iDEN Mobile Devices – Advanced Technology
Seamless Health Center of Excellence

IEEE-SA Health IT Standards Study Group

For more information contact the editor at:

José C. Lacal
Senior Manager
Seamless Health Center of Excellence
+1 (954) 553-1984 Jose.Lacal@Motorola.com

Summary

This paper compiles a list of proposed Use Cases as part of the standards for Personalized Health Informatics ("PHI") systems.

Public Version 05 of 07/06/2006.

NOTICE: This document complements those available at http://grouper.ieee.org/groups/hit/theme_phi.html and elsewhere:

- http://grouper.ieee.org/groups/hit/files/Lacal_Proposal_IEEE-SA_Health_IT_Standards_Study_Group_032006_06.ppt
- http://grouper.ieee.org/groups/hit/files/Lacal_Proposal_IEEE-SA_Health_IT_Standards_Study_Group_032006_Scope_03.ppt
- Latest version of Personalized Health Informatics proposal
- Latest version of FAQ document
- Latest version of Next Steps document
- http://lacal.net/files/hs/HealthSmart_M.pdf

Please read these files first and only then this document. Thanks.

Executive Summary

This document outlines specific Use Cases for the proposed PHI standards. The purpose of this document is to facilitate the development of a reference implementation of a Free/Open Source Software PHI-compliant system.

An open question at this time is whether we should target a specific disease to further define and refine the Use Cases. For example, to target information- and processes-heavy[1] diseases such as Cystic Fibrosis or autism.

[1] By "information- and processes-heavy" I mean those diseases where the afflicted person and his/her caregivers need to search, analyze, and manage massive amounts of information about the disease and its effects. Some of that information includes: tracking of encounters with providers; management of medications usage and compliance; administration of payments and reimbursements; analysis of research and published materials relevant to the patient's disease. In the case of CF, please see http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1290129

Table of Contents

Table of Tables

Table of Figures

Acronyms and Definitions

This section defines the acronyms and terms used throughout this document.

Term	Meaning
ACL	Access Control List: a set of data that informs a computer's operating system which permissions, or access rights, that each user or group has to a specific system object, such as a directory or file. Each object has a unique security attribute that identifies which users have access to it, and the ACL is a list of each object and user access privileges such as read, write or execute.[2]
Actor (within a UML framework)	An actor is something or someone who supplies a stimulus to the system. It can also be thought of as something the system requires in order to function. Without a customer (an actor) in a restaurant, the process of ordering food cannot begin. In addition to actors there also exist primary actors. Primary Actors interact directly with a system to achieve their goals. Supporting actors may be humans or systems called in to support the Primary Actor. Stakeholders can also be modeled as actors. They do not directly interact with the system but they are affected by the success of Primary Actor interactions. We differentiate active actors, who initiate interactions with a system, and passive actors, who are targets of requests or who are activated by the system.[3]
API	Application Programming Interface. The interface that a computer system, library or application provides in order to allow requests for services to be made of it by other computer programs, and/or to allow data to be exchanged between them.[4]
F/OSS	Free and Open Source Software, also F/OSS or FOSS, is software which is liberally licensed to grant the right of users to study, change, and improve its design through the availability of its source code. F/OSS is generally synonymous with free software and open source software, and describes the same licenses, culture, and development models.[5]
HRA	Health Risk Assessment. An assessment which uses statistics from the Center for Disease Control to calculate the user's top health risks. The assessment reports the user's appraised and achievable health age and provides suggestions for reducing risk and achieving a healthier lifestyle.[6]
OOA&D	Object Oriented Analysis and Design Over the past decade, Object Oriented Analysis and Design (OOA&D) has become the dominant software development paradigm. With it has come a major shift in the thought processes of all involved in the software development life cycle. Programming language support for objects began with Simula 67, but it was the emergence in the 1980's of hybrid languages, such as C++, Ada and Object Pascal that allowed OOA&D to take off. These languages provided support for both OO and procedural programming. Object Oriented programming became mainstream. An OO system is designed and implemented as a simulation of the real world using software artifacts. This premise is as powerful as it is simple. By using an OO approach to design a system can be designed and tested (or more correctly simulated) without having to actually build the system first.[7]

[2] Source : http://www.webopedia.com/TERM/A/ACL.html
[3] Source : http://en.wikipedia.org/wiki/Actor_%28UML%29
[4] Source : http://en.wikipedia.org/wiki/API
[5] Source : http://en.wikipedia.org/wiki/F/OSS
[6] Source : http://beta.testwell.org/index.php?id=1627&id_tier=3430
[7] Source : http://argouml.tigris.org/documentation/defaulthtml/manual/ch01.html#s2.ooad

PHI	Personalized Health Informatics. A comprehensive set of Internet-based tools that place the individual (and his/her dependents) at the center of an encompassing architecture of services that promote and enhance health. Focused on the family as the unit of care (each person is truly embedded into a larger family unit).
PHR	Personal Health Records
Reference Implementation	In computing, a reference implementation (or, infrequently, sample implementation) is a software example of a standard for use in helping others implement their own versions of the standard. A standard is much easier to understand with a working example in hand. The purpose of a reference implementation is generally to increase awareness and familiarization of the spec within the development community. While it is entirely possible for RI software to serve in the academic cause of pure knowledge, on a more pragmatic level they are generally intended to familiarize the market with a spec so that developers will be more likely to purchase or develop commercial implementations of the spec.[8]
Role (within a UML framework)	
Use Case	In software engineering, a use case is a technique for capturing the potential requirements of a new system or software change. Each use case provides one or more scenarios that convey how the system should interact with the end user or another system to achieve a specific business goal. Use cases typically avoid technical jargon, preferring instead the language of the end user or domain expert. Use cases are often co-authored by Business Analysts and end users.[9]
UML	The Unified Modeling Language (UML) is a non-proprietary, object modeling and specification language used in software engineering. UML is a general-purpose modeling language that includes a standardized graphical notation that may be used to create an abstract model of a system, sometimes referred to as the UML model. UML may be considered as an extensible modeling language since it offers a profile mechanism to customize the language. If a concept you need is not present in the base language, you may introduce it by defining a stereotype. The semantics of extension by profiles has been improved with the UML 2.0 major revision.[10]
XMI	XML Metadata Interchange (XMI) is the standard for saving the meta-data that make up a particular UML model.

Table 1: Acronyms and Definitions.

[8] Source : http://en.wikipedia.org/wiki/Reference_implementation
[9] Source: http://en.wikipedia.org/wiki/Use_Case
[10] Source: http://en.wikipedia.org/wiki/Unified_Modeling_Language

Version History

Throughout the document the editor readily acknowledges all the comments, criticisms, and suggestions for improvement he has received. Any and all remaining errors are solely the editor's fault.

01: José C. Lacal 06/30/2006
02: José C. Lacal 07/03/2006
03: José C. Lacal 07/04/2006
04: José C. Lacal 07/05/2006
05: José C. Lacal 07/06/2006 (First public release for comments).

Purpose of PHI Use Cases

This document is part of the project to develop Personalized Health Informatics ("PHI") standards. This document will serve as an initial repository to provide a high-level overview of what a PHI-compliant system should look like. This document will be freely available and distributed for comments. And it will be used as a tool to keep all volunteers focused on end goal, and to clarify interfaces.

NOTICE: You might want to take a free on-line tutorial on developing and using Use Cases.[11]

Reference Implementation

These Use Cases will guide the team to design, develop, deploy, debug, and distribute a reference implementation of a PHI-compliant system using 100% F/OSS.

Design

Develop

Deploy

Debug

Distribute

[11] "Specifying Functional Requirements With Use Cases." Available at http://www.cragsystems.co.uk/SFRWUC/index.htm

Actors and Roles

This section presents detailed information about each actor., and about the different roles those actors could take in the system.

Actors are Different and Separate From Roles[12]

Mr Southerby as CEO needs access to management information. Fred Smith as Order Entry Clerk needs to enter orders. Sometimes Mr Southerby may need to enter an order when Fred isn't there. That doesn't mean that we draw an arrow between Mr Southerby and the Enter Order use case. Entering orders is not the role of the CEO. This is Mr Southerby taking on the role of Order Entry Clerk, a mapping which occurs when Mr Southerby logs on and is granted access to the Enter Order use case by security settings defined by the system administrator.
Ultimately an actor has no properties other than its links to use cases. It is a logical set and nothing more. When naming actors we need to think of the role that a person takes on rather than their title. Most job titles involve the putting on of a number of different hats in different situations. So name the actor with the role rather than the job title if there is any confusion. Think hats!

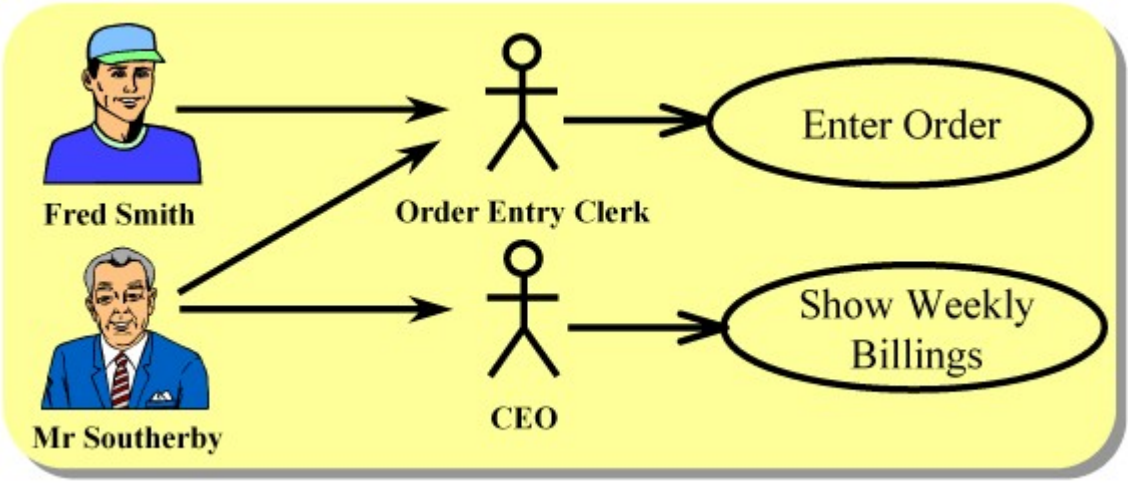

Figure 1: Actors as Roles.

List of Actors

Please notice: actors do not interact with the system. Roles do (see below).

Actor	Definition
Caregiver	Note: "Self" is also a caregiver (for oneself).
Guardian	
Guardian, secondary	• Divorced, non-custodial parent • Secondary, or non-caregiver relative or interested individual (trust or bank employee; government agency; court-appointed attorney)
Health Engineer, Community (HCE)	
Health Engineer, Personal (PHE)	
Provider	
System Administrator.	

Table 2: List of Actors.

[12] Source: http://www.cragsystems.co.uk/SFRWUC/index.htm, item 1.7

List of Roles

Everybody who interfaces with the system has to be considered as a role.

Roles	Definition
User_Admin	
User_Input	
User_Manage	
User_Monitor	

Table 3: List of Roles.

Format of Use Cases

This section defines the elements of each Use Case detailed below.

Use Case Name
Iteration
Summary
Preconditions
Triggers
Basic course of events
Alternative paths
Postconditions
Business rules
Notes
Author and date

Scenario
Actor(s)
Current challenges
Proposed solution
Required elements
Owner
Mapping to processes, Operating Cases ("OC") defined below

Each Use Case must be defined using UML tools in order to allow for re-use and maximum portability across contributors to this project.

Reference Documents

This document will be broken down into separate documents, one for each specific use case. There are a few seemingly useful "use case templates" available on the Internet. Such as:

- www.bredemeyer.com/pdf_files/UseCase_Template.PDF
- http://www.bredemeyer.com/pdf_files/functreq.pdf
-

PHI Use Cases

The following sections will present a high-level overview of each Use Case. Detailed information about each Use Case will be provided on a separate document for each one.

Use Case Nomenclature

These are the conventions used to number each Use Case.

UC_xyzz_aa	
x	"1" Create "2" Manage "3" Review "4" <TBD> "5" <TBD> "6" Actor is passive (system drives actions)
y	"1" Community "2" Family "3" Individual "4" <TBD> "5" <TBD> "6" Target is the system
zz	Auto increment number, from 01 to 99
aa	Version number (if greater than 0, initial version)

Table 4: Use Case Nomenclature.

The UML diagram below presents a graphical representation of the Use Cases detailed in the following sections.

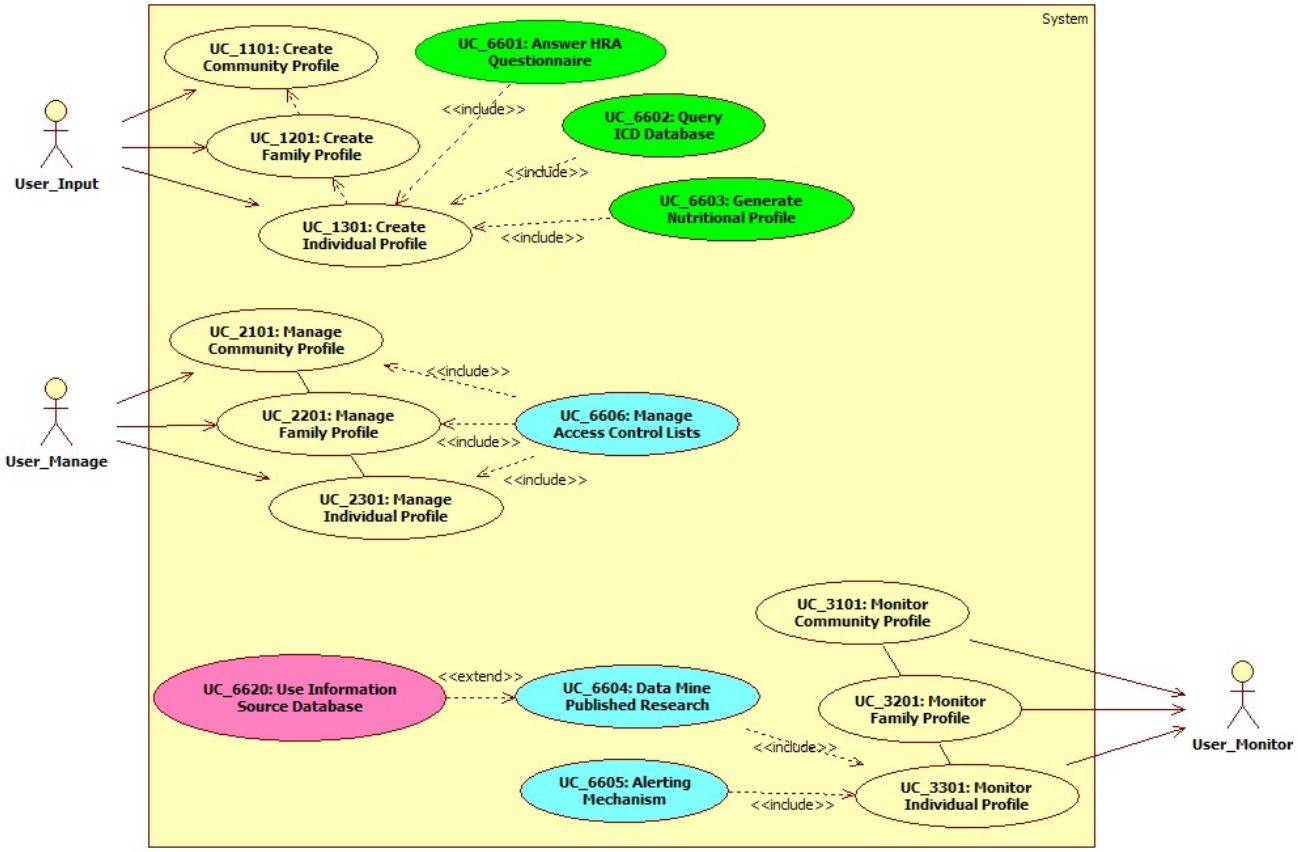

Figure 2: Overview of Use Cases.

UC_1101: Create Community Profile.

…

A “community” is defined as a group of families and individuals under the care of a single practitioner. For example: a nurse might be responsible to provide care to one or more individuals across one or more families.

Roles and Actors

UC_1201: Create Family Profile.

Define Family Unit

a.) Consumer for individual / family
b.) Caregiver for individual / family
c.) Community worker for individual / family

Roles and Actors

UC_1301: Create Individual Profile.

Inside each family, there are several individuals.

Create a biological, physical, and medical profile for each individual.

Questions to ask:
Sex (M/F) => sex-specific parameters
Age => age-specific parameters
Race => race-specific parametrs
Family history => family-specific parameters
Weights
Height
  => Compute BMI
    => Weight-specific parameters
Previous conditions
  => ICD-9 query
  => condition-specific parameters
Tobacco use => tobacco-specific issues
Socio-Economic Level
Marital Status

Health Index (self-reported)
Physical Activity Index
Issues
Environment

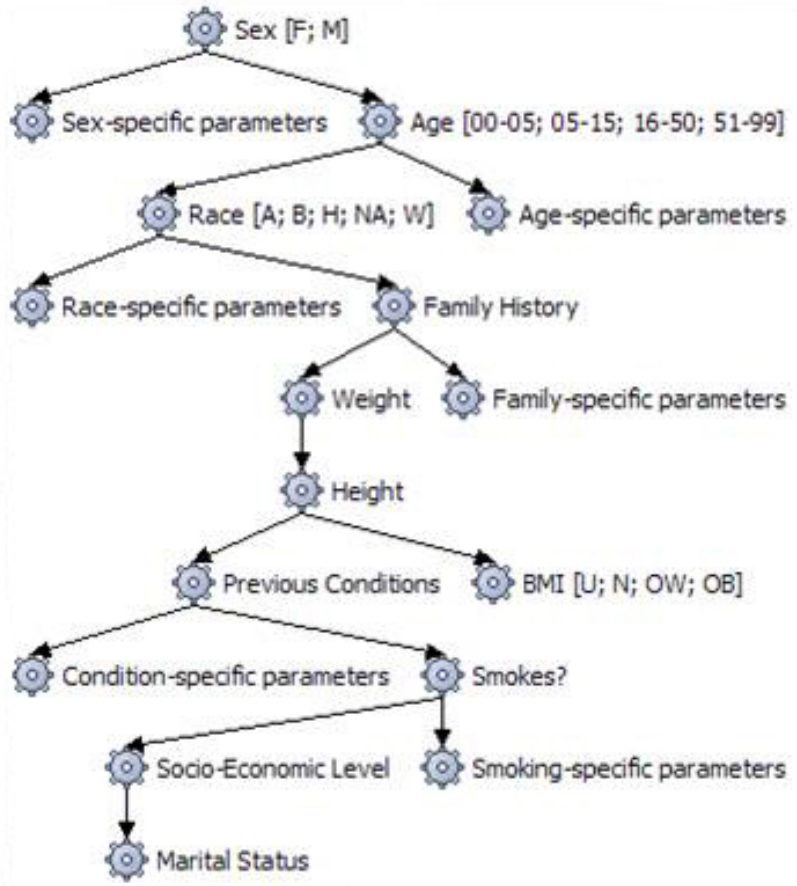

Figure 3: Create an Individual's Profile.

These questions and branching logic are already available from Health Risk Assessment documents.

The end goal is to generate an XML-based profile that allows the system to conduct data mining based on the elements highlighted in the profile.

>> Narrate a story! <<

277.39[13]
Other amyloidosis
Hereditary cardiac amyloidosis
Inherited systemic amyloidosis
Neuropathic (Portuguese) (Swiss)
amyloidosis
Secondary amyloidosis

>> Output: an XML-based Personal Profile file.
? Extend the CCR template?

[13] Source: ICD-9-CM TABULAR LIST OF DISEASES ADDENDA, Effective October 1, 2006 (FY07). Available at http://www.cdc.gov/nchs/datawh/ftpserv/ftpicd9/icdtab_addenda07.pdf; p. 05.

Roles and Actors

UC_2101: Manage Community.

…
Add, delete, edit familes.
Manage ACLs.

Roles and Actors

UC_2201: Manage Family.

..

Add, delete, edit individuals.
Manage ACLs.

Roles and Actors

UC_2301: Manage Individual.

..

Manage ACLs.

Roles and Actors

UC_3101: Monitor Community Profile.

Results of Data Mining
Present dashboard

Roles and Actors

Community Health Engineer (CHE) Monitors Community

UC_3201: Monitor Family Profile.

Results of Data Mining
Present dashboard

Roles and Actors

Personal Health Engineer (PHE)
monitoring an individual or family

UC_3301: Monitor Individual Profile.

Results of Data Mining
Present dashboard

Once a personal profile has been created for a target user

The end goal is for the system to provide the Guardian or caregiver actors with a complete guide as to how best to treat the target user.

Relevant and actionable information.
Prevention information.

Roles and Actors

Child / caregiver caring for elder

Consumer for self

Personal Health Engineer (PHE)
monitoring an individual

UC_6601: Answer HRA Questionnaire.

Health Risk Assessment Templates

Sources: http://www.mcare.org/include/template.cfm?ID=1009

UC_6602: Query ICD Database.

This Use Case provides an interface to the ICD database.

ICD-9-CM[14]

The International Classification of Diseases, Ninth Revision, Clinical Modification (ICD-9-CM) is based on the World Health Organization's Ninth Revision, International Classification of Diseases (ICD-9). ICD-9-CM is the official system of assigning codes to diagnoses and procedures associated with hospital utilization in the United States. The ICD-9 is used to code and classify mortality data from death certificates.
The ICD-9-CM consists of:

- a tabular list containing a numerical list of the disease code numbers in tabular form;
- an alphabetical index to the disease entries; and
- a classification system for surgical, diagnostic, and therapeutic procedures (alphabetic index and tabular list).

The entire ICD-9-CM database is available for free download.[15]

ICD-10[16]

The Tenth Revision (ICD-10) differs from the Ninth Revision (ICD-9) in several ways although the overall content is similar: First, ICD-10 is printed in a three-volume set compared with ICD-9's two-volume set. Second, ICD-10 has alphanumeric categories rather than numeric categories. Third, some chapters have been rearranged, some titles have changed, and conditions have been regrouped. Fourth, ICD-10 has almost twice as many categories as ICD-9. Fifth, some fairly minor changes have been made in the coding rules for mortality.

[14] http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm
[15] Available at ftp://ftp.cdc.gov/pub/Health_Statistics/NCHS/Publications/ICD9-CM/2005/
[16] Available at http://www.cdc.gov/nchs/about/major/dvs/icd10des.htm

UC_6603: Generate Nutritional Profile.

This Use Case will allow User to create or edit a Nutritional Profile for an Individual.

The US Department of Agriculture provides freely-available data sources with the following information:

- NDB_No: 5-digit Nutrient Databank number that uniquely identifies a food item
- FdGrp_Cd: 4-digit code indicating food group to which a food item belongs
- FdGrp_Desc: Name of food group
- Food Description: Description of food item

USDA National Nutrient Database for Standard Reference - Release 18[17]

Release 18 of the USDA National Nutrient Database for Standard Reference: look up the nutrient content of 7,146 different foods.

USDA Database for the Added Sugars Content of Selected Foods.[18]

The table contains carbohydrate, total sugar, and added sugar values for 2,038 foods across 23 food groups. The carbohydrate and total sugar values are taken directly from the USDA National Nutrient Database for Standard Reference, release 18 (SR).

A Microsoft Excel version of this report can also be downloaded.[19]

[17] Available at http://www.ars.usda.gov/Services/docs.htm?docid=8964

[18] Nutrient Data Laboratory; Beltsville Human Nutrition Research Center (BHNRC); Agricultural Research Service (ARS); U.S. Department of Agriculture (USDA); Release 1, February 2006. Available at http://www.nal.usda.gov/fnic/foodcomp/Data/add_sug/addsug01.pdf

[19] Available at http://www.nal.usda.gov/fnic/foodcomp/Data/add_sug/addsug01.xls

UC_6604: Data Mine Published Research.

Open Access Journals

PLoS[20]

The Public Library of Science (PLoS) is a nonprofit organization of scientists and physicians committed to making the world's scientific and medical literature a public resource.

Directory of Open Access Journals[21]

The Directory of Open Access Journals service covers free, full text, quality-controlled scientific and scholarly journals. The project aims to cover all subjects and languages. There are now 2,298 journals in the directory. Currently 654 journals are searchable at article level. As of 07/2006, 101,536 articles are included in the DOAJ service.

There is a content search interface available at http://www.doaj.org/findarticles

Or, to extract information more dynamically, PHY systems will use the Open Archives Initiative Protocol for Metadata Harvesting ("OAI-PMH").[22] OAI-PMH provides an application-independent interoperability framework based on metadata harvesting. There are two classes of participants in the OAI-PMH framework:

- Data Providers administer systems that support the OAI-PMH as a means of exposing metadata
- Service Providers use metadata harvested via the OAI-PMH as a basis for building value-added services

http://www.openarchives.org/OAI/2.0/guidelines.htm
Implementation Guidelines

http://www.openarchives.org/OAI/2.0/guidelines-harvester.htm
Implementation Guidelines for the Open Archives Initiative Protocol for Metadata Harvesting
Guidelines for Harvester Implementers

PubMed[23]

Entrez Programming Utilities[24]

[20] Available at http://www.plos.org/
[21] Available at http://www.doaj.org/
[22] Available at http://www.openarchives.org/OAI/openarchivesprotocol.html
[23] Available at http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?DB=pubmed
[24] Available at http://eutils.ncbi.nlm.nih.gov/entrez/query/static/eutils_help.html

Guidelines Mining

Guidelines Clearinghouse

*** Centers for Disease Control and Prevention ***

Questions You May Want to Ask Your Child's Audiologist
http://www.qualitytools.ahrq.gov/summary/summary.aspx?view_id=1&doc_id=9290

Questions You May Want to Ask Your Child's Ear, Nose, and Throat (ENT) Doctor
http://www.qualitytools.ahrq.gov/summary/summary.aspx?view_id=1&doc_id=9289

Questions You May Want to Ask Your Child's Speech-Language Pathologist
http://www.qualitytools.ahrq.gov/summary/summary.aspx?view_id=1&doc_id=9291

Future Interfaces to Implement

Create specific "connectors" for each of the above data repositories to perform the required data mining.

The PHI-compliant system must be flexible enough to accommodate these "data mining connectors" to be able to extract data from additional sources of validated information.
The software development community should be free and capable to develop additional "data mining connectors" to their own chosen sources of data.

Z39.50 Standard

Industry standards such as the Z39.50[25] standard used in the publishing industry. A more recent version of that protocol is that referred to as the "Bath Profile."[26]

> .. 1. Introduction to the Profile
> This document identifies a subset of specifications from the Z39.50 Information Retrieval Protocol (ANSI/NISO Z39.50/ISO 23950) for use in Z39.50 client and server software. Conformance to this profile's specifications will improve international or extranational search and retrieval among library catalogues, union catalogues, and other electronic resource discovery services worldwide…
>
> 2. Purpose and Scope
> The purpose of the Bath Profile is to identify those features of the Z39.50 standard that are required to allow effective use of Z39.50 software in a range of library applications, including search and retrieval of bibliographic data from library catalogues; transfer of holdings information; cross-domain searches between libraries, museums and archives; updating union catalogues; item ordering and document delivery.
>
> .. 3. Functional Requirements
> This section identifies the functional requirements informing the Z39.50 specifications in this release. These requirements focus on search and retrieval between library catalogues, the search and retrieval of bibliographic holdings information, and cross-domain search and retrieval for resource discovery. The requirements detailed in the sections below comprise three Functional Areas:
> - Functional Area A for Basic Bibliographic Search & Retrieval, with Primary Focus on Library Catalogues
> - Functional Area B for Bibliographic Holdings Search & Retrieval
> - Functional Area C for Cross-Domain Search & Retrieval.

[25] http://en.wikipedia.org/wiki/Z39.50
[26] Available at http://www.collectionscanada.ca/bath/bp-current.htm

Fedora[27]

"Fedora open source software gives organizations a flexible service-oriented architecture for managing and delivering their digital content. At its core is a powerful digital object model that supports multiple views of each digital object and the relationships among digital objects. Digital objects can encapsulate locally-managed content or make reference to remote content. Dynamic views are possible by associating web services with objects. Digital objects exist within a repository architecture that supports a variety of management functions. All functions of Fedora, both at the object and repository level, are exposed as web services. These functions can be protected with fine-grained access control policies.

This unique combination of features makes Fedora an attractive solution in a variety of domains. Some examples of applications that are built upon Fedora include library collections management, multimedia authoring systems, archival repositories, institutional repositories, and digital libraries for education."

[27] Available at http://www.fedora.info/

UC_6605: Alerting Mechanism.

This Use Case will process the information generated by the system and will inform the User that some of such information is ready for the User's review.

- Passively (only when User logs in to Dashboard)
- Actively to mobile device, if requested

UC_6606: Manage Access Control Lists.

Access Controls Lists ("ACLs") are used to used to manage a matrix of user, role, and access rights to each section of the system. ACLs can also provide increased granularity by allowing the assignment of access rights on a per individual basis, beyond the traditional role-based access control process.

Manage users
Manage roles
Manage ACLs

UC_6620: Use Information Source Database.

This UC will allow the system to use the library of validated sources of information provided with the system. This library provide the system with pointers to the original sources of such data in order to maintain data consistency. The system operator will also be able to modify (add, edit, delete) such library.

It is important to identify the specific communication protocol ("API") required by the PHI-compliant system to connect to such source of information (either existing or needed to be developed).

The entries in this library will be grouped by section:

- Free journals
- Nutrition
- Validated government databases
- Other query-ready sources of information

This library should be language- and country-specific to allow system administrator to customize such library to the system user's local needs.

Appendix A: Introduction to UML.

This section provides a brief introduction to the Unified Modeling Language ("UML"). UML will be used extensively throughout this document, and throughout the development of the reference implementation of the PHI system.

[NOTE: The material below is extracted from an excellent document by Sinan Si Alhir.[28]]

> The UML is an evolutionary general-purpose, broadly applicable, tool-supported, and industry-standardized modeling language for specifying, visualizing, constructing, and documenting the artifacts of a system-intensive process. The language is broadly applicable to different types of systems (software and non-software), domains (business versus software), and methods and processes. The UML enables and promotes (but does not require nor mandate) a use-case-driven, architecture-centric, iterative, and incremental process that is object oriented and component based. The UML enables the capturing, communicating, and leveraging of knowledge: models capture knowledge (semantics), architectural views organize knowledge in accordance with guidelines expressing idioms of usage, and diagrams depict knowledge (syntax) for communication.
>
> ..Models are complete abstractions of systems. Models are used to capture knowledge (semantics) about problems and solutions. Architectural views are abstractions of models. Architectural views are used to organize knowledge in accordance with guidelines expressing idioms of usage. Diagrams are graphical projections of sets of model elements. Diagrams are used to depict knowledge (syntax) about problems and solutions.
>
> Within the fundamental UML notation, concepts are depicted as symbols and relationships among concepts are depicted as paths (lines) connecting symbols.
>
> ..Use Case Diagrams
> To successfully apply use case diagrams, we must first understand the types of elements used in use case diagrams.
>
> *Actors*
> Actor classes are used to model and represent roles for "users" of a system, including human users and other systems. Actors are denoted as stick person icons.
> *..Use Cases*
> Use case classes are used to model and represent units of functionality or services provided by a system (or parts of a system: subsystems or classes) to users… Use case classes have use case instances or objects called scenarios that represent specific interactions. Scenarios represent a singe sequence of messages and actions.
> *.. Relationships*
> Association relationships between actor classes and use case classes are used to indicate that the actor classes participate and communicates with the system containing the use case classes.
>
> ..Use Cases
> When modeling use cases, we ought to be aware of the following guidelines:
>
> - Use cases should be named using verb-noun phrases.
> - Use cases should be described, indicating how they are started and end, any conditions that must be satisfied before the use case starts (pre-conditions), any conditions that must be satisfied when the use case ends (post-conditions), the sequence of exchanged messages and performed actions, the data exchanged, and any non-functional characteristics (reliability, performance, supportability, etc. constraints). This description may be captured using text and other UML diagrams.
> - Use cases define the scope of a system and define the functionality provided by the system and those elements on which the system depends in order to provide the functionality…
> - Use cases should facilitate actors in reaching their goals. Use cases are system functionality or responsibilities (requirements) that actors use in order to reach or satisfy their goals. Use cases are not simply actor goals…
> - Use cases should facilitate the architecture of a system. Use cases may be organized and partitioned using includes, extends, and generalization relationships to identify, extract, and manage common, optional, and similar functionality…

[28] Source: "Understanding Use Case Modeling," available at http://www.methodsandtools.com/PDF/DMT0100.pdf (p. 10-16). Sinan's personal website is available at http://home.comcast.net/~salhir/

- Use cases provide flexibility and power throughout the life-cycle process. They provide the freedom to work with a use case as a whole or any subset of a use case via scenarios…
- Use cases may be used as the basis for planning. Time and resource estimates may be associated with use cases…
- Use cases may be used as the basis for analysis, design, and implementation..
- Use cases may be used as the basis for testing…
- Use cases may be used as the basis for documentation since use cases capture how users will use the system.

UML Based Processes for OOA&D[29]

It is important to understand that UML is a notation for OOA&D. It does not prescribe any particular process. Whatever process is adopted, it must take the system being constructed through a number of phases.

- **Requirements Capture**. This is where we identify the requirements for the system, using the language of the problem domain. In other words we describe the problem in the "customer's" terms.
- **Analysis**. We take the requirements and start to recast them in the language of a putative solution -the solution domain. At this stage, although thinking in terms of a solution, we ensure we keep things at a high level, away from concrete details of a specific solution—what is known as abstraction.
- **Design**. We take the specification from the Analysis phase and construct the solution in full detail. We are moving from abstraction of the problem to its realization in concrete terms.
- **Build Phase**. We take the actual design and write it in a real programming language. This includes not just the programming, but the testing that the program meets the requirements (verification), testing that the program actually solves the customer's problem (validation) and writing all user documentation.

UML Tools

Software developers interested in a set of F/OSS UML tools might want to consider using ArgoUML[30] or StarUML[31], both are excellent (and free) F/OSS UML modeling tools.

NOTICE: All Use Cases for the PHI reference implementation will be developed using UML. Below please find an example of a UML-based Use Case developed using UML.

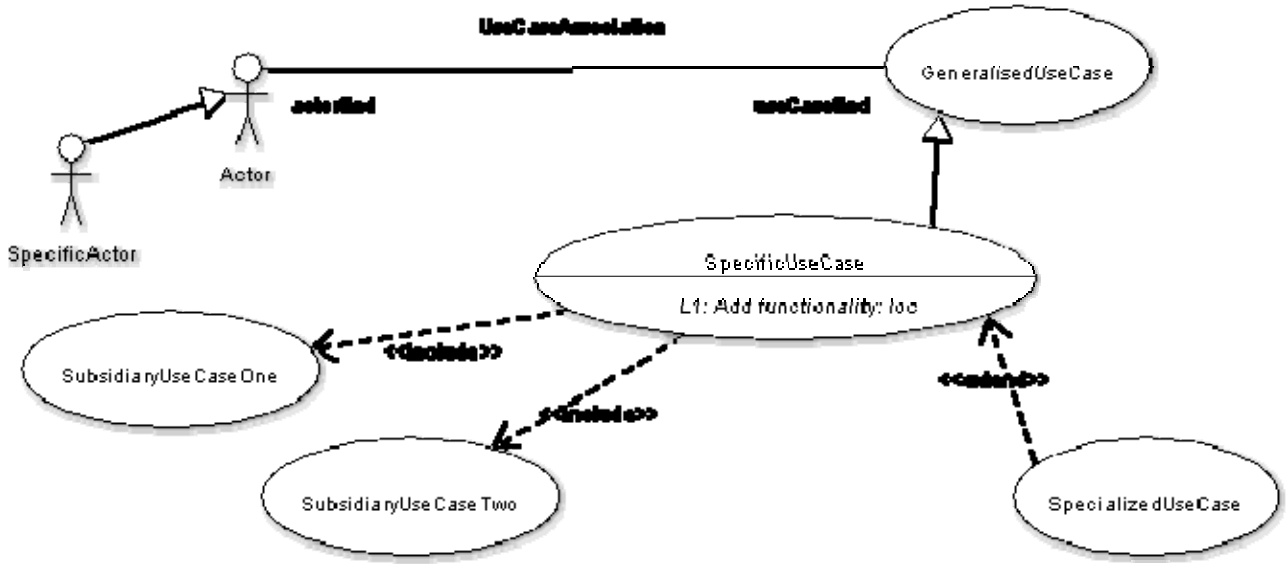

Figure 4: Possible Artifacts on a UML Use Case Diagram.[32]

[29] Source: http://argouml.tigris.org/documentation/defaulthtml/manual/ch02s03.html
[30] Available at http://argouml.tigris.org/
[31] Available at http://www.staruml.com
[32] Source: http://argouml.tigris.org/documentation/defaulthtml/manual/ch16.html#d0e13806

You might want to read a good UML book or tutorial.[33]

Appendix B: A Development Process Using ArgoUML[34]

Requirements Capture
Our requirements capture will use the UML concept of Use Cases. Starting with a Vision Document we will see how Use Cases can be developed to describe all aspects of the system's behavior in the problem domain.

Analysis
During the analysis stage, we will introduce the UML concept of classes to allow us to build a top level view of the objects that will make up the solution—sometimes known as a concept diagram. We will introduce the UML sequence diagram and statechart diagram to capture requirements for the overall behavior of the system. Finally we will take the Use Cases from the requirements capture stage, and recast them in the language of the solution domain. This will illustrate the UML ideas of stereotyping and realization.

Design
We use the UML package diagram to organize the components of the project. We then revisit the class diagram, sequence diagram and statechart diagram, to show how they can be used recursively to design the complete solution. During this part of the process, we need to develop our system architecture, to define how all the components will fit together and operate. Although not strictly part of our process, we'll look at how the UML collaboration diagram can be used as an alternative to, or to complement the sequence diagram. Similarly we will look at the UML activity diagram as an alternative or complement to the statechart diagram. Finally we shall use the UML deployment diagram to specify how the system will actually be realized.

Build
UML is not really concerned with code writing. However at this stage we will show how ArgoUML can be used for code generation. We will also look at how the UML Use Case Diagram and Use Case Specification are invaluable tools for a test program.

Cognitive Psychology and ArgoUML[35]

2.4.1.1. Theory
ArgoUML is particularly inspired by three theories within cognitive psychology: i) reflection-in-action, ii) opportunistic design iii) and comprehension and problem solving.

2.4.1.1.1. Reflection-in-Action
This theory observes that designers of complex systems do not conceive a design fully-formed. Instead, they must construct a partial design, evaluate, reflect on, and revise it, until they are ready to extend it further.
As developers work hands-on with the design, their mental model of the problem situation improves, hence improving their design.

2.4.1.1.2. Opportunistic Design
A theory within cognitive psychology suggesting that although designers plan and describe their work in an ordered, hierarchical fashion, in reality, they choose successive tasks based on the criteria of cognitive cost.
Simply stated, designers do not follow even their own plans in order, but choose steps that are mentally least expensive among alternatives.

2.4.1.1.3. Comprehension and Problem Solving
A design visualization theory within cognitive psychology. The theory notes that designers must bridge a gap between their mental model of the problem or situation and the formal model of a solution or system.
This theory suggests that programmers will benefit from:

[33] A free UML tutorial is available at http://www.cragsystems.co.uk/uml_tutorial_download.htm
[34] Source: http://argouml.tigris.org/documentation/defaulthtml/manual/ch02s03.html
[35] Source: http://argouml.tigris.org/documentation/defaulthtml/manual/ch02s04.html

- Multiple representations such as program syntactic decomposition, state transitions, control flow, and data flow. These allow the programmer to better identify elements and relationships in the problem and solution and thus more readily create a mapping between their situation models and working system models.
- Familiar aspects of a situation model, which improve designers' abilities to formulate solutions.

Vision Document[36]

Typical sections of this document would be as follows.

- **Summary**. A statement of the context, problem and solution goals.
- **Goals**. What are we trying to achieve (and how do we wish to achieve it).
- **Market Context or Contractual Arrangements**. For a market led development, this should indicate target markets, competitive differentiators, compelling events and so forth. For a contractual development this should explain the key contractual drivers.
- **Stakeholders**. The users (in the widest sense) of the system. Many of these will map in to actors, or control equipment that maps into actors.
- **Key Features**. At the very highest level what are they key functional aspects of the problem/desired solution. These will largely map down to the use cases. It is helpful to give some prioritization here.
- **Constraints**. A high level view of the non-functional parameters of the system. These will be worked out in detail in the supplementary requirements specification.
- **Appendix**. A listing of the actors and use cases that will be needed to meet this vision. It is useful to link to these from the earlier sections to ensure comprehensive coverage.

[36] Source : http://argouml.tigris.org/documentation/defaulthtml/manual/ch03s03.html

Appendix C: Proposed Development Model

It is proposed that the PHY reference implementation will include the following element:

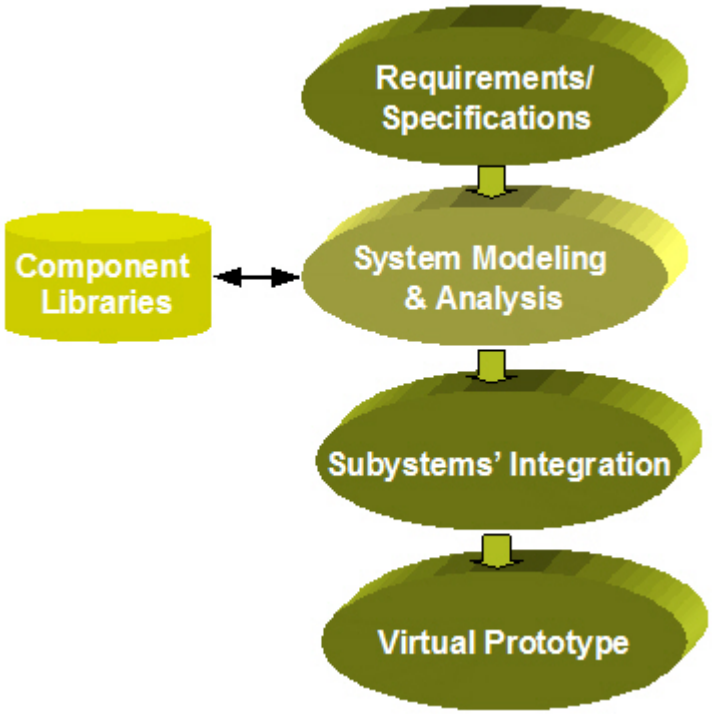

Figure 5: Requirements to Prototype Process.[37]

Develop a methodology and tools that help translate product requirements to architectural requirements without the need for extensive and time consuming simulations.

Build a three-level (user, architect, and programmer) component library of system components commonly used in building applications, algorithms, and architectures.

Development Process

[NOTE: Some ideas presented in this section were generated based on a discussion with Hari Kalva, Ph.D.[38]]

Given the variety of individuals participating in this project, the project will segment the required activities to allow each paticipant to be able to contribute the most to the project. These are some of the roles that participants in the project can assume:

User Role

[37] From materials developed by Ionut Cardei, Assistant Professor at Florida Atlantic University. http://www.cse.fau.edu/~icardei/

[38] Assistant Professor in the Dept. of Computer Science and Engineering at Florida Atlantic University. http://www.cse.fau.edu/~hari

- Requirements
- Validation
- Testing

Architect Role

- Design components
- Design and test interfaces

Programmer Role

- Create software, modules
- Conduct software-level testing

I need to back-track: having a clear idea of the end goal (virtual prototype), I must now define all the intermediate steps to ensure that all participants contribute in an appropriate and productive manner.

b.) Define sub-systems:

- Componentization
- software-defined health advisor?
- interfaces!

f.) >> Create "user animations:" turn UML Use Cases, and Operating Cases, into "animations" to allow target stakeholders to "experience" the system and provide feedback on it. <<
* Explore conversion tools: UML to XMI

- - -

g.) Test-driven Design Methodology

+ Start very small, design the core elements of the entire system.
+ Test those core elements:
IF fail predictably:
{
* Code to address the failures
* Re-test
}
ELSE: add incremental functionality

Structured Analysis[39]

In structured analysis there are three orthogonal views:

[39] Source: http://www.cragsystems.co.uk/ITMUML/index.htm; item 1.4

- The functional view, made up of data flow diagrams, is the primary view of the system. It defines what is done, the flow of data between things that are done and provides the primary structure of the solution. Changes in functionality result in changes in the software structure.
- The data view, made up of entity relationship diagrams, is a record of what is in the system, or what is outside the system that is being monitored. It is the static structural view.
- The dynamic view, made up of state transition diagrams, defines when things happen and the conditions under which they happen.

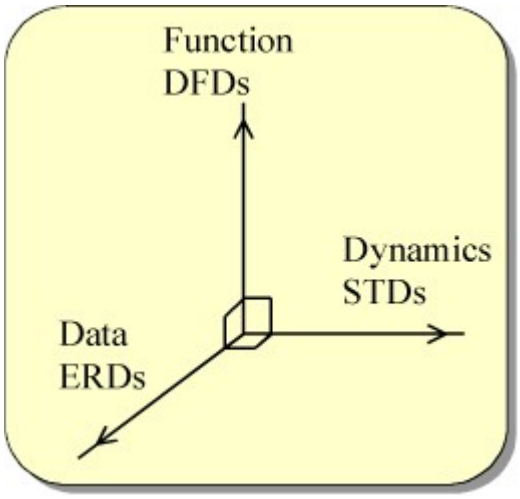

Figure 6: Axis of Structured Analysis.

Screen Prototyping[40]

Screen prototyping can be used as another useful way of getting information from the users. When integrated into a UML model:

- The flow of the screen is made consistent with the flow of the use case and the interaction model.
- The data entered and displayed on the screen is made consistent with the object model.
- The functionality in the screen is made consistent with the interaction and object models.

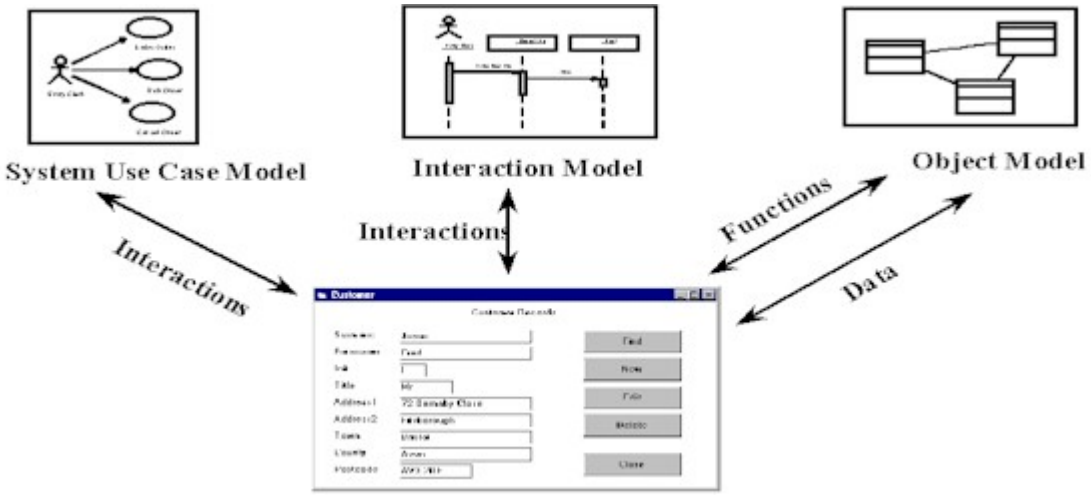

Figure 7: Screen Prototyping.

[40] Source: http://www.cragsystems.co.uk/ITMUML/index.htm, item 4.15.

Appendix D: Additional Resources

This is a list of additional resources that might be of interest to readers of this document.

- Introducing SPARQL: Querying the Semantic Web.
  http://www.xml.com/pub/a/2005/11/16/introducing-sparql-querying-semantic-web-tutorial.html
- UML Tutorials:
  http://bdn.borland.com/article/images/31863/usecase.html
  ftp://ftp.software.ibm.com/software/rational/web/whitepapers/2003/intro_rdn.pdf
  http://www-128.ibm.com/developerworks/rational/library/769.html