**Office of the National Coordinator (ONC)**

**For Health IT**

**Federal Health Architecture (FHA)**

**Information Modeling (IM) Project and Federal Health Information Model (FHIM)**

**Requirements for the S&I Framework Repository**

**April 21, 2011**

**Release 1.2 Draft**



Table of Contents

[List of Figures and Tables 2](#_Toc291857734)

[Acknowledgements 3](#_Toc291857735)

[Requirements Scope 4](#_Toc291857736)

[Assumptions 4](#_Toc291857737)

[Approach 4](#_Toc291857738)

[FHIM Modeling Repository General Requirements 8](#_Toc291857739)

[Version Control 9](#_Toc291857740)

[Document Management 9](#_Toc291857741)

[Issue Tracking (Tracker) 9](#_Toc291857742)

[Release Management 11](#_Toc291857743)

[Integration 11](#_Toc291857744)

[S&I Repository (SIR) Requirements 12](#_Toc291857745)

[Release Management: Publishing FHIM artifacts to the S&I Framework Wiki 12](#_Toc291857746)

[Issue Tracking (Tracker) 12](#_Toc291857747)

[Glossary of Terms 13](#_Toc291857748)

List of Figures and Tables

[Figure 1: Lifecycle of an S&I Framework Initiative](#TOC290909935) 6

[Figure 2: S&I Framework Initiative Phases](#TOC290909852) 6

[Figure 3: FHIM Related Artifacts](#TOC290909936)..................................................................................................................7

[Table 1: Artifacts](#Table1) 8

[Figure 4: ICDE – GForge – Hosting the FHIM Model Repository](#TOC290909937) 9

[Figure 5: Example issue state machine](#TOC290909938) 12

# Acknowledgements

This document was developed as part of a two month effort to determine how best to integrate FHA Information Modeling (IM) Project and Federal Health Information Model (FHIM) with the S&I Framework. The work was carried out under the direction and support of Anand Basu Acting Director of FHA.

# Requirements Scope

This document defines a set of requirements for the S&I Framework repository process to support the artifacts of the IM Project and the FHIM modeling effort to help ensure there is traceability between the two and integration of the IM Project and FHIM within the S&I Framework. The intent is to manage the process and artifacts in as automated and integrated a manner as possible while providing full life-cycle support.

# Assumptions

At the beginning of this pilot project, the FHIM modeling team met with the S&I Repository team to determine whether we could leverage the S&I Framework Repository for the FHIM modeling environment. The S&I Repository team recommended a separate repository be used for the FHIM working environment. Final FHIM artifacts (models, harmonized standards, publications, etc.) will be published to the S&I Repository for public consumption.

* The FHIM modeling team will continue to have access to a fully-fledged source forge repository (either the one currently in use from [Open Health Tools](https://www.projects.openhealthtools.org/sf/projects/forge/) or one provided by the Office of Interoperability and Standards (OIS)). The FHA Information Modeling project uses tools (Model Driven Health Tools (MDHT) provided by OIS, which in conjunction with the repository comprise an Integrated Collaborative Development Environment (ICDE). For the purposes of this document, the repository will be called the FHIM Model Repository (private repository)
* The S&I Framework Communities of Interest participants will have access to the IM Project/FHIM artifacts through the S&I Repository (SIR) hosted on the Alfresco Share Enterprise Content Management System (ECM) through the S & I Framework Initiatives portal (wiki/public repository)
* A workflow will be developed to integrate the public and private repositories using manual and automated processes

# Approach

The Standards and Interoperability (S&I) Framework is a set of integrated functions, processes and tools under development and guided by healthcare and technology industry experts to achieve harmonized semantic and structural interoperability for healthcare information exchange. The lifecycle of an S&I Framework initiative is depicted in [Figure 1](#TOC290909935).

The IM Project/FHIM are intended to support the lifecycle of these initiatives and to encourage a model-driven approach within the S&I Framework. A model-driven approach helps ensure end-to-end traceability between business requirements (use cases and functional requirements) and the standards and models selected to fulfill those requirements in the form of information models, terminology models/value sets and interoperability specifications. To complete the end-to-end traceability cycle, artifacts generated from the FHIM ICDE (stored in the FHIM Model Repository) can be used in the certification and testing process where test cases are based on the use cases.

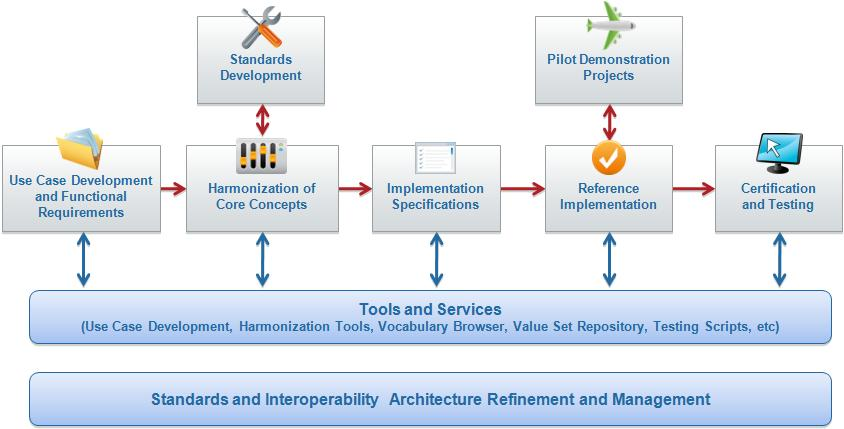


Figure 1: Lifecycle of an S&I Framework Initiative

[Figure 2](#TOC290909852) illustrates the typical activities that take place during each phase of an S&I Initiative. Many of the artifacts generated during the Discovery, Implementation and Pilot phases by the public (stored in the SIR) are related to artifacts created by the FHIM modeling team (stored in the FHIM Model Repository).

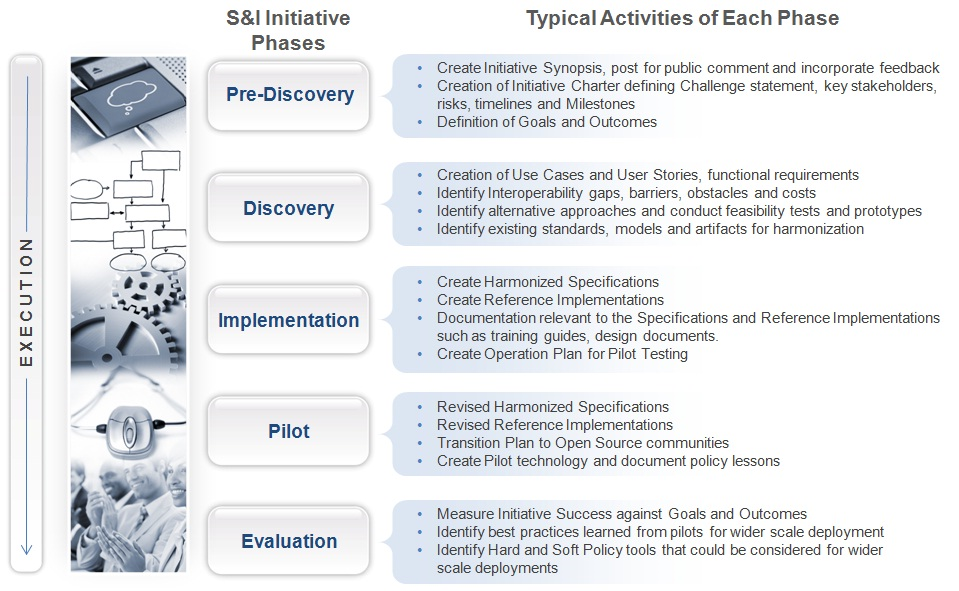


Figure 2: S&I Framework Initiative Phases

The IM Project/FHIM produces and/or contributes to artifacts that are generated and maintained during the activities performed in the Discovery, Implementation and Pilot phases.

The information models and standards will be published and stored in the SIR and made available to the public (S&I Communities of Interest) via the S&I Framework wiki portal.

[Figure 3](#TOC290909936) depicts the type/format of the artifacts maintained in the FHIM Model Repository:

 Figure 3: FHIM-Related Artifacts

The FHIM Repository Format column in [Table 1](#Table1) reflect the IM Project/FHIM and Model Driven Health Tools (MDHT) related artifacts.

Understanding the trace dependencies between process design and implementation is vital for change impact analysis, change propagation, documentation and many other S&I Initiative activities. The Tools and Services of the S&I Framework are designed to facilitate the development lifecycle and to maximize re-use of concepts and components. It is envisioned that repository tools will be used to browse and select the appropriate standards and artifacts.

In an ideal world, the cycle and flow of information between the public-facing (SIR) and private (FHIM Model) repositories would be integrated using automated and manual processes, starting and ending at the S&I Framework public portal. Metadata accompanying the artifacts shared between the SIR and the FHIM Model repositories would ensure traceability between modeling artifacts and between the business requirements and a particular version of the FHIM or a FHIM artifact.

Table 1 Artifacts

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Artifact** | **SIR Format** | **FHIM**  **Repository Format** |
| **Discovery** | * **Use Cases Model** – Use Case Diagram, Activity Diagrams & Flows (Computationally Independent Model) | * XML Metadata Interchange (XMI) output | * **UML Model** files (e.g., emx, efx, uml) – these are XML files that cannot be easily merged and should be treated like binary files |
|  | * **Functional Requirements** | * Excel Spreadsheet with mapping to Use Cases. | * Excel Spreadsheet with mapping to Use Cases * UML models containing UML Activity Diagrams. Business Process diagrams |
|  | * Interoperability Gaps and Barriers | * Wiki document | * Wiki document |
|  | * Alternative Approaches and Feasibility Prototypes | * Wiki document | * Wiki document |
|  | * Models and Standards (Platform Independent Model) | * XMI Output for Models. Wiki document for narrative text. | * Standard artifacts including PDF, HTML, XSDs, etc. * UML model files (e.g. emx, efx, uml) –these are XML files that cannot be merged very easily and should be treated like binary files |
| **Implementation** | * Harmonized Specifications (Platform Independent Model) | * XMI Output for Models. Wiki document for narrative text. | * **UML Models** * **Model Publication** (e.g., DITA, PDF) |
|  | * Reference Implementation (Platform Specific Model) | * Source code. * XMI output for Design Documents. | * **Implementation Artifacts**   Source code generated from models (e.g. Java, Referenced open-source libraries (e.g., OCL )   * Doclets/API Documentation * **Implementation Artifacts Documentation**   User documentation generated from models |
|  | * Test Harness | * TBD | * Not relevant to FHIM directly |
|  | * Validated Reference Implementation | * Source code. * XMI output for Design Documents. | * Source code. * XMI output for Design Documents. |
| **Pilot** | * Validated Reference Implementation | * Wiki document. | * Wiki document |
|  | * Revised Specifications and Models | * XMI Output for Models. Wiki document for narrative text. | * XMI Output for Models. Wiki document for narrative text. |
|  | * Revised Reference Implementation | * Source code * XMI output for Design Documents. | * Source code. * XMI output for Design Documents. |
|  | * Pilot Execution | * Wiki documents capturing pilot information and lessons learned. | * Wiki documents capturing pilot information and lessons learned. |
|  | * Policy and Technology Lessons Learned | * Wiki documents, Word/PDF documents. | * Wiki documents capturing policy and technology lessons learned. |

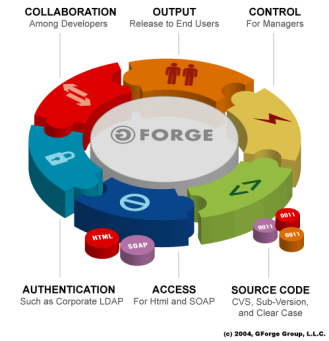
# FHIM Modeling Repository General Requirements

An Integrated Collaborative Development Environment (ICDE) including a repository such as the OpenHealthTools.org [Open Health Forge](https://www.projects.openhealthtools.org/sf/sfmain/do/viewProject/projects.fhims) illustrated in [Figure 4](#Figure4) is required by the FHIM modeling team in support of the *Collaboration, Output and Control* activities that comprise a model-driven approach to the S&I Framework. The benefits for using a source forge environment are:

* **Collaboration:** The tool prevents users from accidentally overwriting the changes of another, allowing many developers to work on the same artifacts without stepping on each other's toes.
* **History:** The tool tracks the complete history of the artifacts, including the exact changes which have occurred between releases and who made those changes.
* **Release notes generation:** Given the tracking of each change, the tool can be used to generate notes for releases which accurately capture all of the changes included in the new release.
* **Documentation and test management:** The tool can be used to manage not just Reference Implementations, but also test suites and documentation for those Reference Implementations.
* **Change notifications:** The tool can generate change notifications to keep interested members of the S&I Framework Communities of Interest informed when changes occur.

Access to this environment should be restricted to the modeling team and is not open to the S&I community at large.

Figure 4: ICDE – GForge – Hosting the FHIM Model Repository



The following capabilities are needed to develop models that support harmonization, implementation, terminology binding and other S&I Framework activities:

## Version Control

* 1. The repository must support version control for all modeling artifacts, e.g., model files, model fragments, model publications, terminology models
  2. The repository must support version control for source code, XML Schemas, and other implementation artifacts generated from models
  3. The repository must support version control for harmonization requirements (e.g. gaps, conflicts, overlaps) intended to be resolved by SDO/SROs based on FHIM analysis
  4. The repository should support versioned metadata just like file contents
  5. The repository should support directory versioning: a “virtual” versioned file system that tracks changes to files and whole directory trees over time. Files and directories are versioned

## Document Management

* 1. The repository must support versioning of all Developers’ and Modelers’ documentation through check-in / check-out functions. Modelers’ documents include:
     1. FHIM Style Guide
     2. Model Transformation Information
     3. How to use UML profiles and Vocabulary bindings
  2. The repository must allow users to retrieve previous versions and to continue work from a selected point if necessary
  3. The repository should store document metadata, including the date the document was stored and the identity of the user storing it. The metadata can be used to assist users in locating documents through keywords or other mechanisms
  4. The repository should support the publication of modeling artifacts for the purposes of peer or public reviewing, printing, etc. in a format that is read-only and not easily altered

## Issue Tracking (Tracker)

* 1. The FHIM Model Repository should be able to automatically receive issues that were initially submitted into the S&I Framework public portal by the S&I Initiatives Communities of Interest (e.g., enhancement/change requests based on new or modified use cases, FHIM model defects). Once the issue is approved by S&I Decision makers for further development, the issue along with the supporting documents from the SIR and their associated metadata should be automatically received into the Tracker function of the ICDE where it can be further triaged to the appropriate member of the FHIM modeling team.
  2. A process for managing issues within the FHIM Model Repository (ICDE) must be developed covering:
     1. Issue evaluation
     2. Issue assignment (or discard)
     3. Ability to associate issues with model artifacts
     4. Issue status updates to submitter closing the loop (issue received, issue assigned, issue complete, new release addressing issue)
  3. The FHIM Model Repository should allow the person assigned to resolve an issue to check-out components and associate those components with an issue entered into the SIR.
  4. The FHIM Model Repository should allow for the capture of Tracker metadata including issue state. Issues entered in the Tracker should include information about the issue. These metadata are required to determine how to resolve the issue:
     1. Identify the artifact affected (e.g., use case, model element, implementation artifacts, documentation)
     2. Identify whether the issue is a defect/bug, an enhancement
     3. Specify how critical the issue is and the consequences
     4. Support Issue Status information. Initially an issue may be “new” but then transitions to “assigned” once the issue has been approved for work. The issue transitions to “resolved” when the modeler or developer checks-in the resolution. The “resolved” issue is forwarded to QA and once the resolved issue has been verified by QA , the Issue Status transitions to “closed”



Figure 5: Example issue state machine

## Release Management

* 1. File releases corresponding to scheduled releases, defect fixes, etc. of the models
  2. FHIM Release Notes should be generated from the documentation, metadata and comments associated with the issues/enhancement requests that triggered changes to the model
  3. Comments entered during check-in become part of the release notes and the comments are linked to the issue – see [Integration](#Integration)

## Integration

* 1. Integration is required at several levels. More stringent manual processes are required when there are fewer points of integration (automation) between the FHIM Model Repository (ICDE) and the SIR (e.g., Alfresco Share ECM).
  2. The S & I Repository (SIR), supporting the Communities of Interest Content Management must be integrated through automated or manual processes with the modeling Integrated Collaborative Development Environment to ensure traceability between business requirements (enhancement requests, defects) and changes to the FHIM
  3. Feedback from S&I Communities of Interest must be forwarded to FHIM model developers once that feedback has been approved by S&I Decision makers. Feedback may include defect reports against the FHIM or change/enhancement requests based on new use cases. Ideally, once the issue has been determined to be related to the FHIM or FHIM-derived artifact, the S&I Framework review team should be able to enter the issue directly into the ICDE Tracker from the S&I Framework portal (wiki or SIR).
     1. Example: Someone from the community (implementer, business analyst) or someone on an S&I Initiative reports an issue. Once the issue is deemed to be caused or related to the FHIM or FHIM-derived artifact, the issue is assigned to the FHIM modeling team for review. Once the ICDA review team agrees the issue is related to the FHIM, a task is assigned to someone on the FHIM modeling team for resolution.
  4. There should be integration between issue tracking, tasks, and file versions in the ICDE environment.
     1. Example: Once an issue is assigned to the FHIM team for resolution, the modeler/developer uses that issue to find the associated file(s) for check out from the ICD version control to make the necessary changes. Once those changes have been made and verified by the QA team, the “closed” issue is communicated back to the SIR where the loop can be closed with the issues submitter. This is a process as yet to be defined, but the point of integration is a notification that a particular issue has been resolved and the “fix” will be available in a future release. When a public release is scheduled, the issue appears as "resolved" in the public release documentation posted on the S&I Framework wiki site. Prior to the public release, the modeling team may choose to issue a private release for regression testing.
     2. This level of integration allows traceability from the issue to the file version that resolves the issue or fulfills the requirement

# S&I Repository (SIR) Requirements

The S&I Framework portal (wiki) is the point of entry for the S&I Communities of Interest (users) looking to learn about and to engage in these Initiatives. The following requirements are related to “integration” between the FHIM Model Repository and the SIR, hosted on the Alfresco Share ECM environment:

## Release Management: Publishing FHIM artifacts to the S&I Framework Wiki

1. There must be the ability to move artifacts between the SIR and FHIM Model Repositories
   1. FHIM artifacts ready for use and public distribution will be stored in the SIR repository and published (made available to the communities of interest) on the S&I Framework Wiki
2. FHIM artifacts published to the S&I Wiki will be read only
3. The SIR must support version control for FHIM artifacts
4. Communities of Interest need a search mechanism within the SIR to find Terminology Models, applicable standards and other FHIM-related artifacts
5. Artifacts residing in the SIR and FHIM Model Repository must be assigned a unique identifier that can be used for searching as well as tying related artifacts (use cases, functional requirements to FHIM-related activities and artifacts; issues entered into the S&I Framework issue tracker tied to FHIM artifacts

## Issue Tracking (Tracker)

1. Users should be able to enter feedback (issues, new requirements, etc.) into an “issue tracking system” through the S&I Framework public portal and to be relate that issue to its associated models, standards and other FHIM published artifacts
   1. Users should be able to associate an issues with the unique identifier (Smart ID) assigned to the artifact for which feedback (issue) is submitted
   2. Issues that have been entered into the S&I Framework issue tracker should be automatically forwarded to the FHIM Model team for triage once they have been reviewed and “approved” by the S&I Initiative

# Glossary of Terms

The following terms are defined for the purpose of clarifying their use throughout this repository requirements document.

Change Management

The term Change Management is used in a number of ways, but within the context of the S & I Framework, a good definition is: The systematic approach and application of knowledge, tools and resources to deal with change. Change management means defining and adopting corporate strategies, structures, procedures and technologies to deal with changes in external conditions and the business environment.

Check-In

Check In is the act of putting content previously checked out of a content management system (or Subversion Repository) back into the system. Content is versioned when it is checked back in.

Check-Out

Check Out is the act of signing out content from a content management system (or Subversion Repository).

Computational Independent Model (CIM)

The Computational Independent Model is intended to illustrate exactly what a system is expected to do by representing it in the environment in which it will operate. However, it does not include details of the system structure. The CIM consists of a vision document, use cases, functional requirements and domain models. A CIM is useful in understanding the problem and also provides a source of shared vocabularies for use in other models. The CIM should be traceable to the subsequent PIM and PSM constructs.

Configuration Management

Configuration Management Systems are used to keep track of large projects. Although version control, which maintains a database of revisions, is part of the system, a full-blown software configuration management system (SCM system or CM system) automatically documents all components used to build executable programs. It is able to recreate each build as well as to recreate earlier environments in order to maintain previous versions of a product. It may also be used to prevent unauthorized access to files or to alert the appropriate users when a file has been altered.

Increasingly, parts of version control and configuration management are being added to application development systems. Examples of stand-alone configuration management systems are PVCS, CCC/Harvest and ClearCase.

Constraint

A constraint is an expression of a business rule applied to an Information Exchange. It can restrict the values that appear within the exchange in a variety of different ways, and appear in both HITSP Specifications and in the standards those specifications select

Content Management

A content management system (CMS) is the collection of procedures used to manage work flow in a collaborative environment. These procedures can be manual or computer-based. The procedures are designed to do the following:

* Allow for a large number of people to contribute to and share stored data
* Control access to data, based on user roles (defining which information users or user groups can view, edit, publish, etc.)
* Aid in easy storage and retrieval of data
* Reduce repetitive duplicate input
* Improve the ease of report writing
* Improve communication between users

In a CMS, data can be defined as nearly anything: documents, movies, pictures, phone numbers, scientific data, and so forth. CMSs are frequently used for storing, controlling, revising, semantically enriching, and publishing documentation. Serving as a central repository, the CMS increases the version level of new updates to an already existing file. Version control is one of the primary advantages of a CMS.

Document Management

A document management system (DMS) is a [computer system](http://en.wikipedia.org/wiki/Computer_system) (or set of computer programs) used to track and store [electronic documents](http://en.wikipedia.org/wiki/Electronic_document) and/or [images](http://en.wikipedia.org/wiki/Digital_image) of paper documents. It is usually also capable of keeping track of the different versions created by different users. The term has some overlap with the concepts of [content management systems](http://en.wikipedia.org/wiki/Content_management_system).

Dynamic Binding

A dynamically bound value set has its definitions fixed, but the values in the set may vary as new versions of the code system on which they are based are released. Intensional value sets are often dynamically bound.

File Locking

File locking is a mechanism that restricts access to a computer file by only allowing one user or process access at any specific time. The purpose of locking is to prevent the classic interceding update scenario (see race condition).

FHIM Model Repository

This is the GForge repository that provides the version control and collaboration ­­capabilities required to support FHIM modeling activities and artifacts.

Harmonization

Harmonization is the name given to the effort by industry to replace the variety of product standards and other regulatory policies adopted by nations, in favor of uniform global standards. Usually used to in the context of trade agreements, harmonization has recently been adopted by the United States government to refer to information technology standards.

Issue Tracking (See Tracker)

Issue tracking is the initial recording of a problem or concern and the tracing of that problem through the various states involved to resolve the problem. And issue tracking system (ITS) is a software application that helps to automate the issue tracking process. Issues can be tracked in various ways: by priority status, issue owner (user submitting the issue), assigned (person responsible for resolving the issue) or other customized criteria.

Metadata

Metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource. Metadata is often called data about data or information about information. There are three main types of metadata:

• Descriptive metadata describes a resource for purposes such as discovery and identification. It can include elements such as title, abstract, author, and keywords.

• Structural metadata indicates how compound objects are put together, for example, how pages are ordered to form chapters.

• Administrative metadata provides information to help manage a resource, such as when and how it was created, file type and other technical information, and who can access it. There are several subsets of administrative data; two that sometimes are listed as separate metadata types are:

* + Rights management metadata, which deals with intellectual property rights
  + Preservation metadata, which contains information needed to archive and preserve a resource.

Model Driven Architecture

Model-driven architecture (MDA) is a software design approach for the development of software systems. It provides a set of guidelines for the structuring of specifications, which are expressed as models. Model-driven architecture is a kind of domain engineering, and supports model-driven engineering of software systems. It was launched by the Object Management Group (OMG) in 2001.

Model Driven Health Tools (MDHT)

The Model-Driven Health Tools (MDHT) Project focuses on the development and promotion of model-driven Health Information standards within the standards community by providing a unified set of modeling tools for standards organizations and standard implementers to design, publish, and implement standards such as Clinical Document Architecture all from a UML model.  
https://www.projects.openhealthtools.org/sf/projects/mdht/

Platform Independent Model (PIM)

A Platform Independent Model (PIM) refers to a model of a software or business system that is independent of the specific technological platform used to implement it. A PIM consists of an interface diagram, class diagrams, a components diagram demonstrating traceability between the Computational Independent Model (CIM) and PIM, behavior diagrams and business rules. A PIM is transformed into a Platform Specific Model (PSM).

Platform Specific Model (PSM)

A Platform Specific Model (PSM) is a model of a software or business system that is linked to a specific technological platform (e.g. a specific programming language, operating system or database). It combines the specifications of the PIM with the details that specify how that system uses a particular platform. Therefore, a PSM is produced through the transformation of the PIM into a PSM. It consists of web services, schema, WSDL, and a components diagram showing traceability between the PIM and PSM. It will be an implementation if it provides all the information needed to construct a system into operation (e.g. Java code, SOAP over HTTP, implementation guidance, etc). Platform-specific models are indispensable for the actual implementation of a system.

Race Condition

In file systems, two or more programs may "collide" in their attempts to modify or access a file, which could result in data corruption. File locking provides a commonly-used solution. A more cumbersome remedy involves organizing the system in such a way that one unique process (running a daemon or the like) has exclusive access to the file, and all other processes that need to access the data in that file do so only via inter-process communication with that one process (which of course requires synchronization at the process level).

Release Notes

Release notes are documents that are distributed with software products, often when the product is still in the development or test state (e.g., a beta release). For products that have already been in use by clients, the release note is a supplementary document that is delivered to the customer when a bug is fixed or an enhancement is made to the product.

Repository

In information technology, the term repository refers to a central place in which an aggregation of data is kept and maintained in an organized way, usually in computer storage. A repository may be directly accessible to users or may be a place from which specific databases, files, or documents are obtained for further relocation or distribution in a network.

S & I Repository (SIR)

The SIR provides the repository capabilities required to support the ONC S&I Framework initiatives and will be used by the S&I Initiative Communities of Interest. The SIR is the repository of specification related content, and will also feature abilities for domain and role contextual searching, content cross-linking, version management and multi-format publishing.[[1]](#footnote-2) The SIR is a browser-based platform for content management and collaboration.

Static Binding

The values in the value set are fixed until a new version of the value set is released. Extensional Value Sets are typically statically bound. When an intensional value set is statically bound, the version of the code system being used must be specified before the members of the value set can be computed.

Subversion/Subversion Repository

Subversion is an open source control system. A Subversion repository is used to manage the files, folders, directories and the modification made to those files, folders and directories over a period of time. The subversion repository is like an ordinary file server, except that it remembers every change ever made to the files and directories within. Many think of Subversion and version control systems in general as a sort of “time machine” because it allows recovery of older versions of files and to examine the history of how, when and who changed the data.. Subversion is a general system that can be used to manage any collection of files, including source code.

Tracker (See Issue Tracking)

Tracker refers to the capability within the ICDE for automating the issue tracking process. Issues can be tracked in various ways: by priority status, issue owner (user submitting the issue), assigned (person responsible for resolving the issue) or other customized criteria.

Version Control

Version control is the art of managing changes to information. Version Control refers to the management of source code, documents, graphics, binary and related files (artifacts) involved in software development projects. Version-control software provides a database that is used to keep track of the revisions (different versions) made to these artifacts by all contributors.

Version control also allows users to identify differences between versions of an artifact or file and to fall back to any version of a particular file. Also Version Control Systems have methods for preventing two developers working on the same file from overwriting the other's changes.

## 

1. http://jira.siframework.org/wiki/pages/viewpage.action?pageId=4195259 [↑](#footnote-ref-2)