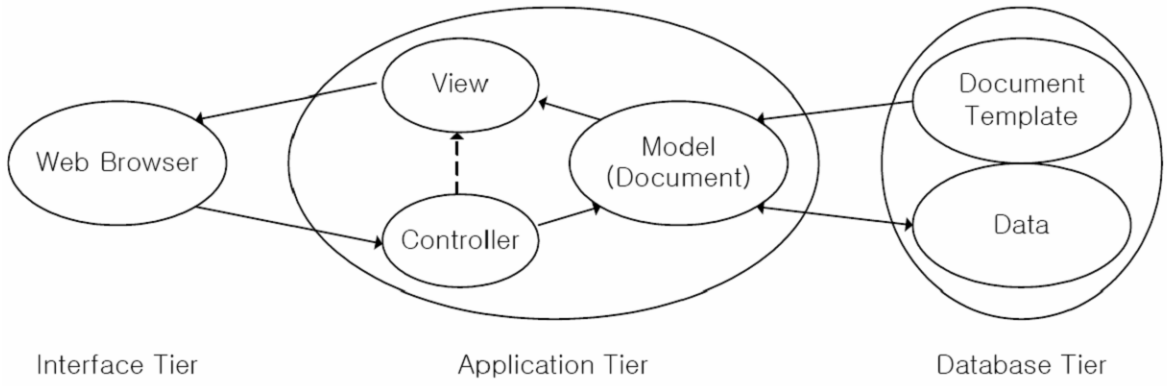

Figure 1. The overall architecture of PhactaManager. Solid arrows represent data flow and the dashed arrow represents command flow. The interface tier is a web browser, which provides an interface for the user. The application tier is implemented using MVC architecture inside a web server. The database tier is a relational DBMS. Model performs document and data abstraction inside the database tier. View translates the document of Model into HTML and delivers then to the interface tier. Controller intermediates View and Model. It also accepts data or requests from the interface tier and manages data of documents inside Model.

resides inside a firewall.

Inside the application tier, MVC architecture has been implemented. View generates HTML page for the interface tier. Controller intermediates between View and Model and accepts data or requests from the interface tier. It also manages Model when updating document in Model. Model is based on documents, which are described using eCRF Grammar, details of which will be discussed later. Documents are made up of document templates and data from the database tier.

We define terms here before we proceed to describe our system further. eCRF is a digitalized paper CRF, and we define eCRF in our system as a data entry interface, a web page, generated by View in the application tier and presented by the web browser at the interface tier. Document is the previous form of eCRF, and resides in Model in the MVC architecture of the application tier. Document template is a document without contained data. Document template has extra information about document, but not clinical trial data. Documents and document templates are described using eCRF Grammar.

Data in the database tier are divided into two categories, inter-study data and intra-study data. Intra-study data are data specific to a clinical trial, whereas inter-study data are common to all studies. Inter-study data include administration information such as system user registration information, access privileges, adverse event coding data such as COSTART, WHOART, and MedDRA, and information used by the system internally.

The EAV Model is used to store intra-study data, contained in the document templates of Model, in the database. The advantages of the EAV model have been described above. Moreover, as the capabilities of the EAV model for clinical trial data management were proven by the pioneering work of Nadkarni, et al[14)15)18)], we implemented this model in a straightforward manner.

Intra-study data are further categorized into two types - basic and complex type data.

- Basic type data are text, integers, real numbers, dates, and Boolean data, which are atomic and account for most clinical trial data. These data types are stored in one table in the database tier. Only basic type data are stored in the database tier using the EAV model.
- Complex type data are those represented by one or several coherent tables. Examples are image data, and X-ray data. For an image to be stored, only one binary column is needed in the table for images, but X-ray data, in some cases, have many mandatory explanatory fields, which are stored in several columns over several tables.

2. Overall Dynamic Modeling

Dataflow starts from the document template stored in the database tier. This template is described using eCRF Grammar. Document templates were designed using PhactaDesigner in the CRF design phase, and registered before clinical trial commencement. PhactaDesigner is an eCRF document template design program based on the eCRF grammar. The detail information of PhactaDesigner described in section E PhactaDesigner. When a user requests a certain eCRF page, Controller in the application tier receives the request, and makes Model generate a document for the requested eCRF page using a document template and data from the database tier. View then converts the document into a HTML page and transfers it to the interface tier. When user updates eCRF data and submits this to the application tier, Controller parses the request and extracts data. Controller then updates the document of Model with the extracted data. Necessary operations, such as validation and vocabulary coding, are performed on the updated document. Finally data are extracted from the document leaving the document template, and stored in the database tier by Model in the application tier. View uses an eXtensible Stylesheet Language Transformations (XSLT) processor to generate a HTML page from document of Model. Model uses a Document Object Model (DOM) processor when it generates a document using data and a document template. It also uses a DOM processor when it extracts data from a document to store the extracted data in the database tier.

3. XML Layer

We define the XML Layer as Model, View, and Controller of the MVC architecture and documents of Model. The modules that manage the documents of Model are on the XML Layer. A layer diagram is shown in Fig. 2. The XML Layer provides a common platform for data processing procedures and allows the application tier to be independent of the DBMS. Therefore, it is the XML Layer that 'hides' the database tier. Clinical trial data are contained in document templates in the XML Layer for document generation.

Data processing procedures, such as data validation procedures and adverse event coding procedures, run in the XML Layer. By moving the platform, where data processing procedures run, from the DBMS to the XML Layer, data processing procedures can be reused regardless of the DBMS type. If document template parts are reused, procedures running on these parts can be also reused without modification. Additionally, the work required to migrate PhactaManager onto other types of DBMS is minimized.

The XML Layer enables the CTMS independent import of bulk data, which we call batch-load. Clintrial 4.2

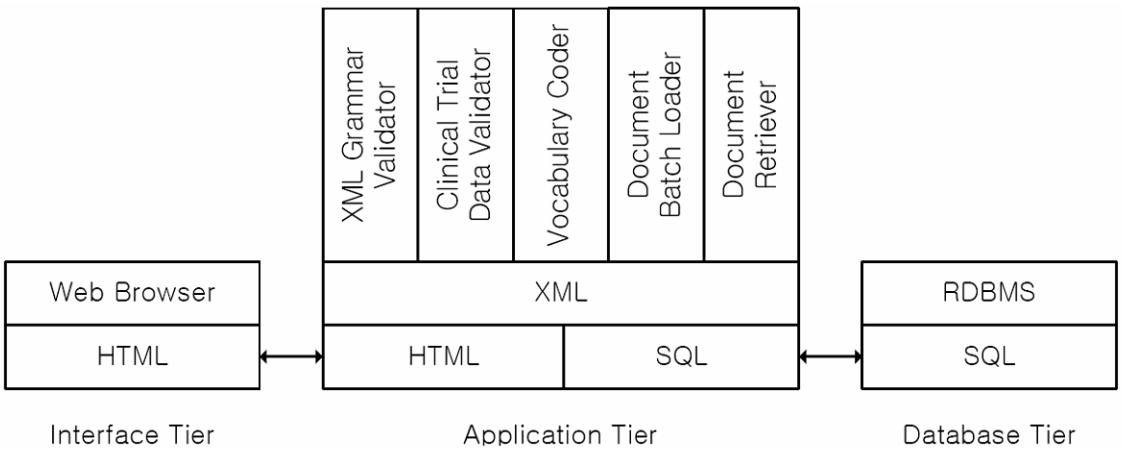

Figure 2. Layer diagram of PhactaManager. Data processing procedures run on the XML Layer of the application tier. The XML Layer communicates with the interface tier through the HTML layer and with the database tier through the SQL layer.

implemented batch-load support using Oracle DBMS batch-load specification[17]. Even Oracle DBMS technicians must process raw data files several times to obtain the final data file for Oracle DBMS batch-load; moreover, the final data file cannot be used for other types of CTMS. However, data contained in documents described using eCRF Grammar can be batch-loaded without modification into CTMS incorporating the XML Layer regardless of the type of source and DBMS. It is also possible to use XML based international standards, such as Clinical Document Architecture of Health Level 7[35] and the Operational Data Model of Clinical Data Interchange Standards Consortium[36] for batch-loading with little effort using XSLT technology.

4. eCRF Grammar

To describe documents and document templates of Model in the application tier, we need a grammar. We chose XML technology to describe documents. Therefore, eCRF Grammar is defined using XML Schema language. XML technology provides the following advantages; it is a self-describing language, which means system designers can define tags; it is supported by several related technologies, such as by XML Schema for the validation of XML documents, and by XSLT for translating XML documents into another types of documents, such as HTML or plain text files; and Simple API for XML (SAX) and DOM specifications can be used to access XML documents. The implementation of these technologies as open source libraries also offers other advantages.

Figure 3 presents an example of an eCRF Grammar based XML description of a document. eCRF Grammar is designed for describing documents and document templates (documents without data) of Model in the application tier. eCRF Grammar is specified in two parts, the document template (the static part of a document), and data (the dynamic part of a document). Document

```
<?xml version="1.0" encoding="UTF-8"?>

<eCRF xmlns="http://www.kospe.or.kr/eCRF"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.kospe.or.kr/eCRF eCRF-3.0.xsd"
        version="3.0" title="CYCLE 0">

<bold><text size="4">Subject General Information :</text></bold>
<table>
        <row>
                <column><text>Birthday</text></column>
                <column>
                        <element name="BIRTHDAY" type="date">
                        </element>
                </column>
        </row>
        <row>
                <column><text>Consent Date</text></column>
                <column>
                        <element name="CONSENT_DAY" type="date">
                        </element>
                </column>
        </row>
        <row>
                <column>
                        <text>Age</text>
                </column>
                <column>
                        <element name="AGE" type="integer" columns="3">
                        </element>
                </column>
        </row>
        <row>
                <column>
                        <text size="3">Sex</text>
                </column>
                <column>
                        <element name="MALE" type="checkbox"/> <text size="3">Male</text>
                        <element name="FEMALE" type="checkbox"/> <text size="3">Female</text>
                </column>
        </row>
</table>

</eCRF>
```

Figure 3. Example of an XML description of a document based on eCRF Grammar. Any document must contain <eCRF> root element. Tags for static parts, such as <table> ,<row>, <column>, <text> tags are shown, as are <element> tags for data input, the dynamic part.

templates are identical, therefore, if two documents are for the same eCRF page relate to different subjects. Only the data they contain differ.

The static part of a document, the document template, is described using presentation tags, whereas the dynamic part used for holding data is described using data elements. The presentation tags are XML tags and are used for HTML output control when translating documents into HTML pages. Most of presentation tags have corresponding HTML tags for clear translation. Data elements are XML tags that can contain clinical trial data. Data elements contain sub-elements for data values and permissible value descriptions. eCRF Grammar has two types of data elements, namely, atomic and composite elements. Atomic elements cannot be divided into lower levels, and contain only basic type data, which is described above in the overall structure subsection. These include text, textarea, integers, real numbers, and checkboxes. When an atomic element has a value, the element contains a child tag that describes the value.

The composite elementconcept was devised to describe paper CRF sections. A composite element is defined as a set of atomic elements, presentation tags, and other composite elements. In paper CRFs, parts can be identified with constituents that are too coherent to be divided. One example is a combination of a checkbox and static text beside the checkbox; because checkboxes cannot explain to data enterers when to check, they must always be combined with static text. In this case, a composite element is used to hold both of them.Composite elements can also be used for larger sections of eCRFs. For instance, a section that has a table, in which there are several checkbox atomic element and static text presentation tag pairs about smoking habits, can be described as one composite element. If needed, a whole document for an eCRF can be described as one composite element in a similar way. Both atomic and composite elements provide a means for the reusability of eCRFs. Commonly used atomic elements and composite elements are described using eCRF Grammar, and are deposited in web sites for future reuse.

To develop eCRF Grammar, we collected paper CRFs of previous clinical trial studies performed at Seoul National University Hospital Clinical Trial Center (SNUH CTC) since 1997. We then identified the data input and CRF decoration parts. The data input part, representing the dynamic part of a document, includes text, integers, real numbers, dates, and checkboxes, and is presented as atomic element in eCRF Grammar, whereas, the decoration part, representing the static part of a document, includes static text, tables, and special symbols, such as degree Celsius and arrows, and is presented using presentation tags in eCRF Grammar.

5. PhactaDesigner

PhactaDesigner is an eCRF design program and provides an integrated eCRF development environment. Because eCRF design is equivalent to empty document template design, PhactaDesigner can be thought as a document template design program. It has three windows; the Edit Window, the HTML Window, and the Output Window. Figure 4 shows a snapshot of PhactaDesigner. With PhactaDesigner, the user edits and validates an eCRF, and transforms it to HTML to view the final HTML output, in which every data element is empty. Usually, each paper CRF is translated into one eCRF using eCRF Grammar. The final set of eCRF files are then registered to PhactaManager for a clinical trial study. Once the eCRF files are registered into PhactaManager, they are preprocessed inside PhactaManager. All composite elements in the files and their sub-level composite elements are decomposed iteratively until only atomic elements and presentation tags remain. After this decomposition step, the resulting document templates are stored in the database tier.

PhactaDesigner is a standalone Java program and internally embeds an XML Schema validator, a preprocessor, and an XSLT processor. XML Schema validator validates the designed eCRFs according to eCRF Grammar. The preprocessor checks for the presence of a composite element in eCRFs and decomposes any

Figure 4. PhactaDesigner, an eCRF design program used for editing, validating, and transforming eCRFs. It consists of Edit Window, HTML Window, and Output Window. Edit Window, the upper larger area directly below the menu bar, is used for editing XML content of an eCRF file. The HTML Window is used for displaying HTML translated from the eCRF XML content in the Edit Window. The Edit Window and the HTML Window are switched using tabs directly below the windows. Only the HTML Window is shown in the figure. The Output Window at the bottom of the figure shows the results of XML Schema validation and XSLT transformation, or errors that have occurred during the validation and transformation procedures.

composite elements iteratively. Finally, the XSLT processor converts the eCRFs with only atomic elements and presentation tags to HTML to show to the user in the PhactaDesigner HTML Window. The XML Schema validator has been implemented using Apache Xerces library 2.6.2 and the XSLT processor using Apache Xalan library 2.6.0.

6. eCRF Display, and Data Input, Processing, and Update

When a user requests a certain eCRF page, the required document for the eCRF page is generated in Model using a corresponding document template and data from the database tier. The document is then translated into a HTML page by the XSLT processor in View. Translation is performed according to the translation information specified in the file used by the XSLT processor in View. Presentation tags are translated to their own matching HTML tags. Atomic elements are translated to input tags in the resulting HTML page. Because all composite elements of document template registered to PhactaManager have been decomposed into atomic elements, no decomposition step is executed. All input tags in the resulting HTML page are contained in one form tag, which appears only once in one HTML page. If an atomic element has a value, the corresponding input tag is filled with that value. When translation completes, the HTML page is transferred to the interface tier, web browser.

After a user enters or updates a value in any input element of the eCRF HTML page, he or she presses the submit button in the page. Values in the input elements of the page are encoded according to Hyper-Text Transfer Protocol (HTTP) and are transferred to the application tier. When the HTTP encoded data arrives at the application tier, they are received by Controller. Because the HTTP encoded data are far from HTML or XML document, Controller parses the data to extract values of input data elements. Names of atomic elements that must hold extracted values can also be extracted from the HTTP encoded data. Resulting values with the name of atomic elements are transferred to Model. Model identifies a document template that must hold all values in the database tier and thus generates a document. The DOM processor of Model then fills the document template with the values.

After the required document has been generated in Model, other required processes are performed on the document. Validity checking of the value of each atomic element is an example. If value of an atomic element must be coded into an adverse event coding system, such as MedDRA, the value is so coded. When all of these necessary processes have been completed, data in the document template are extracted and stored or updated in the database tier by Model. The document template is not sent to the database tier because PhactaManager does not modify it. Document template modification occurs only when a modified document template is registered to PhactaManager.

7. Database Schema for Intra-study Data

Intra-study data are specific to a clinical trial study. Here we implemented the EAV model for intra-study data storing. Figure 5 shows a conceptual table of our EAV model. The database schema for intra-study data consists of one table and extra tables for complex type data and eCRF document templates. We place all columns for the basic type data together in one table called a primary table, which contains many null values. However, putting all basic type data together does not waste physical storage space because null values do not consume space, and because indexes on basic type data columns do not hold the location of null-valued cells. Storing basic type data in the primary table enables a data type conversion with little effort if the conversion is feasible. In our EAV model, data type conversion in the primary table requires only a

Value Columns

Element ID	Element Name	Data Type	Text	Integer	Real	Date	Boolean
PM94302	Subject Name	Text	Kim	/	/	/	/
PM94303	Subject Height	Real	/	/	182.7	/	/
PM94304	Subject Age	Integer	/	30	/	/	/
PM94305	Smoking	Boolean	/	/	/	/	True

Figure 5. A conceptual table of our EAV model. '/' in a cell means a null value. The name of a data element is stored in the 'Element Name' column and the value of a data element in a 'Value Column' according to data type. Data type information is stored in the 'Data Type' column. A table manipulation processor consults the 'Data Type' column when deciding which 'Value Column'to read.

moving value from the source column to the destination column and the updating of the type information of the basic type data. If different types of basic type data are stored in different tables, type conversion results in deleting rows from the source table and inserting rows into destination table. This process places a heavy load or worse on DBMS if many indexes exist on source and destination tables.

The primary table is also used for complex type data. When the database tier stores complex type data, the type information of complex type data is stored in the primary table as a row. Actual complex type data are then stored in tables designated for complex type data, thus different kinds of complex type data are stored in different tables. When the database tier needs to access complex type data, it first references the primary table to determine which tables to read.

8. Interface-Lead-Schema-Follow Model

Most contemporary CTMSs require users to access the database from the clinical trial design step to the data retrieval step. Users create database schema, design user interfaces, and connect the database schema with the user interface. Despite the advantages that a strong relationship between the user interface and database schema provides, i.e., better system performance with fast data exchange and precise data handling, this system type makes it difficult for non-computer-oriented users to learn how to use the system from scratch. It is also difficult to change user interface or types of data elements because these processes require profound DBMS knowledge.

We devised an ILSF model to remove the problems caused by the strong relationship between the user interface and the database schema. Our devised ILSF model runs in the XML Layer. In this model, the user only designs the eCRF as defined by eCRF Grammar and the database schema is automatically generated from the eCRF. This model also enables automatic database schema changes according to eCRF modifications. In cases of basic type data changes, the resulting process has been described in the previous subsection. When a user adds new elements in an eCRF, new rows are added to the primary table when the user inserts new data. When a user deletes elements from the eCRF, corresponding rows in the primary table are deleted. If the deleted elements are of complex type data, rows from the related complex type data tables are also deleted. These automatic schema change functions are available only before the clinical trial starts. The primary table schema is fixed after the clinical trial starts.

9. Data Retrieval

PhactaManager currently supports data retrieval directly from the database tier and document query from Model. For the former, we are confronted with the EAV model problems presented by Nadkarni[15]. Because an EAV model stores values in row direction rather than in column direction of orthodox models, even simple queries containing several columns in the orthodox model can cause a relatively heavy DBMS load due to complex self-joining. Moreover, attempts to solve this problem using DBMS specific features limit our system design objective, i.e., DBMS independency. Direct access to the database tier also violates our intention to distance non-computer-oriented users from the database tier. The document query approach causes the following problems; the document template part of documents becomes redundant. To get rid of the redundant document template part, we need a way of obtaining only data from documents in a new format XML file. Data exchange using XML technology as presented by Marenco offers an example[29].

We are currently developing a new data retrieval module that implements a totally different approach. This approach is based on set theory and our epidemiological research experiences. Case and control group concepts for Intention to Treat (ITT) or Per-Protocol (PP) analysis are defined as a set comprised of sub-sets. The ITT is an analysis based on the initial treatment intent, not on the treatment eventually administered[37]. The PP is a strategy that only patients who sufficiently complied with the trial's

protocol should be considered in the anlalysis[38]. This approach is under implementation.

10. Status Report

Research nurses, epidemiologists, physicians, and coordinators at the SNUH CTC are evaluating PhactaManager for anti-cancer clinical trials, because PhactaManager and the eCRF Grammar were initially designed for anti-cancer drug phase II clinical trials. To date PhactaManager has been found to significantly reduce the time required for eCRF design, maintenance, and clinical trial data management. The average time used to design an eCRF with a database schema design was four weeks, which compares to less than one week using the PhactaManager ILSF model. Moreover, PhactaManager users now concentrate on eCRF design rather than database schema. Currently PhactaManager is being used to manage clinical trials of antibiotics, anti-hypertensives, psychotropic drugs, and NSAIDs with a view towards expanding its architecture.

The time on data cleaning procedure programming is under evaluation. However, because the clinical trials that are being managed with PhactaManager are estimated to take almost three years, it is too early to determine the efficiency of PhactaManager data cleaning support. However, one phenomenon we can identify at this point of time is that as common parts of eCRF are reused for different clinical trials, the same data validation procedures for the parts can be also reused, which ultimately reduces the time required for data cleaning procedure programming.

Though PhactaManager is being run without major problems or the need for modification, we identified the following limitations when it was applied to real clinical trials.

- Commonly used data elements are used repeatedly for the design of eCRFs for different clinical trials. An eCRF Grammar based Common Data Elements (CDEs) management system is needed to improve eCRF design performance.
- Functionalities supported by PhactaManager are limited from the perspective of data management. Support for randomization and eligibility testing for subject enrollment are essential for executing clinical trials, as inappropriate subject enrollment negates the efforts of any subsequent clean and well-managed clinical trial data.
- PhactaManager must support central coordination for multicenter clinical trials. As SNUH CTC manages more multicenter clinical trials, status reporting and standardized data collection will receive more attention.

In response to the limitations described above, PhactaManager and its related supporting technologies are being improved. Standard eCRFs and CDEs are under development using the paper CRFs of clinical trials performed at SNUH CTC since 1997. Hundreds of clinical trial protocols and their paper CRFs provide a mean of improving eCRF Grammar. We are also enhancing its functionalities to support phase II and phase III multicenter clinical trials. SNUH CTC has been conducting nation-wide multicenter phase III clinical trials for global pharmaceutical companies. Multicenter clinical trial related experiences and problems are reported at an annual workshop by the SNUH CTC and are a major source of direction with respect to the prioritization of functionalities requiring development. Randomization, and inclusion and exclusion criteria testing procedures for phase III clinical trials are also being developed.

Biostatisticians prefer using Structured Query Language (SQL) directly at the relational DBMS rather than the document query approach for clinical data retrieval after database locking and data cleaning. Biostatisticians and PhactaManager developers identified the same EAV model problem described by Nadkarni[16]. Accordingly, relational DBMS View technology is used to transiently transform the format from the EAV model to a conventional model. We are currently implementing a set theory based data retrieval approach, in the hope that it offers performance, maintenance, and friendliness for non-computer-oriented users.

IV. Discussion

PhactaManager is a new generation system for clinical trial management. It is a three-tier system and has an XML Layer in its application tier. We have improved architecture and robustness without losing sight of the importance of non-computer-oriented users' opinions. One of the motives behind developing of this system was to enable simply usage by non-computer-oriented personnel. Therefore, we concentrated on developing the system from the point of view of non-computer-oriented users and have used feedback from this customer group. One of the comments made, for example, was that the value validation support of data elements be enhanced at data entry. To ensure that transcription errors and logical mismatches among eCRF data elements are corrected as soon as possible significantly reduces the cost of correction[1)2)]. In addition, users can do most of the data processing work in the XML Layer with little knowledge of the DBMS, which should streamline clinical trial execution.

The insertion of the XML Layer in the application tier enables the standardization of eCRF design and data collection. eCRF Grammar, developed using the paper CRFs of clinical trials performed at SNUH CTC, provides the clinical trial community with a new means of describing an eCRF. The reusability of CDE is also a concern, and we are developing a website for registering, managing, searching, and retrieving CDEs, and standard eCRFs. By centrally managing CDEs or a whole eCRF set as described using eCRF Grammar and registered at the web site, data element names and data element types become standardized; moreover this should pave the way to the sharing of clinical trial data between clinical trial institutions.

PhactaManager is in the first phase of its development cycle. Many parts remain to be developed before PhactaManager becomes robust enough in practice. Especially, we didn't consider the clinical trial standards such as CDISC and CDASH in the current version of PhactaManager. Main objective of PhactaManager is to enable non-computer-oriented user to study clinical trial easily. The CDISC standard is too complex to use for non-computer-oriented user and the CDASH standard is not available yet. In the next version of PhactaManager, we will apply these standards for automatic generating STDM report from eCRF as defined by eCRF Grammar.

Therefore, we are still involved in the identification and prioritization of aspects requiring further standard, research and development. Past research on clinical trial management offers good source material in this context. TrialDB, Clintrial 4.2, and NCI's CDE are examples. Other sources of material are based on our experiences related to multicenter clinical trials involving multi-national pharmaceutical companies. We have formalized and documented experiences of audit and inspection preparation, quality assurance, quality control, and study documentation processes in as Standard Operating Procedure (SOP), and we attempted to identify processes requiring reengineering and automating using PhactaManager. Functionality development prioritization is largely based on practicality, but we also emphasize the importance of research to improve clinical trial execution performance.

References

1. Kush RD. eClinical Trials: Planning & Implementation. Boston, MA: Thomson CenterWatch, 2003.
2. Rondel RK, Varley CA, Webb CF. Clinical Data Management 2nd Edition. West Sussex, England: John Wiley & Sons. Ltd., 2000.
3. Silva JS, Ball MJ, Chute CG, Douglas JV, Langlotz CP, Niland JC, et al. Cancer Informatics: Essential Technologies for Clinical Trials. New York, NY: Springer, 2002.
4. McCray AT, Ide NC. Design and Implementation of a National Clinical Trials Registry. J Am Med Inform Assoc. 2000; 7: 313-323.
5. Stahl DC, Stahl CL, Niland JC. Evaluating the Impact of Clinical Trials On-Line On Clinical Trial Awareness and Accrual. Proc 33rd Annual Hawaii Intl Conf on System Sciences. 2000.

6. Breitfeld PP, Weisburd M, Overhage FM, Sledge G, Tierney WM. Pilot Study of a Point-of-use Decision Support Tool for Cancer Clinical Trials Eligibility. J Am Med Inform Assoc. 1999; 6: 466-477.
7. Oracle Clinical Version 3.0: User's Guide. Redwood Shores, CA: Oracle Corporation, 1996.
8. Shea S, DuMouchel W, Bahamonde L. A meta-analysis of 16 randomized controlled trials to evaluate computer- based clinical reminder systems for preventive care in the ambulatory setting. J Am Med Inform Assoc. 1996; 3: 399-409.
9. Mekhjian HS, Bentley TD, Ahmad A, Marsh G. Development of a Web-based Event Reporting System in an Academic Environment. J Am Med Inform Assoc. 2004; 11: 11-18.
10. Afrin LB, Kuppuswamy V, Slater B, Stuart RK. Electronic Clinical Trial Protocol Distribution via the World-Wide Web: A Prototype for Reducing Costs and Errors, Improving Accrual, and Saving Trees. J Am Med Inform Assoc. 1997; 4: 25-35.
11. Gillen JE, Tse Tony, Ide NC, McCray AT. Design, Implementation and Management of a Web-Based Data Entry System for ClinicalTrials.gov. Medinfo. 2004; 2004: 1466-1470.
12. Inform. Phase Forward Inc. Available at http://www.phaseforward.com. Accessed June 19, 2007.
13. Unutzer J, Choi Y, Cook IA, Oishi S. A Web-Based Data Management System to Improve Care for Depression in a Multicenter Clinical Trial. Psychiatric Services. 2002; 53: 671-678.
14. Nadkarni PM, Brandt CM, Marenco L. WebEAV: Automatic Metadata-driven Generation of Web Interfaces to Entity-Attribute-Value Databases. J Am Med Inform Assoc. 2000; 7: 343-356.
15. Nadkarni PM, Brandt C, Frawley S, Sayward FG, Einbinder R, Zelterman D, et al. Managing Attribute-Value Clinical Trials Data Using the ACT/DB Client-Server Database System. J Am Med Inform Assoc. 1998; 5: 139-151.
16. ClinTrial. Phase Forward Inc. Available at http://www.phaseforward.com. Accessed June 19, 2007.
17. Deshpande AM, Brandt C, Nadkarni PM. Metadata-driven Ad Hoc Query of Patient Data: Meeting the Needs of Clinical Studies. J Am Med Inform Assoc. 2002; 9: 369-382.
18. Nadkarni PM, Brandt C. Data Extraction and Ad Hoc Query of an Entity-Attribute-Value Database. J Am Med Inform Assoc. 1998; 5: 511-527.
19. Oliveira AG, Salgado NC. Design espects of a distributed clinical trials information system. Clin Trials. 2006; 3(4): 385-396.
20. Standards. Clinical Data Interchange Standards Consortium. Available at http://www.cdisc.org/standards/index.html Accessed June 19, 2007
21. Tobias J, Chilukuri R, Komatsoulis GA, Mohanty S, Sioutos N, Warzel DB, Wright LW, Crowley RS. The CAP cancer protocols-a case study of caCORE based data standards implementation to integrate with the Cancer Biomedical Informatics Grid. BMC Med Inform Decis Mak. 2006 Jun 20; 6: 25.
22. Clinical Data Acquistion Standards Harmonization. Clinical Data Interchange Standards Consortium. Available at http://www.cdisc.org/standards/cdash/index.html Accessed June 19, 2007
23. Kuchinke W, Wiegelmann S, Verplancke P, Ohmann C. Extended cooperation in clinical studies through exchange of CDISC metadata between different study software solutions. Methods Inf Med. 2006; 45(4): 441-446.
24. Shepherd M, Zitner D, Watters C. Medical Portals: Web-Based Access to Medical Information. Proc 33rd Annual Hawaii Intl Conf on System Sciences. 2000.
25. Huff SM, Huang DJ, Stevens LE, Dupont CC, Pryor TA. HELP the next generation: a new client-server architecture. Proc 18th Symp Comput Appl Med Care. 1994: 271-275.
26. Huff SM, Berthelsen CL, Pryor TA, Dudley AS. Evaluation of SQL model of the HELP patient database. Proc 15th Symp Comput Appl Med Care. 1991: 386-390.
27. Friedman C, Hripcsak G, Johnson S, Cimino J, Clayton P. A generalized relational schema for an integrated clinical patient database. Proc 14th Symp Comput Appl Med Care. Washington, DC: IEEE

Computer Society Press. 1990: 335-339.
28. Bleeker SE, Derksen-Lubsen G, van der Lei J, Moll HA. Structured data entry for narrative data in a broad specialty: patient history and physical examination in pediatrics. BMC Med Inform Decis Mak. 2006 Jul 13; 6: 29.
29. Marenco L, Tosches N, Crasto C, Shepherd G, Miller PL, Nadkarni PM. Achieving Evolvable Web-Database Bioscience Applications Using the EAV/CR Framework: Recent Advances. J Am Med Inform Assoc. 2003; 10: 444-453.
30. Salgado NC, Gouveia-Oliveira A. Towrds a common framework for clinical trials information systems. Proc AMIA Symp. 2000: 754-758.
31. Turau V. A framework for automatic generation of web-based data entry application based on XML, Proceedings of ACM Symposium on Applied Computing 2002.
32. Schweiger R, Hoelzer S, Altmann U, Rieger J, Dudeck J. Plug-and-Play XML: A Health Care Perspective. J Am Med Inform Assoc. 2002; 9: 37-48.
33. Furht B, Sheen J, Aganovic Z. Internet-Based Delivery and Deployment of Document Management System. Proc 34th Annual Hawaii Intl Conf on System Sciences. 2001.
34. Lowry PB. XML Data Mediation and Collaboration: A Proposed Comprehensive Architecture and Query Requirements for Using XML to Mediate Heterogeneous Data Sources and Targets. Proc 34th Annual Hawaii Intl Conf on System Sciences. 2001.
35. Dolin RH, Alschuler L, Beebe C, Biron PV, Boyer SL, Essin D, et al. The HL7 Clinical Document Architecture. J Am Med Inform Assoc. 2001; 8: 552-569.
36. Operational Data Model Ver. 1.3. Clinical Data Interchange Standards Consortium. Available at: http://www.cdisc.org/models/odm/v1.3/index.html. Accessed June 19, 2007.
37. Heritier SR, Gebski VJ, Keech AC. Inclusion of patients in clinical trial analysis: the intention-to-treat principle. Med J Aust. 2003; 179(8): 438-440.
38. SackettD, Gent M. Controversy in counting and attributing events in clinical trials. N Engl J Med 1979; 301: 1410-1412.