Purpose:
This document provides a high level description of the design and implementation of Laika, an open source electronic health record (EHR) testing framework.

Intended audience:
This document has been written for consumption by software engineers, software architects, user interface designers.

Rob Dingwell
Andy Gregorowicz
Rob McCready
Harry Sleeper
Tim Taylor
Amit Trivedi
Dennis Wilson

Document Revision LDD_003
January 29th, 2008
# Document Revision History

<table>
<thead>
<tr>
<th>Revision Number</th>
<th>Date</th>
<th>Revision Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDD_000</td>
<td>10/26/2007</td>
<td>First draft of the design document to vet material with CCHIT lead technologists</td>
</tr>
<tr>
<td>LDD_001</td>
<td>11/7/2007</td>
<td>Version approved for public release to coincide with the LAIKA project page</td>
</tr>
<tr>
<td>LDD_002</td>
<td>11/14/2007</td>
<td>Minor updates to address feedback from MITRE principle systems engineer Tim Taylor</td>
</tr>
<tr>
<td>LDD_003</td>
<td>1/29/2007</td>
<td>Updated to reflect latest changes in the Laika v1.0 functionality for CCHIT alpha testing</td>
</tr>
</tbody>
</table>

**Figure 1 – LAIKA Design Document Revision History**
1 Introduction

This document provides a high level description of the planned design and implementation of LAIKA, an open source electronic health record (EHR) testing framework. LAIKA will be the software testing tool used by the Certification Commission for Healthcare Information Technology (CCHIT) to automate validation of candidate EHRs. The latest information about LAIKA will be available at the project’s website http://projectlaika.org

1.1 Background

The public and completely open testing criteria used by the Certification Commission for Healthcare and Information Technology (http://www.cchit.org) has greatly contributed to their successful EHR interoperability testing and certification. Any interested vendor, organization, or individual has free and unfettered access to the evaluation criteria used in the CCHIT interoperability certification process.

This "open-book testing" strategy facilitates the desired end goal of more accurate, efficient, and portable EHRs for the United States. CCHIT's open testing criteria have provided EHR vendors with better situational awareness into the functions required for their products, both for current and future EHR certification milestones.

Open source software principles and practices have historically allowed The MITRE Corporation (http://www.mitre.org) to demonstrate and share both conceptual and detailed design examples of software services with a broad community including industry, government, and academia. It has been a successful technology transfer and community collaboration strategy for MITRE for many years.

LAIKA was created as a joint effort between CCHIT and The MITRE Corporation to develop an open source EHR interoperability testing framework. LAIKA will initially be designed to support interoperability testing of patient registration, medication and allergy information, aligned with the functional requirements that are defined by CCHIT for 2008 interoperability testing.

Over time, LAIKA will expand to support other Healthcare Information Technology standards that likewise have their criteria defined by the Certification Commission for Healthcare Information Technology.
1.2 LAIKA and Open Source

LAIKA is an open source software project licensed under the Apache 2.0 open source license (http://www.apache.org/licenses/LICENSE-2.0.html), a free software license authored by the Apache Software Foundation. The Apache License allows the user of the software the freedom to use it for any purpose. Users may distribute it, modify it, and distribute modified versions of the software.

The Apache License does not require modified versions of the software to be distributed using the same license nor even that it be distributed as free/open-source software. The Apache license only requires that a notice is kept informing recipients that Apache licensed code has been used. This license has been selected to allow maximum flexibility with the use of the LAIKA software by project contributors, EHR vendors and anyone interested in our work.

Any future software packages that are to be included in the LAIKA project will either be open source software projects or software that has its associated source code made freely available in the public domain. Software that does not support either of these criteria will not be considered for inclusion in the LAIKA software code. Further, the license of any component included in LAIKA must have a license compatible with the Apache 2.0 open source license.

1.3 Open Source Methodology

As an open source software project, LAIKA endeavours to create a community dedicated to addressing the need for systems to test interoperable EHRs. All activities on the LAIKA project are open. Not only does this include access to the underlying source code, but also planned features, forums, and all email discussions.

Embracing an open source model allows us to build trust with a community of users, collaborators, and contributors. With many eyes on all aspects of the LAIKA project, we also hope to increase the reliability, accuracy, stability, and auditability of the software.

Examples of some of the successful open source software projects embracing these tenets include GNU/LINUX: a free open source operating system, Apache: the most popular web server used on the internet, and Firefox: the second most popular web browser used on the internet.
Through this open source model, we hope to establish a level of trust among all members of the project’s community. It is our intention that LAIKA will demonstrate how open source software can likewise create a community and software that has a positive impact in the healthcare information technology domain of interoperable EHRs.

1.4 Project Governance

LAIKA is an active collaborative effort between CCHIT and MITRE. CCHIT is leading the functional requirements definition of the LAIKA testing framework. MITRE is leading the technical software design and is prototyping the software service.

All the software, documentation, and discussions around LAIKA are accessible to anyone. Further any individual is free to propose ideas, features and even software to the LAIKA project. Final commit rights to the main line of the LAIKA source code tree is currently gated by CCHIT and The MITRE Corporation.

Both CCHIT and MITRE will determine when LAIKA commit rights are granted to other organizations or individuals based on contributions to the LAIKA project. The eventual inclusion of additional individuals and organizations from the LAIKA community who will accelerate the development and impact of the LAIKA project, is a shared objective of both CCHIT and MITRE.

The Certification Commission is a private, not-for-profit organization whose mission is to accelerate adoption of health information technology in the United States. MITRE is a 501(c)(3) not-for-profit corporation that manages three Federally Funded Research and Development Centers (FFRDCs) and works in partnership with the government applying systems engineering and advanced technology to address issues of critical national importance.
1.5 Contact

To send general questions, comments and feedback about LAIKA, send email to talk@projectlaika.org. Read, write, join, and archive access to this list is open to the general public.

To send questions, comments, and feedback to the software engineers who are designing, developing and contributing software code to LAIKA, send email to dev@projectlaika.org. Read, write, join, and archive access to this list is open to the public.

Announcements about significant activities around LAIKA will be sent out to announcements@projectlaika.org. Registration for membership in this email list is available via http://projectlaika.org/subscribe.html. Read and archive access to the announcements list is open to the public. Write contributions are limited to members of the LAIKA steering committee. This may or may not include all committers to the LAIKA project source code tree.
2 Overview

CCHIT is recognized as the certification body for EHRs, and their networks. The criteria that the CCHIT uses is in the public domain. In 2008 CCHIT plans to support the certification of the interoperability of EHR systems.

LAIKA will implement automated testing of the certification criteria that CCHIT has defined and will be used as the interoperability testing framework supporting validation of the CCD data, detailed by CCHIT for EHR interoperability testing.

Below is a diagram detailing how LAIKA v1.0 will be used by CCHIT to support interoperability testing in 2008:

Figure 2 – CCHIT Interoperability Testing with LAIKA
Continuity of care is the initial focus for Laika v1.0 support of interoperability testing. The data standard to represent continuity of care data that CCHIT will require for EHR vendors will be a Continuity of Care Document (CCD) that has been constrained by the **HITSP C32 v2.1 specification** (CCD/C32).

There will be two types of tests that LAIKA will support with the release of LAIKA v1.0. These two types of tests are “File and Display” and “Generate and Format”.

### 2.1 “File and Display” Tests

For “File and Display”, LAIKA will allow its users to author a test that will account for all modules and elements identified in the HITSP C32 specification. The data entered in the LAIKA user interface will be provided to an EHR user via a single CCD/C32 XML document.

This CCD/C32 XML document may be transported by either physical medium, such as a flash drive; unregulated network communication, such as email or FTP; or via structured network communication, such as an XDS repository.

The transport of these CCD/C32 files to an EHR vendor will be an implementation that the Laika software will be agnostic of at the release of Laika v1.0. Future versions of Laika will support automating the transport of these CCD/C32 files out of Laika and directly into an EHR vendor. At the time that this document is being written, Cross-Enterprise Document Sharing (**XDS.b**) is the most likely implementation that Laika will use to support this.

The EHR vendor will demonstrate to the CCHIT jurors that they have received the CCD/C32 XML file, and that is has been filed it in their system. This will be visually confirmed by CCHIT jurors. To support this, CCHIT jurors will make printout the associated LAIKA File and Display Test Tracking Page (see Figure 13 for an example). Jurors will confirm that all of the data associated with this CCD/C32 test has been properly filed in the EHR vendor with a visual confirmation over GoToMeeting by checking off the data elements in their paper printout.

The final result of the CCHIT jurors will be communicated back to the CCHIT proctor (either by email, instant messenger, or phone).
Details of how LAIKA will support “File and Display” tests:

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Active User</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><strong>Proctor</strong></td>
<td>CCHIT proctor uses LAIKA user interface to create or modify a CCD/C32 test template for a candidate EHR under test. There will be CCD/C32 test templates provided inside LAIKA prior to a proctor logging in.</td>
</tr>
<tr>
<td>Step 2</td>
<td><strong>Juror</strong></td>
<td>LAIKA exports a CCD/C32 XML document with all the items from the original test included in a valid CCD/C32 XML file.</td>
</tr>
<tr>
<td>Step 3</td>
<td><strong>EHR Vendor</strong></td>
<td>EHR vendor administrator is sent the LAIKA CCD/C32 XML document and is asked to display and file it in their EHR system EHR under test</td>
</tr>
<tr>
<td>Step 4</td>
<td><strong>Juror</strong></td>
<td>The CCHIT juror will make a paper printout of the associated LAIKA CCD/C32 test from the browser via “File-&gt;Print”</td>
</tr>
<tr>
<td>Step 5</td>
<td><strong>Juror</strong></td>
<td>The CCHIT juror will communicate the pass/fail status of the EHR vendor back to the CCHIT proctor who is administering the test</td>
</tr>
<tr>
<td>Step 6</td>
<td><strong>Proctor</strong></td>
<td>The proctor will record the pass/fail status of the EHR in</td>
</tr>
</tbody>
</table>

*Figure 3 – LAIKA “File and Display” Test Interaction Sequence*
2.2 “Generate and Format” Tests

A requirement for LAIKA will be to ensure that data is maintained when transitioning from one EHR system to another. In this sense, v1.0 of LAIKA will verify that documents support the standards specified for continuity of care data and that the content of messages account for the items specified in test scripts produced by LAIKA users.

LAIKA will verify that a CCD/C32 document produced by an EHR system is valid with respect to the standard as specified by HL7, ASTM, and HITSP and will verify that the content of the document addresses all of the elements that the proctor has defined.
Details of how LAIKA will support “Generate and Format” tests:

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Active User</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Proctor</td>
<td>CCHIT proctor uses LAIKA user interface to create or modify a CCD/C32 template for candidate EHR under test. There will be CCD/C32 test templates provided inside LAIKA prior to a proctor logging in.</td>
</tr>
<tr>
<td>Step 2</td>
<td>Juror</td>
<td>CCHIT juror continuously monitors the EHR vendor administrator’s actions and has read access to all of the associated test data in LAIKA</td>
</tr>
<tr>
<td>Step 3</td>
<td>EHR Vendor</td>
<td>EHR vendor administrator uses LAIKA to access a prose document detailing all the patient data associated with the CCD/C32 test with human-readable instructions</td>
</tr>
<tr>
<td>Step 4</td>
<td>EHR Vendor</td>
<td>EHR vendor administrator follows the LAIKA test script to create a patient record in the candidate EHR under test with all of the elements detailed in the LAIKA test</td>
</tr>
<tr>
<td>Step 5</td>
<td>EHR Vendor</td>
<td>EHR vendor produces this patient’s record via a CCD/C32 XML document and uploads it into LAIKA</td>
</tr>
<tr>
<td>Step 6</td>
<td>Proctor</td>
<td>Once the vendor’s CCD/C32 is validated, and contents are inspected against test script stored in LAIKA, proctor can view the pass/fail status of the test</td>
</tr>
<tr>
<td>Step 7</td>
<td>Proctor</td>
<td>Proctor then exports an Excel Interoperability report from LAIKA and uploads it into SalesForce Test Record and Process Management Service</td>
</tr>
</tbody>
</table>

Figure 4 – LAIKA “Generate and Format” Test Interaction Sequence
3 Standards

Continuity of care is the initial focus for Laika v1.0 support of interoperability testing. The data standard to represent continuity of care data that CCHIT will require for EHR vendors will be a Continuity of Care Document (CCD) that has been constrained by the **HITSP C32 v2.1 specification** (CCD/C32).

### 3.1 Other Standards Under Post-1.0 Consideration

Today, there are numerous data, communication and transaction standards that describe different parts of an individual’s EHR. In some cases there are multiple standards covering the same section of an EHR. A partial list of just some of these standards is detailed below:

<table>
<thead>
<tr>
<th>Section of EHR</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clinical Data</td>
<td>HL7/ASTM CCD; HITSP C48</td>
</tr>
<tr>
<td>Laboratory Results</td>
<td>HITSP IS-01; HL7 CDA R2; IHE XDS; HITSP C37</td>
</tr>
<tr>
<td>Transportation</td>
<td>IHE XDS.b</td>
</tr>
<tr>
<td>Disease Registration</td>
<td>CDC Disease Registries; PHIN Disease Reporting</td>
</tr>
<tr>
<td>Pharmacy</td>
<td>NCPDP Script 8.1</td>
</tr>
<tr>
<td>Billing</td>
<td>X12</td>
</tr>
</tbody>
</table>

*Figure 5 – List of EHR Data Standards*
LAIKA v1.0 will focus on interoperability testing of CCD documents that conform to the **HITSP C32 v2.1 specification**. It is our goal to expand LAIKA support to include additional standards and more sections of an EHR and to address how these data standards are communicated over networks.

We believe that the next data standard to be included in Laika, post v1.0 will be the **HITSP C48**, which supports an “Encounter” using IHE Medical Summary.
4 Software Architecture

The LAIKA system will be separated into various software modules. The main packages provide a user interface for test script management, CCD/C32 generation, and a report generator detailing how well an EHR has complied with test criteria.

LAIKA will be written in the Java (http://java.sun.com) programming language and the Ruby on Rails web framework for all of the software modules. These modules include.

4.1 Web Interface for Testing

LAIKA will have a software package dedicated to the management of EHR data standards and data modules associated with them. With the first version of LAIKA, V1.0, we are planning to support a CCD constrained by the HITSP C32 document as our first module or a CCD/C32 document. Later versions of LAIKA will include additional standards in this module.

4.2 CCD/C32 Validation

LAIKA will have a software package dedicated to validating a CCD further constrained according to the HITSP C32 specification. Access to the LAIKA CCD/C32 validator will be supported via a function call within the Java virtual machine, and through a SOAP web service.

This particular component will be designed to identify any errors or warnings with a CCD/C32 document and express either issues in a human-readable format.

This validator module will use NIST CCD/C32 Schematron rules files that are available in the public domain. LAIKA will be enhancing with rules by providing a module that can validate coded entries via the National Library of Medicine’s (NLM) Unified Medical Language System (UMLS). This will allow the LAIKA CCD/C32 validator to both confirm syntax of CCD/C32 files, as well as the proper values of coded data inside the CCD/C32 document.

A RESTful web service will wrap this validation component. This is to provide other individuals with a CCD/C32 validator, for use outside of the LAIKA project.
4.3 Transport (future)

LAIKA will have a software package dedicated to managing transport protocols to both import and eventually export clinical documents with the LAIKA system. We initially plan on interfacing with XDS.b registries and repositories for transfer of CCD/C32 documents.

This package will use software that have already been released by the National Institute of Standards and Technology (NIST) via either an open source license or made freely available in the public domain.

4.4 Rules Engine (future)

LAIKA will have a software package dedicated to managing and synchronizing certifications rules with functional software. This package will allow leaders to specify testing rules in a format that will not require intimate knowledge of high level programming languages in order to change the detailed tests that LAIKA will enforce on communicated data.

4.5 Database

To persist the information from a LAIKA test case that will be verified from one EHR system test to another, LAIKA will rely on a relational database that supports a Java Database Connectivity (JDBC) driver. This will allow the LAIKA system to store parts of an EHR to memory or disk. LAIKA will use MySQL 5.0 for our relational database system.
5 Interface Design

All users of LAIKA (vendors, jurors, proctors, administrators) will access the LAIKA user interface over a web browser.

5.1 Web Pages

LAIKA will provide several different web page views for users to authenticate, create, edit and view LAIKA tests.

➔ 5.1.1 Login Page

When accessing LAIKA, all users will be first prompted with a login screen. This screen will request that a user provide an email address and password to authenticate with the service and start using LAIKA.

![Figure 7 – LAIKA Login Page](image-url)
5.1.2 User Account Page

If a user does not have a LAIKA account setup, anyone can

![LAIKA Account Creation Page](image)

Figure 8 –LAIKA Account Creation Page
5.1.3 Forgotten Password Page

If a user has forgotten his/her password, LAIKA will provide a page that will prompt the user for their email address associated with their account. LAIKA will send an email to that account with a link that user can access to reset their LAIKA password and authenticate with the LAIKA login screen (see Figure 7).

Figure 9 – Design LAIKA Forgotten Password Page
5.1.4 Landing/Dashboard Page

Figure 10 – LAIKA Landing/Dashboard Page
5.1.5 CCD/C32 Library Page

All tests executed in LAIKA will need to start with a CCD/C32 test template. A single CCD/C32 test template will span the CCD/C32 specification.

There will be a view to allow users to select from any of the CCD/C32 test templates in the system, called the Library Page. It is from this page that a user will be able to create new CCD/C32 templates, edit/delete existing CCD/C32 templates, and to select an existing CCD/C32 template to start a LAIKA test.

![Figure 11 – LAIKA Library Page](image-url)
5.1.6 CCD/C32 Template Page

By either creating a new CCD/C32 Temple, or clicking on an existing CCD/C32 Template, the user will be brought to a screen that accounts for all the modules and elements specified by the HITSP C32 v2.1 specification. Users can both view the data associated with this test, and edit fields of the CCD/C32 Template.

![Figure 12 – LAIKA CCD/C32 Template Page](image-url)
5.1.7 CCD/C32 File and Display Test Tracking Page

Jurors will print out this page to track the results of “File and Display” tests within the EHR systems which they access using GoToMeeting. All the data associated with a CCD/C32 test in progress will be printed out, with a checklist for jurors to record whether or not all of the fields of the test are accounted for in the EHR system.

![Figure 13 – File and Display Test Tracking Page](image-url)
5.1.8 CCD/C32 Generate and Format Results Page

If any part of the data in the CCD/C32 document does not match the test in LAIKA, associated with the EHR vendor’s CCD/C32 document, the “Generate and Format” test result page will display that the CCD/C32 document has failed the test.

Individual fields where the errors have occurred will be highlighted, showing the vendors actual/supplied value against the expected value in LAIKA.

Figure 14 – Generate and Format Test Result Page (Failed)
If all the data in the CCD/C32 document matches the test in LAIKA, associated with the EHR vendor’s CCD/C32 document, the “Generate and Format” test results page will display that the CCD/C32 document has passed testing.

**Figure 15 – Generate and Format Test Result Page (Passed)**
Lastly, there will be a page that allows anyone to view all the users in the LAIKA system.

Figure 16 – Design of a LAIKA CCD/C32 Test Result
5.2 User Interface Technologies

The LAIKA user interface will be developed for the web and will support a variety of technologies supported in newer web applications.

All the LAIKA pages will have their presentation defined in Cascading Style Sheets (CSS).

AJAX (Asynchronous JavaScript and XML) is used throughout LAIKA’s interface to support a richer user experience during the testing of EHRs. AJAX is a cross-platform technique usable on many different operating systems, computer architectures, and Web browsers as it is based on open standards such as JavaScript and XML.

Users of AJAX-enabled web applications perceive that the interface is more responsive. This is achieved by exchanging small amounts of data with the server, so that the entire web page does not have to be reloaded each time the user requests a change. This is intended to increase the web page’s interactivity, speed, functionality, and usability.

LAIKA employs the open source JavaScript toolkits script.aculo.us (http://script.aculo.us) and Prototype (http://www.prototypejs.org) to manage the Ajax-based components of the application. Additionally, SurveyEngine and WuFoo (http://wufoo.com) are used to create LAIKA form pages.

5.3 User Interface Development Description

For designers either on the LAIKA project or even interested in our work interested in making suggestions for changes or new features with the LAIKA user interface, we are requiring that all communications about the design are sent to the talk@projectlaika.org list. At a minimum we need composite images that are compatible with the Omnigraffle drawing tool. Ideally, functional HTML examples that use CCS to specify the presentation will be provided in addition to Omnigraffle-compatible designs.

The latest functional user interface code will be located in the org.projectlaika.ui package in the subversion repository.
6 Testing

The technical team on LAIKA is constantly striving to reduce the time to develop new features, and simultaneously ensure that the reliability of the software we produce is of the highest quality. To support this goal, we are requiring several testing tools and paradigms from developers on the project.

6.1 Unit Testing

LAIKA code coverage with unit testing tools will support overall code 50% branch coverage. LAIKA’s testing code coverage will be measured with Cobertura (http://cobertura.sourceforge.net).

6.2 Continuous Integration (CI) Strategy

Continuous integration is the name that has emerged in programming communities for the software engineering practice of immediately committing every change, no matter how small, to a revision control system. Other developers should always work with the latest version of the baseline of the source code.

Developers for LAIKA are required to continuously integrate their contributions to the project. There are several reasons for imposing this requirement on the LAIKA developers:

- Immediate unit testing of all changes.
- Early warning of broken/incompatible code.
- Early warning of conflicting changes.
- When unit tests fail, or a bug is discovered, developers might revert the code baseline back to a bug-free state, without wasting time debugging.
- Integration problems are detected and fixed continuously - no last minute hiatus before release dates.
- Constant availability of a "current" build for testing, demo, or release purposes.
- The immediate impact of checking in incomplete or broken code acts as an incentive to developers to learn to work more incrementally with shorter feedback cycles.

Developers for LAIKA are required to continuously integrate their contributions to the project. There are several reasons for imposing this requirement on the LAIKA developers.
At the time that this document is being written, a decision on which continuous integration system for use in LAIKA has not been made. Still under consideration are CruiseControl (http://cruisecontrol.sourceforge.net), Hudson (https://hudson.dev.java.net), and Continuum (http://maven.apache.org/continuum).

6.3 Stress and Volume Testing

LAIKA will stress tested using JMeter (http://jakarta.apache.org/jmeter). JMeter is an Java desktop application designed to load test functional behaviour and measure performance that is available under an Apache 2.0 open source license. JMeter will be used with LAIKA to report the performance under different load types of the LAIKA user interface. It will also be used to display a graphical analysis of performance or to test the server.
7 Non-Functional Requirements

7.1 Performance

It is our goal, that with the LAIKA web interface, all pages for authoring and viewing CCD/C32 test cases will respond and completely render to the calling web browser in under 1.0 seconds.

Because the validation of CCD/C32 XML documents relies on the Schematron rules specified by NIST, the LAIKA code can’t guarantee the performance when validating CCD/C32 XML files.

7.2 Languages

English is planned as the sole spoken language that the LAIKA user interface will support upon the release of version 1.0
8 Build and Deployment

8.1 Code Repository

Version control systems allow users to keep track of changes made to any type of electronic data, typically source code, web pages or design documents. All LAIKA source code will be available via the Subversion version control system from the LAIKA source forge project site.

Subversion is currently a popular alternative to CVS (another widely used version control system), and Subversion is particularly among open source projects.

We have also selected SourceForge.net (http://sourceforge.net) as the open source hosting service for LAIKA based upon an investigation into a few alternatives to hosting an open source project. Our findings comparing SourceForge, Google Code, CodePlex and Savannah are detailed in the following figure:

![Comparison Matrix for Selection of OS Code Repository](image-url)

Figure 17 – Comparison Matrix for Selection of OS Code Repository
8.2 Build Tools

LAIKA will use Maven 2.0 (http://maven.apache.org) to assist in the software project management and building process of the Java code in the CCD/C32 validator module. Maven is an open source software project management and comprehension tool that can manage a project's build, reporting and documentation from a central piece of information.

Maven uses a construct known as a Project Object Model (POM) to describe the software project being built, its dependencies on other external modules and components, and the build order. The core engine dynamically downloads plugins from a repository that strives to be the de facto distribution mechanism for Java applications.

Maven comes with pre-defined targets for performing certain well defined tasks such as compilation of code, software packaging and automatic generation of documentation.

8.3 Installation and Operation

At a minimum, the use of LAIKA will require both a server that supports the web application and a web browser.

⇒ 8.3.1 Supported Server Platforms

For the purposes of deploying LAIKA on a server to host an instance for testing, LAIKA will be tested to support Fedora Core, Windows XP Service Pack 2 and Apple's Leopard Operating System. On any of these platforms, LAIKA will require at least 2 GB of free RAM on the system, and at least 500 MB of available disk space.
8.3.2 Supported Web Browsers

LAIKA is tested on all major browsers. A table of the latest target versions is below:

<table>
<thead>
<tr>
<th>Browser Icon</th>
<th>Browser</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Internet Explorer Icon]</td>
<td>Internet Explorer</td>
<td>7.0</td>
</tr>
<tr>
<td>![FireFox Icon]</td>
<td>FireFox</td>
<td>2.0</td>
</tr>
<tr>
<td>![Safari Icon]</td>
<td>Safari</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Figure 18 – LAIKA support for web browsers

We will attempt to support Firefox 3.0 assuming the planned released of this version is made well in advance of the release of LAIKA v1.0 on March 21st, 2008.
9.0 Appendices

For the information about LAIKA, please refer to the following resources:

9.1 Web Sites

*Project LAIKA Home Page*
http://projectlaika.org/

*CCHIT Home Page*
http://www.cchit.org/

*MITRE Corporation Home Page*
http://www.mitre.org/

9.2 Articles and Publications

*Health Data Management 9/10/2007*

*Healthcare IT News 09/11/2007*
http://www.healthcareitnews.com/story.cms?id=7766

*Federal Computer Week 09/11/2007*

*MITRE-hosted Laika Press Release 11/09/2007*

*LINUX Med News 11/12/2007*
http://www.linuxmednews.com/1194895474/index_html
9.3 LAIKA Project Timeline

The primary focus for LAIKA is a production/stable release of **LAIKA v1.0 by March 21st, 2008** that supports interoperability testing of a CCD document.

Beyond LAIKA v1.0, the project will be focused on supporting its users, developing a module to integrate CCHIT rules through a rules engine, and support for a second interoperability standard in addition to the CCD/C32 standard.

A timeline for 2007 and 2008 follows:

---

Figure 19 – LAIKA 2007/2008 notional project plan
9.4 Acronyms and Terms

**AHIMA** – The **American Health Information Management Association** is a non-profit association for health information management professionals. The organization was founded in 1928, and has 51,000 members.

**AJAX** – Asynchronous **JavaScript and XML**, is a web development technique used for creating interactive web applications. The intent is to make web pages feel more responsive by exchanging small amounts of data with the server behind the scenes so that the entire web page does not have to be reloaded each time the user requests a change.

**ASF** – The **Apache Software Foundation** is a non-profit corporation to support Apache software projects and is made up of a decentralized community of developers. The ASF was formed from the Apache Group and incorporated in Delaware, USA, in June 1999.

**ASTM** – The **American Society for Testing and Materials** is an international standards organization that develops and publishes voluntary technical standards for a wide range of materials, products, systems, and services.

**C32** – A HITSP specification to constrain a Continuity of Care Document and require data elements for patient registration, medications, allergies and conditions.

**C48** – A HITSP specification to detail an Encounter using IHE Medical Summary.

**CCD** – Continuity of **Care Document**. An XML document detailing a snapshot of part of a patients electronic health record. The HL7 Continuity of Care Document is the result of a collaborative effort between the Health Level Seven and ASTM organizations to “harmonize” the data format between ASTM’s Continuity of Care Record and HL7’s Clinical Document Architecture specifications.

**CCR** – The Continuity of **Care Record** is a health record standard specification developed jointly by ASTM International, the Massachusetts Medical Society, HIMSS, the American Academy of Family Physicians, the American Academy of Pediatrics, and other health informatics vendors.

**CCHIT** – The **Certification Commission for Healthcare Information Technology**. The Certification Commission for Healthcare Information Technology is a private not-for-
profit organization that serves as the recognized US certification authority for electronic health records and their networks.

**CDA** – The HL7 Clinical Document Architecture is an XML-based markup standard intended to specify the encoding, structure and semantics of clinical documents for exchange.

**CSS** – Cascading Style Sheets is a stylesheet language used to describe the presentation of a document written in a markup language. It is most common application is to style web pages written in HTML and XHTML, but the language can be applied to any kind of XML document, including SVG and XUL.

**Derby** – Derby is a Java relational database management system that can be embedded in Java programs and used for online transaction processing. Derby is developed as an open source project under the Apache 2.0 license.

**EHR** – Electronic Health Record. An EHR refers to an individual patient's health record in digital format. Electronic health record systems co-ordinate the storage and retrieval of individual records with the aid of computers.

**FFRDC** – Federally Funded Research and Development Center.

**HHS** – The United States Department of Health and Human Services. This is a Cabinet department of the United States government with the goal of protecting the health of all Americans and providing essential human services.

**HL7** – Health Level Seven is an all-volunteer, not-for-profit organization involved in development of international healthcare standards.

**HITSP** – Healthcare Information Technology Standards Panel. The American National Standards Institute Healthcare Information Technology Standards Panel was created in 2005 as part of efforts by the Office of the National Coordinator for Health Information Technology to promote interoperability in healthcare by harmonizing health information technology standards.

**HIMSS** – Healthcare Information and Management Systems Society is a healthcare industry membership organization exclusively focused on providing leadership for the optimal use of medical informatics technology and management systems.
HIT – Healthcare Information Technology or Health Informatics. This is the intersection of information science, computer science and health care. It deals with the resources, devices and methods required to optimize the acquisition, storage, retrieval and use of information in health and biomedicine.

IHE – Integrating the Healthcare Enterprise is an international organization that focuses on the development of global standards and on the regional deployment of interoperable products.

JDBC – The Java Database Connectivity API for the Java programming language defines how a client may access a database. It provides methods for querying and updating data in a relational database.

MITRE – The MITRE Corporation is a not-for-profit organization chartered to work in the public interest. MITRE manages three Federally Funded Research and Development Centers.

NIST – The National Institute of Standards and Technology is a non-regulatory agency of the United States Department of Commerce. The institute's mission is to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve quality of life.

NLM – The National Library of Medicine, operated by the United States federal government, is the world's largest medical library. The collections of the National Library of Medicine include more than seven million books, journals, technical reports, manuscripts, microfilms, photographs, and images on medicine and related sciences, including some of the world's oldest and rarest works.

ONC – The Office of the National Coordinator for Health Information Technology provides counsel to the Secretary of the Department of Health and Human Services and departmental leadership for the development and nationwide implementation of an interoperable health information technology infrastructure.

Open Source – Open source is a set of principles and practices that promote access to the design and production of goods and knowledge. This allows users to create software content through incremental individual effort or through collaboration.
**POM** – The **Project Object Model** is the fundamental unit of work in Maven. It is an xml file that contains information about the project and configuration details used by Maven to build the project.

**RCB** – **Recognized Certification Body**. The designation granted by the United States Department of Health and Human Services to an organization that certifies HIT systems.

**Ruby** – Ruby is a reflective, dynamic, object-oriented programming language. It combines syntax inspired by Perl with Smalltalk-like object-oriented features, and also shares some features with Python, Lisp, Dylan, and CLU. Ruby is a single-pass interpreted language.

**Ruby on Rails** – Ruby on Rails is a free web application framework. It aims to increase the speed and ease with which database-driven websites can be created, and offers skeleton code frameworks (scaffolding) from the outset. Often shortened to Rails, or RoR. Ruby on Rails is an open source project written in the Ruby programming language.

**REST** – **Representational State Transfer** is a style of software architecture for distributed hypermedia systems such as the World Wide Web.

**SOAP** – SOAP refers to both **Simple Object Access Protocol**, and also lately **Service Oriented Architecture Protocol**. SOAP is a protocol for exchanging XML-based messages over computer networks.

**UMLS** – The **Unified Medical Language System** is a compendium of many controlled vocabularies in the biomedical sciences. It provides a mapping structure between these vocabularies and thus allows to translate between the various terminology systems. It may also be viewed as a comprehensive thesaurus and ontology of biomedical concepts.

**Wiki** – A type of computer software that allows users to easily create, edit and link web pages. Wikis are often used to create collaborative websites and power community websites.

**XDS** – **Cross Enterprise Document Sharing** is a profile created by Integrating the Healthcare Enterprise to facilitate the sharing of clinical documents between
institutions. XDS re-uses ebXML registry methodology to provide a centralised method of indexing documents.

**XML** – The **Extensible Markup Language** is a general-purpose markup language. It is classified as an extensible language because it allows its users to define their own tags. Its primary purpose is to facilitate the sharing of structured data across different information systems, particularly via the Internet.

### 9.5 Project Name Background

LAIKA is named after the dog and first living creature from earth to enter orbit, paving the way for human space flight. This software development effort will likewise demonstrate that the grand challenge of interoperable EHRs is attainable and will inspire others to follow.

![Figure 20 – Stock photograph of LAIKA prior to her 1957 spaceflight](image)