

Web Based Data Capture for Clinical Research

Stephen Smith
Datapoint Technologies,
Erwinna, PA, USA

Samuel Sambasivam
Computer Science
Department, Azusa Pacific
University, Azusa, CA, USA

stephen@datapoint-echнологies.com

ssambasivam@apu.edu

Abstract

Electronic Data Capture (EDC) is increasingly being used in the pharmaceutical, biotech and medical device industries to gather research data worldwide from doctors, hospitals and universities participating in clinical trials. In this highly regulated environment, all systems and software must be thoroughly tested and validated, a task that is burdensome in terms of time and cost.

Starting with database structures that are designed to be copied easily, this paper proposes a simple framework that allows for rapid development and minimal testing. The framework includes tools for building modules, for copying modules from one trial to the next, and tools to validate that the modules are the same as modules that have been fully tested previously. A proof-of-concept prototype has been built to demonstrate certain tools and techniques that can be used when designing and building a simplified EDC interface.

Keywords: electronic data capture, object orientation, prototyping, clinical research

Introduction

Electronic Data Capture (EDC) is increasingly being used in the pharmaceutical, biotechnology and medical device industries to gather research data worldwide from doctors, hospitals and universities participating in clinical trials.

“Over the past fifteen to twenty years, many different forms of site-based capture of clinical trial data have been developed. Data collection technology that has been applied to improve clinical trial data collection is generally referred to as electronic data capture (EDC)” (Kush et al., 2003)

The Clinical Data Interchange Standards Consortium (CDISC) defines an electronic clinical trial (eCT) as:

“Clinical Trial in which primarily electronic processes are used to plan, collect (acquire), access, exchange and archive data required for conduct, management, analysis and reporting of the trial” (CDISC, 2006).

Material published as part of this publication, either on-line or in print, is copyrighted by the Informing Science Institute. Permission to make digital or paper copy of part or all of these works for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage AND that copies 1) bear this notice in full and 2) give the full citation on the first page. It is permissible to abstract these works so long as credit is given. To copy in all other cases or to republish or to post on a server or to redistribute to lists requires specific permission and payment of a fee. Contact Publisher@InformingScience.org to request redistribution permission.

This paper hopes to explore the technologies, designs and best practices for web-based data capture of clinical research data. The project proposes a lightweight framework for deploying EDC-based clinical trials quickly by leveraging the use of pre-validated and reliable "modules".

Starting with database structures that are designed to be copied, a simple, less fully- featured approach should allow for rapid development and minimal testing. It is recognized that in this scenario, much of the intelligence for managing and cleaning the data would be missing but could be built in after the go-live date. I suggest that this is a more efficient and logical approach when, in most cases, sponsors of clinical trials want to start collecting data as soon as possible after the study design has been finalized.

Scope

This project is primarily a software design and engineering effort, a framework has been built for constructing web-based data collection forms. This building tool itself is web-based, the framework includes tools for building modules, for copying modules from one trial to the next, and tools to validate that the modules are the same (or not) as modules that have been fully tested previously. In this way, validation of the components used for each trial will involve identifying and testing only the delta from previously tested modules. This proof-of-concept prototype was built to demonstrate certain tools and techniques that can be used when designing and building a simplified EDC interface.

In addition to the building tool, the system for serving up the data entry forms was constructed to show how clinical research forms, or CRFs, might look on the web. Both the building tool and the data entry system is available on the publicly available World Wide Web so that interested parties will effectively be able to build a clinical trial and then access the data entry forms themselves.

The whole system will contain only the level of detail and functionality required to demonstrate the use of certain design techniques. It is acknowledged that in real life, a system for capturing clinical data would necessarily have to include other functionality including (but not limited to) features such as a security sub-system, multiple role capabilities, reporting and export features etc. While this functionality could be added, the intent was not to build a fully featured, “ready to use, out of the box” EDC system.

Approach

While the use of Open Source software tools is often controversial, the technologies used for this prototype were PHP and MySQL. Within that technology framework, an object-oriented design and programming approach was taken. While not strictly an Object Oriented language like Java, C# or Smalltalk – PHP version 5 does support OO constructs and methods.

Key to the design is the encapsulation of all Trial elements within database structures including edit-check logic to be applied to data elements. This “metadata” approach allows for simple data copy and compare processes that are important tools for building reusable modules. The Trial elements mentioned here refer to a hierarchy of objects which together go to make up the Data Capture aspects of a Clinical trial. They are discussed in greater detail in a later section, but briefly they are as shown in Figure 1.

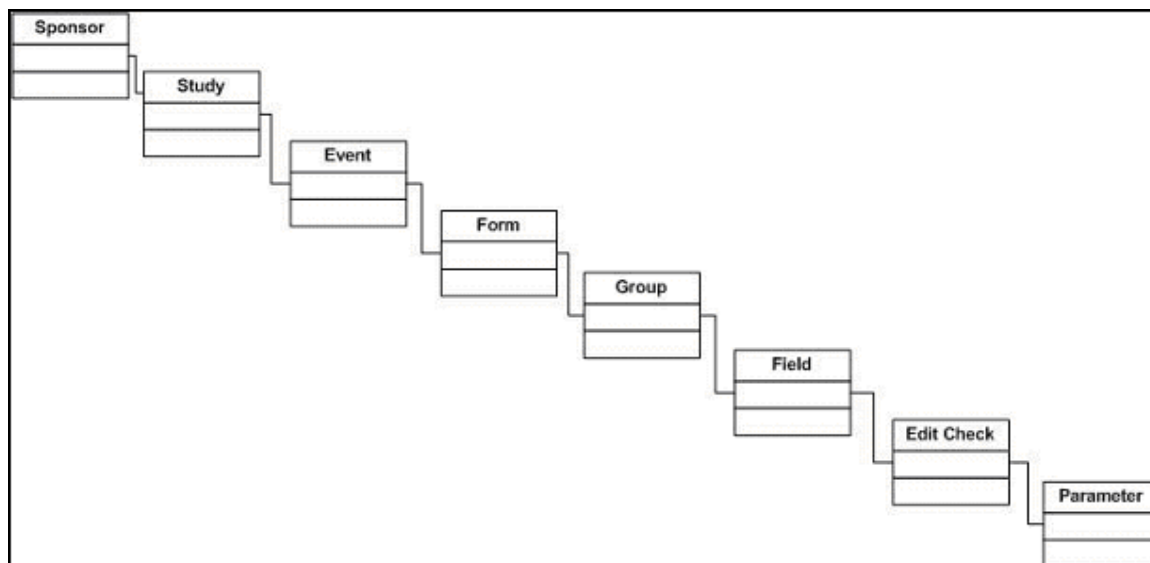


Figure 1 Hierarchy of Trial Objects

Outcome

The Study Build tool is available for use on the public Internet; the URL is <http://208.131.145.200/studymaster/main.php>

Functionality includes full ability to list, add, edit, copy and delete all trial components.

After building or modifying a trial, the results can be viewed at the following link: <http://208.131.145.200/studymaster/studies/login.php>

A user with the following credentials has been set up to access the studies of the “Bristol Myer” sponsor:

Username: steve
Password: dtg

Industry Literature

According to a report from the Tufts Center for the Study of Drug Development, it takes US drug companies typically between 10 to 15 years from drug discovery to approval from the FDA to market the drug. Also, for every 5 drugs that enter the clinical testing phase, only one will finally be approved for patient use (Hewitt, 2001). Clinical testing proceeds through a series of phases defined by the FDA:

Phase 1 – studies are conducted on a small number of healthy volunteers (between 20 and 80) to examine how a drug is metabolized by the human body. Dosing level decisions are made based upon the results, testing will not proceed to the next phase if any serious adverse side effects are discovered.

Phase 2 – conducted on several hundred subjects who have a disease or condition to assess the efficacy of the drug. If the drug proves to be effective and any identified risks are considered to be acceptable, the drug moves on to phase 3.

Phase 3 – the drug is administered to thousands of subjects with a disease, typically in a double-blinded fashion (both the patient and the doctor do not know whether the drug being used contains the

active ingredient or placebo). Often the drug is tested against existing therapies to statistically prove or disprove superior efficacy. The larger patient populations will often uncover less common adverse side effects (FDA, 2006).

Historically, data collected for these trials have been paper-based with doctors recording information on paper case report forms (CRF's) and sending them to the drug company where the information is analyzed.

“The paper-based approach to clinical trial data collection, exchange and entry presents multiple opportunities for errors and delays” (IBM, 2007).

Over the past ten to fifteen years, several remote and electronic data capture technologies have been used to exchange this data, significantly reducing the time to data review. In addition, edit check logic can be applied at the point of data entry to catch invalid, non-conformant or conflicting data resulting in cleaner data earlier. There is also a potential for improvement in safety with adverse events being identified faster.

“EDC appears to be catching on, not because it is now possible, but because a small number of sponsors have shown that clinical development can be accelerated while actually reducing cost” (Collins, 2007).

Industry analysts predict a sharp increase in the use of these technologies over the next few years. One report from IDC Health Industry Insights predicts that 45% of all trials will be conducted using EDC by the end of 2007, up from 24% at the end of 2005:-

“The life science industry is undergoing a fundamental shift away from batch-processes where clinical data recorded on paper can be locked up for weeks. EDC enables life science companies to conduct adaptive clinical trials that operate in real-time” (Connor, 2007).

Central to each clinical study is a document called the “Protocol” designed with clinical and statistical aims in mind. From the Clinical Data Interchange Standards Consortium (CDISC) glossary of terms, the protocol is:

“...a document that describes the objective(s), design, methodology, statistical considerations, and organization of a trial” (CDISC, 2006).

There is a critical time period after protocol finalization and before data-entry is enabled during which the technical components of an EDC system are designed, configured and tested. The goal of this project was to propose techniques to reduce this period through the use of easily reusable, standard components.

Project Documentation

Requirements were broken down into the Study Build system and the Data Entry system.

Study Build

In this simplified version of EDC, a study builder will need to list, add, modify, copy, or delete a number of study components:-

- Sponsor – any company or institution conducting clinical trials. This could include pharmaceutical or biotech companies, medical device manufacturers or educational institutions.
- Study – describes the studies that are conducted by a sponsor
- Event – analogous to a patient visit at which time clinical information is collected. However it could also include non-scheduled events such as an adverse experience.

- Form – similar to a paper case report form, each form is a “module” of information that is collected at an event (vital signs, ECG, blood chemistry for example).
- Group – a logical group of questions on a form, can be useful for describing repeating groups.
- Question – a single piece of information collected on a form.
- Edit check – logic applied to a question as a “sanity check”, used as a data quality aid.
- Parameter – an edit check may contain one or more parameters describing the constraints that will be applied to a question’s data.

The Use Cases for viewing, manipulating or copying these components are as shown in Figure 2.

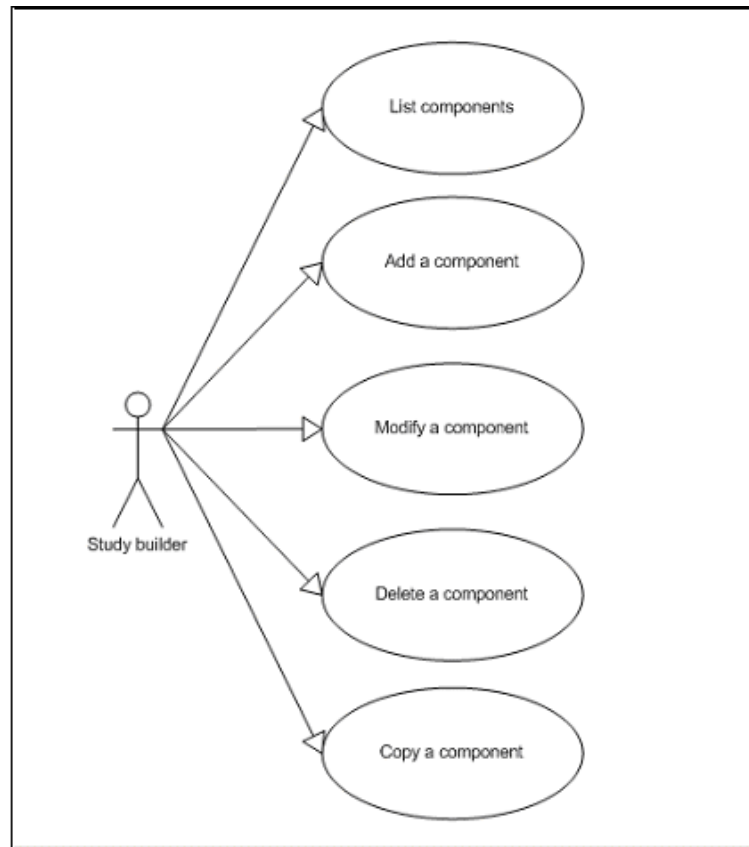


Figure 2 Study build use cases

Key to the concept of reusability is that a study builder will have the ability to copy any component, whether it is a form, an edit check or an entire study.

Data Collection

Run-time use cases based upon a study sponsor’s particular work stream processes could typically include those shown in Figure 3.

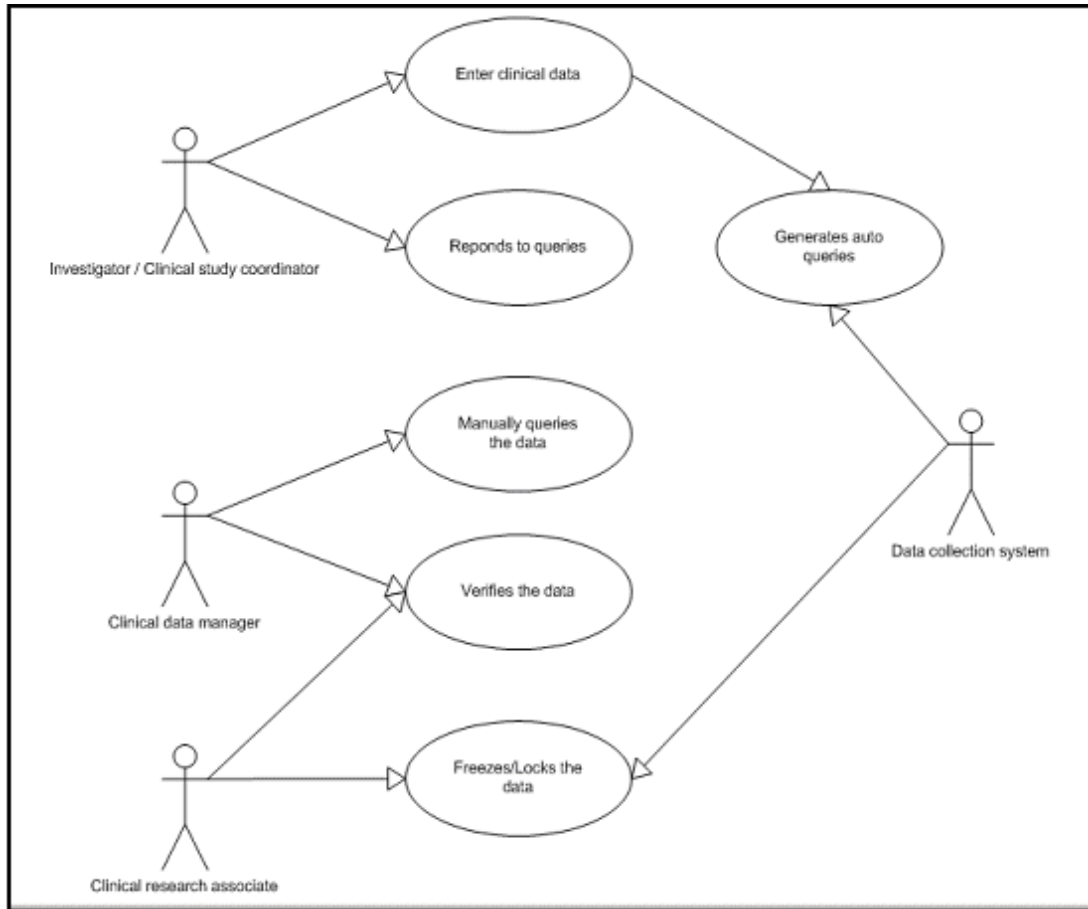


Figure 3 Run time use cases

As previously mentioned the functionality of this prototype is limited but demonstrates:

Web-based forms for entering clinical data automatically generated from metadata architected in the study build process.

System generated queries based upon automatic metadata-defined edit checks.

The ability to copy study components from one trial to the next.

Using this metadata approach, validation of new trials can include simple data comparison procedures.

Architecture

Both the Study Build tool and the Data Collection tool are web-based. The build tool constructs the *metadata* that defines the structure of each study, the metadata drives the forms and edit check logic of the Data Collection system.

Study Build Tool

The hierarchy of the primary objects for the study build tool are as shown in Figure 4.

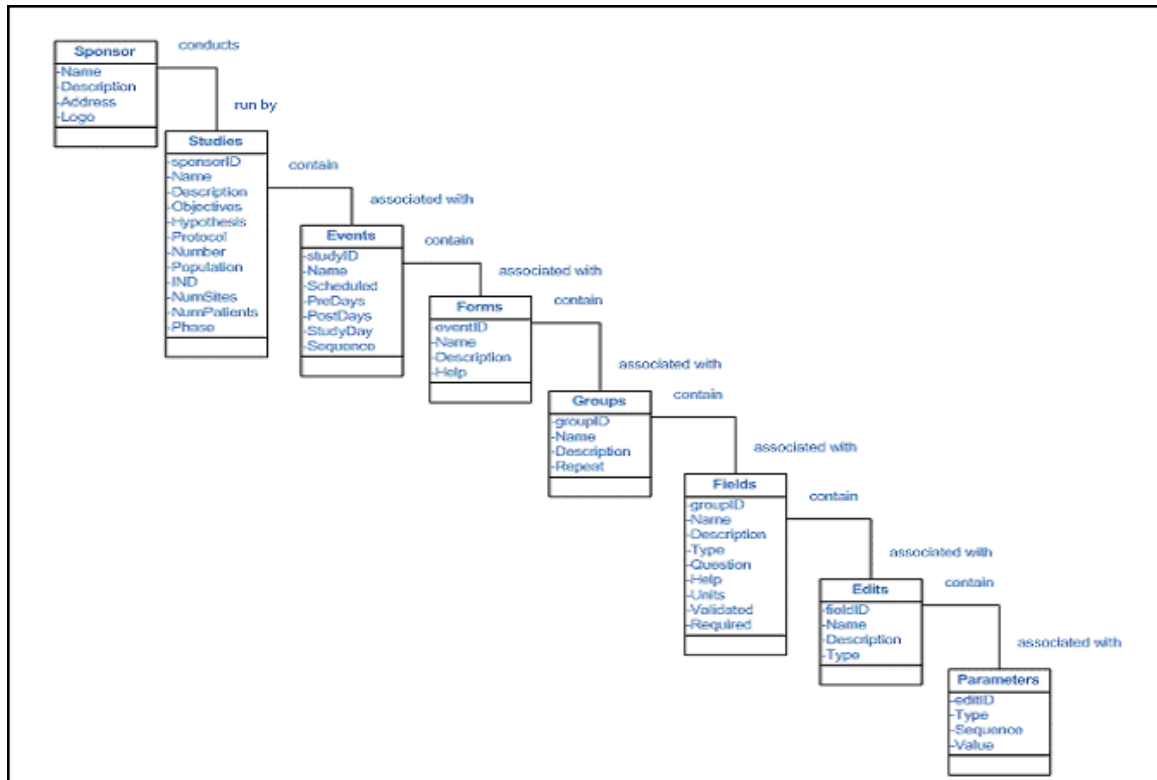


Figure 4 Study Build Object Hierarchy

All of these objects inherit from a single “Base” parent that allows for a single UI for each of the tool’s use cases.

The logic for each of these functions are implemented in the parent class; for example the following shows stubs for the constructor and copy methods of the “Base” parent class:-

```

class Base

/* Constructor */
function TrialComponent ($classID=0, $className="", $classKey="")
{
    // constructor code...
}

/* Copy function */

function copy_instance ()
{
    // copy this
    // copy children
}
  
```

Each of the specific trial components inherit from this base object and will be intelligent enough to know about the objects above and below it in the hierarchy. As an example, the Forms object might look like:-

```
Class Forms extends Base  
{  
var $keyName="formID";  
var $childrenName="Groups";  
var $childKeyName="groupID";  
var $parentKeyName = "eventID";  
var $parentName = "Events";  
}
```

Erich Gamma in his book “Design Patterns” describes this form of inheritance as “white box reuse” referring to the visibility of the internals of the parent class (Gamma, 1995).

The screenshot in Figure 5 is the list UI for the Form component:-



Figure 5 List Forms (list.php)

The same UI is used for all other Trial components, for example Sponsors and Studies as shown in Figures 6 and 7.

StudyMaster - List Sponsors

	ID	Sponsor Name	Description
<input type="radio"/>	1	SuperHuge Pharma	Big pharmaceutical company
<input checked="" type="radio"/>	4	Sharing Plough	A cooperative farmer-ceutical company
<input type="radio"/>	5	Bristol Myers and Coke	Doing research on sclerosis of the liver
<input checked="" type="radio"/>	6	Murk	Where PATENTS come first - pain killers and cardiovascular events

Figure 6 List Sponsors (list.php)

StudyMaster - List Studies



	ID	Study Name	Number	Description
<input type="radio"/>	13	Diabetes study 1	DIAB0012	Adult diabetic study
<input checked="" type="radio"/>	14	Diabetes study 2	DIAB0099	Juvenile diabetes study
<input type="radio"/>	15	Oncology study 1	ONC0001	Breast cancer study
<input checked="" type="radio"/>	16	Bird flu vaccine study	VAC0002	Vaccination study
<input type="radio"/>	17	Weight loss	WGT0003	Weight loss study
<input checked="" type="radio"/>	18	Diabetes study 3	DIAB0012	Adult diabetic study
<input type="radio"/>	19	Diabetes study 4	DIAB0101	Juvenile diabetes study
<input checked="" type="radio"/>	20	Oncology study 2	ONC0003	Breast cancer study
<input type="radio"/>	21	Flu study	VAC0005	Vaccination study
<input checked="" type="radio"/>	22	Cardiovascular study	CAR0003	Improve cardiovascular function

Figure 7 List Studies (list.php)

A Study Builder can navigate down through the object hierarchy by clicking on the hyperlinked object name in the list. For example, clicking on a study name will bring up a list of associated events, clicking on an event name will bring up a list of forms.

From any level, a new component can be added by clicking the Add button. Through the “fieldquestions” metadata table, the Add form is smart enough to know what fields to paint for any given object. In the example shown in Figure 8, this new Demographics form once submitted will be added to the parent Event:-

Event	1
Name of Form	Demog
Description	Demographics
Help Text	Help text

Submit Cancel Back to Top

Figure 8 Add Form

Also from the List UI, any row can be selected by clicking on one of the radio buttons in the first column, then Edit, Delete or Copy can be chosen. Edit will bring up a UI very similar to the populated Add form. Information for any of the objects can be modified and submitted through this form.

Hitting the Delete button once a row is selected will bring up the warning screen shown in Figure 9.

Delete this object - Are you sure?

Study	1
Event Name	Screening
Scheduled	Yes
Study Day	0
Allowable days early	0
Allowable days late	0
Sequence	1

Delete Cancel Back to Top

Figure 9 Delete screen

Copy is the powerful feature that allows a Study Builder to select an object at any level and copy it along with all its children objects to a new sibling. As an example shown in Figure 10 – if an event called Visit 4 was identical to a subsequent visit 5, the row can be selected and a sibling created and renamed.



Figure 10 Copy functionality

This feature guarantees that the object and all children in the branch below it are copied to the sibling. The recursive code that accomplishes this is implemented in two functions defined in the parent “Base” class – `copy_instance` and the function that it calls `copyChildRows`:-

```
function copy_instance ($parentKeyValue=0, $parentKey="")
{
    /* Copy the row */

    $query = "select * from " . $this->name . " where " . $this->key . "=" . $this->id;
    $result = $mysqli->query($query) or die(mysql_error());
    $row = $result->fetch_array(MYSQLI_BOTH);

    ....A lot of other code, see Appendix A for full list

    /* Copy children rows */

    if ($this->copyChildRows($mysqli))
        {return TRUE;}
    else
        {return FALSE;}
}
```

The function copyChildRows then recursively calls copy_instance for each of its children:-

```
function copyChildRows($mysqli)
{
    foreach ($this->children as $childKey)
    {
        $obj = new $this->childrenName($childKey,$this->childrenName,$this->childKeyName);
        if (!$obj->copy_instance($key,$this->keyName))
        {
            return FALSE;
        }
    }
}
```

As far as edit checks to validate the data that is entered through the Data Collection Tool, the types of check shown in Figure 11 have been implemented for the sake of demonstration:-

Field	35
Name of Edit	After1950
Description	Patient must have been born after 1950
Type	After
Discrepancy Text	Patient must have been born after 1950

Submit Cancel Back to Top

Figure 11 Edit Checks

A field can be defined as “Required”, meaning that leaving it blank would fire a discrepancy message. “Range” checks that it is between 2 numbers requiring 2 child parameters. “GT”, “GE”, “LT” and “LE” are simple numeric comparisons against a single parameter. “After” is a date comparison – in the example shown in Figure 11, a date of birth field is checked to ensure that a patient was born after 1950. This edit check and others will be demonstrated through the Data Collection Tool.

StudyMaster - List Edits



	ID	Edit Name	Description	Type
<input type="radio"/>	3	Greaterthan50	Systolic must be greater than 50	GT
<input type="radio"/>	4	Lessthan150	Systolic must be less than 150	LT
<input type="radio"/>	5	Required	Systolic is a required field	Required

Figure 12 Edit Checks on Systolic Blood Pressure field

The Figure 12 shows that three edit checks have been set up for a field designed to collect systolic blood pressure. The field is required, and it must be between 50 and 150. It will be shown how these edit checks appear in the Data Collection tool.

Data Collection Tool

The run-time Data Collection system is driven by the metadata entered through the build process. The major objects of this Data Collection tool are shown in Figure 13.

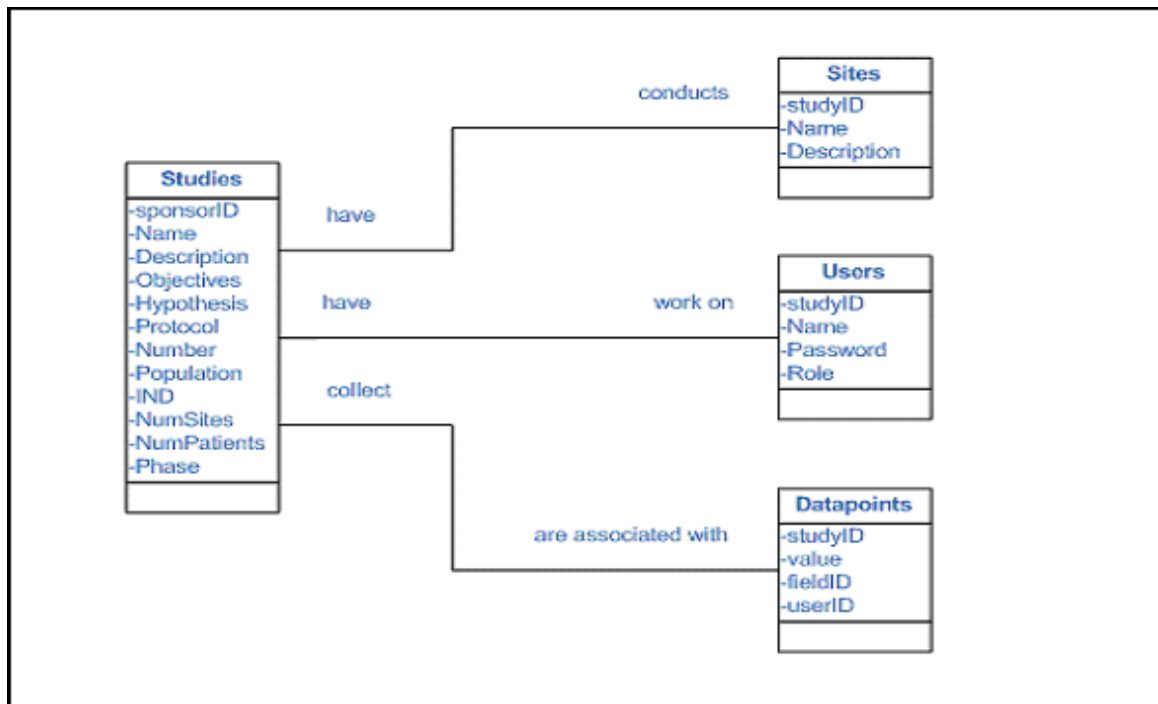
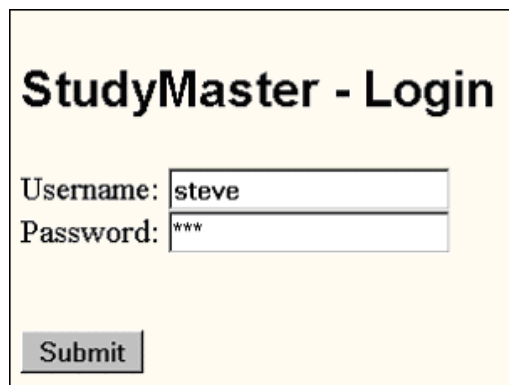


Figure 13 Data Collection Object Hierarchy

Web Based Data Capture

- Studies – describes the studies that are conducted by a sponsor
- Sites – a location where clinical data is collected and recorded, it can be a doctor’s office, a hospital or a university department.
- Users – anyone who records or accesses a study’s clinical information (doctor, study coordinator, clinical research associate, clinical data manager).
- Datapoints – the actual data that is collected, only meaningful in the context of the metadata that describes it.

For security purposes, all users of the Data Collection tool must logon with a unique username and password (see Figure 14).




StudyMaster - Login

Username:

Password:

Figure 14 Login screen

For the sake of simplicity it is assumed that each user defined in the Data Collection system belongs to a single sponsor. After logging on, the user will see a list of trials that have been defined for the user’s sponsor as shown in Figure 15.



Bristol Myers and Coke

Study Name	Number	Description
Diabetes study 1	DIAB0012	Adult diabetic study
Diabetes study 2	DIAB0099	Juvenile diabetes study
Oncology study 1	ONC0001	Breast cancer study
Bird flu vaccine study	VAC0002	Vaccination study
Weight loss	WGT0003	Weight loss study
Diabetes study 3	DIAB0012	Adult diabetic study
Diabetes study 4	DIAB0101	Juvenile diabetes study
Oncology study 2	ONC0003	Breast cancer study
Flu study	VAC0005	Vaccination study
Cardiovascular study	CAR0003	Improve cardiovascular function

Figure 15 Study List

After selecting the study by clicking on the hyperlink, the list of sites for that study appears (Figure 16).

DIAB0012 - Site List

Name	Description
Atlanta Diabetes Center	Dr. Martin Fox
Gulf Coast Clinical Research	Research Center in Tampa
UCLA	UCLA Clinical Research Department

Figure 16 Site List

After selecting the site to be worked on, a list of patients is displayed (Figure 17).

Atlanta Diabetes Center - Patient List

Patient Initials
SSS
SSS
TTT
TTT

Figure 17 Patient List

The user at this point has the option of selecting an existing patient or entering the initials of a new patient. If the patient already existed, the complete trial structure of visits, forms and fields will have already been created and the user is taken directly to the patient trial tree. If on the other hand a patient is being entered for the first time, the structure must be created. The code that accomplishes this is:-

```

/* create new patient record */
$query = "insert into patients (initials, siteID) values ('" . $inits . "', '" . $site . "')";
if (!$result = $mysqli->query($query))
{
    printf("Insert failed: %s\n", mysqli_connect_error());
    exit();
}
$obj = $result->fetch_object();
$pt = new Patient($obj->patientID, $study);

```

A row is inserted into the actual “patients” table, then a new patient object is instantiated. The patient class constructor method instantiates an event for each visit defined in metadata by the Build Tool for this trial:-

```

/* return all scheduled events for the study */

$query = "select * from events where eventScheduled = 'Yes' and studyID = '" . $study . "'
order by eventSequence";

if ($result = $mysqli->query($query))
{
    while ($obj = $result->fetch_object())
    {
        $event = new Event($obj->eventID, $study, $ptID, 0);
    }
}

```


Each event created will instantiate all forms defined for that visit, and each form will instantiate its fields. In this way, the complete trial structure is navigated and created. Persistence of this information is performed by the Base object from which all trial objects inherit. The code below externalizes each object by writing a record to the “patientinstances” table:-

```
function Base ($ID=0, $study="", $pt="", $parent=0)
{
    $values = $study . "," . $pt . "," . $this->instanceType . "," . $ID . "," . 'Yes' . "," . $parent;
    $query = "insert into patientinstances (studyID, ptID, instanceType, ID, scheduled,
parentID) values (" . $values . ")";
    $result = $mysqli->query($query);
    $this->instantiateChildren ($ID, $study, $pt, $obj->instanceID, $mysqli);
    $mysqli->close();
}
```

Whether a new patient is created or an existing patient is reopened, the patient’s trial hierarchy will be displayed (Figure 18).

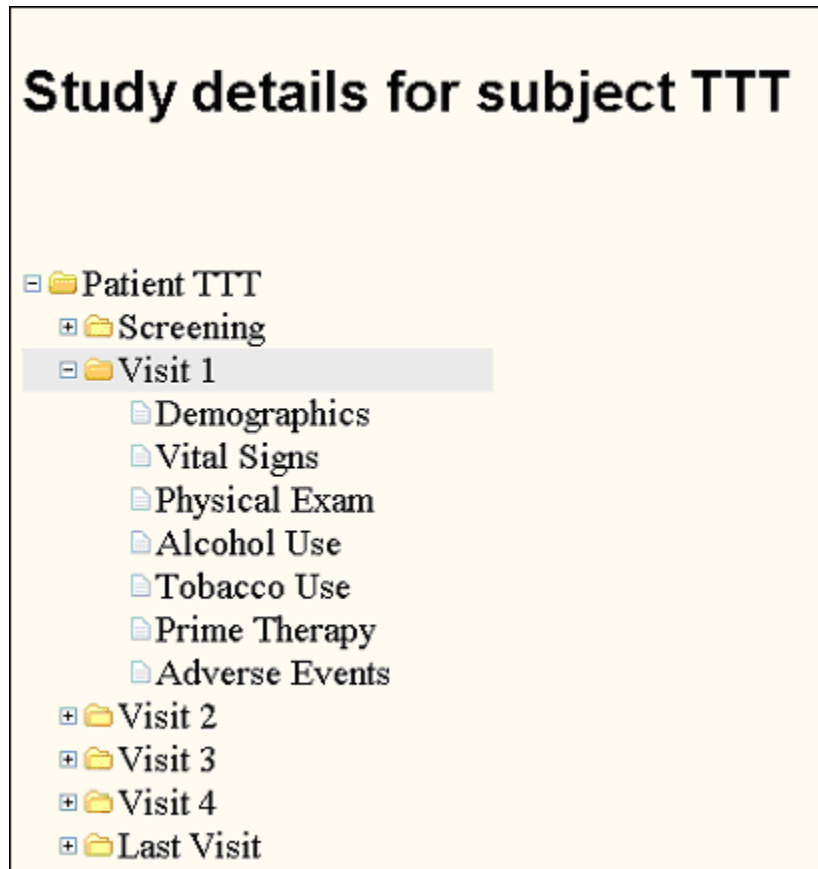


Figure 18 Patient Trial Hierarchy

Nodes on the tree can be expanded and collapsed until the individual case report forms can be seen (Demographics, Vital Signs, etc.). Clicking on a form hyperlink invokes the form.php script with a parameter of node ID that is used as a key to look up the row in the patientinstances table. The code then renders all fields for that form and the data associated with it. When a form is entered for the first time, the fields will appear empty (Figure 19).

Vital Signs	
Date of Exam <i>mm/dd/yyyy</i>	<input type="text"/>
Height	<input type="text"/>
Weight	<input type="text"/>
Pulse	<input type="text"/>
Systolic	<input type="text"/>
Diastolic	<input type="text"/>

Figure 19 Empty Vital Signs Form

As the “Date of Exam” and “Systolic” fields have been set up with a “Required” edit check through the Study Build tool, submitting the form without any further data entry would result in the form being repainted with the following discrepancy text in red (Figure 20).

Vital Signs	
Date of Exam <i>mm/dd/yyyy</i> Exam date is required	<input type="text"/>
Height	<input type="text"/>
Weight	<input type="text"/>
Pulse	<input type="text"/>
Systolic Systolic is a required field	<input type="text"/>
Diastolic	<input type="text"/>

Figure 20 Required discrepancies

Instead if data is entered correctly and submitted, the user is returned to the patient’s study tree structure. Figure 21 shows the “GT” or greater than edit check that was set up on the Systolic field.

Vital Signs

Date of Exam <i>mm/dd/yyyy</i>	12/12/2002
Height	59
Weight	165
Pulse	65
Systolic	315
Systolic must be less than 150	
Diastolic	80

Submit Cancel

Figure 21 GT edit check

Similarly, Figure 22 shows how the date check that was set up on the Date of Birth field appears on the Demographics forms.

Demographics

Date of Birth <i>mm/dd/yyyy</i>	12/12/1948
Patient must have been born after 1950	
Race	Caucasian ▾

Submit Cancel

Figure 22 Date edit check

In this way, data entered can be preliminarily validated by using simple database-driven edit checks. As all form metadata and logic is encapsulated in simple data structures, simple procedures for copying and comparing trial objects can be employed to assist in ensuring quality when building and deploying trials.

Conclusion

A key design goal of the project has been to have a trial completely described within data structures without additional code which will need to be validated each time a trial is copied or built from scratch. When this is done, it should be fairly simple to build in procedures that compare trial com-

ponents with those that have been previously tested. These procedures could even be kept within the database in the form of stored procedures. As an example, an SP to compare forms might look like:-

```

DELIMITER $$
-----
DROP PROCEDURE IF EXISTS `studymaster`.`compareforms` $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `compareforms`(tbl1 int, tbl2 int, OUT msg
varchar(80))
BEGIN
DECLARE cnt int;
select count(t1.formID) into cnt from (select * from forms where formID = tbl1) t1,
      (select * from forms where formID = tbl2) t2
where t1.formName = t2.formName and
      t1.formDescription = t2.formDescription and
      t1.formHelp = t2.formHelp and
      t1.formSequence = t2.formSequence
group by t1.formID;
if cnt > 0 then
/* you could call a lower-level component compare procedure in here */
  set msg = 'They are the same';
else
  set msg = 'They are different';
end if;
END $$
-----
DELIMITER ;

```

Calling this stored procedure with 2 form ID's from the command line would look like the following:-

```
Mysql>call compareforms(104,106,@msg);
```

```
Mysql>select @msg
```

```

+-----+
| @msg          |
+-----+
| They are the same |
+-----+

```

```
1 row in set (0.00 seconds)
```

The procedure could itself call compare procedures for the lower level component (group) which would call a compare procedure for all fields, and so on.

In this way, it is hoped that quality is automatically built in to trial components that have been copied and previously tested. This should result in a decrease in the time, effort and expense of the technical design and configuration of each trial in a field where this activity is often on the critical path towards the drug approval process.

References

- CDISC. (2006). Clinical research glossary. Retrieved 2007 March 15 from <http://cdisc.org/glossary/CDISCGlossaryV5.pdf>
- Collins, S. (2007). *Next generation pharmaceutical – The need for speed*. Retrieved 2007 May 5 from <http://www.ngpharma.com/pastissue/article.asp?art=25516&issue=143>
- Connor, C. (2007). *EDC poised to disrupt life sciences industry*. Retrieved 2007 May 5 from <http://www.healthindustry-insights.com/HII/getdoc.jsp?containerId=prUS20671907>
- Deitel, H., & Deitel, P. (2003). *Java – How to program*. ISBN 0-13-183661-7.
- Deitel, H., Deitel, P., & Goldberg, A. (2004). *Internet & World Wide Web – How to program*. ISBN 0-13-145091-3.
- Eclipse. (2007) *Eclipse – An open development platform*. Retrieved 2007 April 22 from <http://www.eclipse.org/>
- FDA. (2006). *Inside clinical trials: Testing medical products in people*. Retrieved 2007 May 2 from <http://www.fda.gov/fdac/special/testtubetopatient/trials.html>
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design patterns*. ISBN 0-201-63361-2.
- Hewitt, P. (2001). *How new drugs move through the development and approval process*. Retrieved 2007 May 2 from <http://csdd.tufts.edu/NewsEvents/RecentNews.asp?newsid=4>
- IBM. (2007). *EDC for life sciences*. Retrieved 2007 May 5 from <http://www-03.ibm.com/industries/healthcare/doc/content/solution/975043105.html>
- Kush, R., Bleicher, P., Kubick, W., Kush, S., Marks, R., Raymond, S., & Tardiff, B. (2003). *eClinical trials – Planning and implementation*. Thomson Centerwatch. ISBN 1-930624-28-X.
- Lee, J., & Ware, B. (2003). *Open source web development with LAMP*. ISBN 0-201-77061-X.
- Mon.itor.us (2007) *Linux versus Windows*. Retrieved 2007 April 2 from <http://www.cs.cornell.edu/courses/cs513/2002fa/opt1.soln.it33.pdf>
- MySQL AB. (2007). *MySQL*. Retrieved 2007 April 22 from <http://mysql.com/>
- Robinson, S., Nagel, C., Watson, K., Glynn, J. Skinner, M., & Evjen, B. (2001). *Professional C#*. ISBN 1861004990.
- Sun. (2007). Sun Java Studio Enterprise at a glance. Retrieved 2007 April 22 from <http://developers.sun.com/jenterprise/index.jsp>
- Tien, I. (2002) *Open source and assurance*. Cornell University. Retrieved 2007 April 2 from <http://www.cs.cornell.edu/courses/cs513/2002fa/opt1.soln.it33.pdf>
- Xinox Software. (2007). *JCreator*. Retrieved 2007 April 22 from <http://jcreator.com/>

Biographies



Stephen Smith has been involved in technology consulting for over 20 years, primarily in the Pharmaceutical and Financial industries. Originally from the UK, he now lives in the Eastern Pennsylvania, USA



Dr. Samuel Sambasivam is the chairman of the Department of Computer Science of Azusa Pacific University. Professor Sambasivam has done extensive research, publications, and presentations in both computer science and mathematics. His research interests include optimization methods, expert systems, Fuzzy Logic, client/server, Databases, and genetic algorithms. He has taught computer science and mathematics courses for over 25 years. Professor Sambasivam has run the regional Association for Computing Machinery (ACM) Programming Contest for six years. He has developed and introduced several new courses for computer science majors.

Professor Sambasivam teaches Database Management Systems, Information Structures and Algorithm Design, Microcomputer Programming with C++, Discrete Structures, Client/Server Applications, Advanced Database Applications, Applied Artificial Intelligence, JAVA and others courses. Professor Sambasivam coordinates the Client/Server Technology emphasis for the Department of Computer Science at Azusa Pacific University.