Appendix D
Other Noteworthy Systems

Indivo

Project Background: Indivo is an open source personally controlled health record system (PCHR) that is being developed by a joint team from the Harvard Children's Hospital Informatics Program (CHIP) and M.I.T.'s Clinical Decision Making Group. The concept and architecture of the PCHR dates back to an article co-authored by Isaac Khan, M.D., Ph.D., in 1994, and have gradually evolved since. The current architecture is described in the paper The PING Personally Controlled Medical Record System: Technical Architecture (Journal of the American Medical Informatics Association, 2004). In August 2006, PING was rebranded as Indivo. The Indivo project has been funded by grants from National Library of Medicine (NLM) and the Markle Foundation.

The source code of the application is offered under the LGPL license. The project's use of open source licensing is not specifically intended to leverage contributions from the larger open source community. The project actually uses a core set of developers for project development and rarely, if ever, accepts code contributions for outside parties (although the source code is publicly and freely available for download). Rather, the primary reason the project is open source, according to the project leaders, is to engender trust in the software among the health community by allowing anyone to view and comment on the product's security and architecture.

Product Description: Indivo functions analogously to personal accounting software that downloads transaction data from users' bank accounts. A patient with an Indivo account may subscribe to data feeds from hospitals and other health information networks that have partnered with Indivo. These partnering organizations keep the patient's Indivo record up to date with medical encounter data, including clinical notes, immunizations, medications, lab results, and so forth. A patient may also manually enter supplemental personal health information that is not included in the data feeds. The patient can grant access to her Indivo account to other Indivo account holders, such as family members, physicians, and clinical research programs. The core security approach of the application is that a patient's data will only be used in ways to which the patient has explicitly consented.

One notable feature of the Indivo product is the data storage model of the Indivo server. All patient data is stored within a set of discrete XML files, comprising the patient's lifetime personal medical history. Each file is individually encrypted on the server and in backup files. The server also maintains a master XML document for each patient, which indexes the different types of clinical documents stored for that individual.

To support this data-storage approach, Indivo has developed a set of standardized document formats, or XML schemas, for representing clinical data within the PCHR. These data schemas accommodate a wide variety of clinical and administrative concepts including, but not limited to, allergies, appointments, contacts, discharge summaries, immunizations, lab tests, and medications. Indivo asserts that its document standards offer a much greater level of detail and granularity than existing standards, such as the CCR or HL7's CDA. At the same time, the Indivo server is agnostic with respect to the actual structure of the XML documents it stores. Any type of data, including binary data, may be stored

by the Indivo server as long as it is wrapped inside the application's basic document XML schema. This generic document structure contains, at a minimum, a header element that describes the contents of the document. In this way, the Indivo server may store any generic data (including CCR and CDA formatted records) alongside clinical data formatted in Indivo's own strongly-typed and coded document formats. The Indivo approach theoretically allows the clinical document model to evolve over time because the architecture of the server is not tied to the underlying data model.

The challenge of this model is that any client application connected to an Indivo server must be prepared to handle any XML documents in a generic, yet user-friendly, way. Naturally, patient data are most useful when transmitted in one of the strongly-typed document formats defined by Indivo. However, the ability to capture patient data in any format has greater value than limiting the accepted data to expected formats. This loose typing does not pose a significant problem for the Indivo model because the Indivo server is not designed as a platform for data-mining or decision support. Indivo is exclusively a personal patient data repository (although a patient may choose to release data to a trusted third-party for the purposes of analysis). In this case, the health care provider downloads the XML patient data and may transmit it to another system that enables reporting and decision support.

The Indivo server uses a J2EE Java Application Server architecture that manages the security and access control for the aggregated health information. Client applications communicate with the server using XML sent over HTTP (although the server does not use a formal SOAP-based web service protocol). Indivo has developed an API for client applications and third-party interface developers to help convert data from external systems into the Indivo document standard formats required for submission to the Indivo server.

There are, at present, two supported client applications developed by the Indivo project team that connect to the Indivo server. The first client is a PHP-based web application that allows Indivo account holders to edit and view their health data, subscribe to new data feeds, and grant others access to their personal data. It provides displays for all of the expected health-related information including clinical notes, lab tests, medications, and so forth. This web client is now being deployed as part of a pilot project to 4,000 students at MIT and will be rolled out at Harvard Children's Hospital starting in October, 2007. The other prominent client application is the Indivo Subscription Agent, which is used by partnering health care organizations to upload data to the Indivo server. This client handles provider authentication, patient subscription management, and document versioning. Transmission of data using the subscription agent is unidirectional, from the health care organization's database to the Indivo server. The details of how an organization submitting data verifies that an Indivo account holder has access privileges to those data are determined separately by each organization.

The Indivo product is still in development and implementations are in their early stages. The implementation at MIT and the planned roll out at Harvard Children's Hospital will be the first Indivo systems to go live. A representative of the Indivo project indicated that there are plans for additional implementations, but was unable to discuss any specifics of these implementations. Indivo has indicated that its ultimate goal is to have the system implemented and hosted at the county, state, and federal levels, with data provided by major regional health care providers, such as large hospitals and retail walk-in clinics. As development of the server architecture is completed, Indivo plans to begin focusing its development efforts on creating additional tools to make interfacing with the Indivo server as simple as possible for health care organizations.

Analysis: The creators of Indivo have developed an interesting, flexible, patient-centric approach to the problem of managing the access to and aggregation of life-long personal health information. To support their approach, they have created their own set of highly structured clinical document formats, while at the same time allowing the use of competing document standards or ad hoc document formats.

The Indivo solution is focused on collecting patient information from large hospital networks and RHIOs. There is currently no role for small practice physicians and community clinics to contribute data to the Indivo server, although physicians may be granted read access to their patients' Indivo records. If adoption of the Indivo solution becomes sufficiently widespread, mechanisms may evolve that allow patient data stored in an Indivo database to be merged with physicians' EMR systems. In the meantime, success of the approach will depend, in part, on whether large health care organizations are willing to contribute patient-specific data to Indivo repositories.

Medsphere

Project Background: Medsphere is a for-profit health care information technology company founded in 2002 that offers commercial open source products and services. The company's primary product, an EHR system called OpenVista, is based on code acquired through the Freedom of Information Act (FOIA) access to the VistA system developed by the U.S. Department of Veterans Affairs (VA). OpenVista reportedly addresses barriers to VistA implementation outside the VA, including lack of consistent commercial billing support, a VA requirement product bias, suboptimal user interfaces for ancillary applications (pharmacy and lab), and insufficient support for women's and children's health.

Medsphere is currently focusing its sales efforts on 100-plus bed acute care hospitals, as well as multi-facility hospital groups and government health organizations. According to Medsphere, the company has performed roughly 100 installations of VistA-based technology to date at a mix of customer sites in the private and government sectors. Medsphere is not targeting the small physician practice market at this time, although the system can be used in satellite offices or offsite practices that are owned and operated by larger hospitals or integrated delivery networks that have implemented OpenVista.

Until recently, Medsphere was led by Kenneth W. Kizer, M.D., M.P.H., the former head of the VistA CPRS project at the VA. Dr. Kizer has stepped down from the CEO position, and Medsphere is currently recruiting a qualified replacement. Medsphere emphasizes that the remaining product management team has a strong combination of commercial, open source, and VA experience, and that the company has an experienced staff of former VA developers. Medsphere also reports that it has substantial funding (the company raised $7.5M in series B funding in 2005).

Product Description: Medsphere offers two versions of its OpenVista solution, differing primarily in terms of available features. The first, a commercial EHR, consists of publicly available source code plus proprietary enhancements contributed by Medsphere. The company does not charge a license fee for this product, but requires an exclusive pay-as-you-go support contract for implementation and support.

Medsphere's OpenVista Community Edition (CE) is a pure open source version of the system that was released to SourceForge in February, 2007. According to Medsphere, OpenVista CE contains about 95

percent of the functionality offered in the commercial version of OpenVista. Medsphere regularly updates the CE version to add features from the commercial version, although the functionality of the CE version will lag a year or two behind the commercial version at any time. Medsphere does not provide service or support for OpenVista CE and predicts that third-party firms will soon begin to offer support for this version of the product, although none do so today.

The commercial and CE versions of OpenVista are architecturally identical and comprise two primary components: the OpenVista server and a suite of client applications centered around the Clinical Information System (CIS) electronic medical record (EMR). The OpenVista server is a fully open source implementation of the original FOIA VistA server stack, modified so that all layers of the server architecture can be open source. For example, an open source product called GT.M serves as an alternative to the commercial Caché for the MUMPS application server layer. OpenVista CIS is a graphical cross-platform client application that interfaces with VistA-based server solutions and provides a number of functional improvements over the original VistA Windows-only display. The OpenVista Bridge, a middleware component, handles communication between client and server and enables client applications to connect with different server implementations (currently, either the standard FOIA VistA server or Medsphere's OpenVista server).

Medsphere has added features to their OpenVista product to allow it to work better in private sector hospitals. Some of the biggest changes include: (1) the use of a general patient identifier in place of a social security number, (2) support for third-party billing, (3) a new client graphical user interface (GUI) with enhanced workflow and security features, (4) and advanced nursing modules. These enhancements are available in both the commercial and community versions of the product.

The OpenVista CE server is licensed under the standard general public license (GPL), while the client and bridge are licensed under a custom Medsphere Public License (MSPL), which is based on the Mozilla Public license. This licensing arrangement offers the ability to modify, recompile, and sell derivative works based on the original code, but requires that all derivative works include attribution in the form of a label in the user interface that reads Powered by Medsphere. In addition, Medsphere requires a Contributor License Agreement that obligates all parties who make enhancements to the application to provide the source code for those enhancements back to Medsphere (which may or may not make the enhancements publicly available).

Assessment: OpenVista is a strong offering in the open source hospital information system market. Medsphere appears to have the funding and resources necessary to provide enhancements to the VistA code-base that allow the product to work better in the private sector. Furthermore, the company has the project management structure and experience to provide effective implementation support for hospitals interested in OpenVista. The company's lack of direct support for small ambulatory practices is a notable gap, but Medsphere is currently focused on improving VistA's core competencies for the inpatient setting. Medsphere is in the process of working out its approach to the open source model as it brings on a new CEO, and changes to its current approach may be forthcoming.

Tolven Healthcare Innovations

Project Background: Tolven Healthcare Innovations is a commercial open source company that offers a collection of electronic health record applications for both patients and clinicians to enable more effective sharing of health information. Tolven was started in February, 2006 in response to its founders' belief that the health care market had too many siloed solutions that didn't address the secure sharing of patient data among providers and patients, were not built with a strong foundation of clinical data standards, and didn't provide adequate interoperability. The solutions offered by Tolven are intended to address these inadequacies in an open source and transparent manner. The company was started with private funds, and it reports that it is already recognizing revenue from implementation and support activities.

The source code for all of the Tolven products is offered under the LGPL license. Tolven chose the less restrictive LGPL license to allow users of the source code to leverage it in an open or proprietary manner. Tolven seeks input from the Health IT community on all aspects of its development process. This input includes an open review of all source code, as well as contributions of bug fixes and enhancements. However, Tolven also extends this open collaborative model to the development of their clinical data definitions (described below). Tolven's open model is meant to tap the clinical expertise of the participating health care and health information technology communities, which Tolven leverages to help implement logic for the Tolven rules engine (also described below).

Product Description: There are three related products offered by Tolven: The Tolven Information Platform, the Electronic Personal Health Record (ePHR), and the Electronic Clinician Health Record (eCHR). The Information Platform serves as the core component for both the ePHR and eCHR and is intended to serve as a basic data services platform for the development of other EMRs and health information technology systems. The concept of the Information Platform is that applications can be built on top of a common architecture and will therefore be able to interoperate more easily than the largely siloed systems available today.

The Information Platform uses a Java-based server architecture. This server may run on any operating system, application server, and database, although most implementations use either the Windows or Linux operating system, the JBoss application server, and the a PostgreSQL relational database, and the JBoss Rules Engine, all of which are open source compatible. The rules engine is the core feature of the platform. The platform is designed to accept any document in any format as an input. The server stores the document in its original format, and the rules engine extracts and processes data from the document based on configurable rule sets. The platform has native support for processing ASTM's Continuity of Care Record (CCR) documents and HL7 v2/v3 messages. It can accept any generic document format including DICOM, PDF, JPG, and so forth. The Information Platform is agnostic with respect to the format of documents that it can handle; the indexing and processing of documents are controlled by configurable rules in the rules engine, which may be designed to handle arbitrary document formats. A simple example of a processing rule would be: If the system receives a CCR document that indicates that a patient is a smoker, record that fact in the patient's existing record, check to see whether the

patient has any complicating factors (such as heart disease or a family history of cancer), and, if so, add the patient to her doctor's 'at-risk smoker' list for follow-up.

The rules that can be written are highly dependent on the degree of formal structure and coding within the received documents. The server is pre-loaded with the Unified Medical Language System (UMLS) metathesaurus to assist in the mapping of codes. When possible, the rules engine converts the specific codes it receives (e.g., LOINC, SNOMED, RxNorm) to higher level UMLS concept identifiers (CUIs). The rules engine can then apply decision logic against this common coding system. To assist the decision logic, the server makes use of clinical data definitions. These definitions are collections of template-based medical concepts that are curated by the open source community via a wiki interface (www.wikiHIT.org). The templates contain a host of clinical content, including human readable definitions of medical concepts, links to concept identifiers from the UMLS, relationships to other concepts, applicable data types, potential responses to questions, and a medical category hierarchy (e.g., chest pain can be defined as a sign or symptom observation and a member of the cardiovascular domain).

These clinical data definitions formally describe medical concepts used by the Tolven rules engine, instances of which are referred to as archetypes-semantically interoperable objects. According to Tolven, as data are extracted from a document, the Tolven system converts the data into these formal archetype entities objects so that rules logic can be applied in a consistent way on the normalized data. For example, the Tolven server may receive a lab result via an HL7 message and use standard LOINC codes in the message to determine that the message contains an Hgb A1c result. The rules engine determines that an archetype exists for this result type, and it extracts the attributes listed in the archetype's clinical data definition from the message. The rules engine can then perform additional rules logic on this normalized representation of the Hgb A1c result data.

The ePHR and eCHR are applications that Tolven has built on top of its Information Platform. Both applications are Java-web-based web applications solutions that are complementary to one anther. The ePHR system provides the typical features of a personal health record, including management of personal demographics, medications, allergies, immunizations, and so forth. Access to the record data can be granted to family members, personal physicians, and clinical research studies in which the patient might participate. Where possible, the data entered by the patient are associated with standardized codes. For example, when a patient fills out a standardized questionnaire, responses are coded to the degree possible. The data in the ePHR is not limited to those entered manually by the patient, but may also include information entered by a health care provider or medical group with whom a patient has shared her record.

The eCHR is the clinician's tool for viewing and adding to a patient record stored in the Information Platform. The eCHR is similar to a traditional EMR application in that it enables health care providers who have been given access to a patient's record to review and update the patient's labs, active medications, active problems lists, and so forth. A notable feature of this application is that it allows the physician to build lists of patient populations based on a variety of criteria. For example, patients can be grouped by primary diagnosis, current medication, comorbidity, and so forth. This is possible because of the underlying information platform's use of the rules engine, coded data, and clinical data definitions. These features give the Tolven eCHR the potential to be a disease management and clinical research tool in addition to a clinical record system. Tolven is planning to add more EHR features, such as the ability

to generate orders and to create SOAP-style progress notes. Tolven does not intend to add any practice-management or billing functionality to the product, preferring instead to interoperate with existing products.

Tolven plans to generate revenue by offering end-to-end implementation and support services for its products, including user support (second and third tier), enhancements and bug fixes, solution design and development, and training. Development of rules for Tolven's rules engine will no doubt be a large part of its business. Tolven will also offer consulting and assistance to those third parties wanting to leverage the Information Platform as the foundation of custom applications. The firm's largest engagement to date is with the UCSF Cancer Center of Excellence, where Tolven is developing rules logic and patient data-collection forms that will enable the center to more easily track and report on patients enrolled in breast cancer studies.

Tolven believes its products are appropriate for a very wide range of organizations. They envision Tolven being used by entities as large as states and even countries to give their populations a way to securely store personal health information for use in the event of an emergency (such as Hurricane Katrina). Tolven also believe its system will work for entities as small as a medical groups that wish to improve their communications with patients and relieve patients from constantly supplying the same personal health information.

Assessment: Tolven is a very generic solution general platform on which a variety of health information technology solutions may be built. The product's focus on standards and rules-based processing builds a strong informatics-based foundation for the development of clinical applications. The flexibility of the product also necessitates a significant amount of effort to implement solutions for the company's initial customers. Decision logic must be customized and interfaces must be established with existing systems for each new customer. The company is young (though the principals have a long collective history of experience in the health information technology domain), and it currently has only a few implementations underway. Still, Tolven's solution is noteworthy as an open-source platform for the collaborative design and cooperative development of clinical information technologies.

Ultimate EMR

Project Background: Ultimate EMR began in 1999 as a proprietary EMR system that was implemented with the support of a small number of physician practice sites in south Florida. Judging that the initial solution was successful, the project stakeholders began to consider commercializing the product. After deciding that they did not want to compete with other established vendors, they settled on developing the system as a commercial open source project. Ultimate EMR partnered with a Brazilian firm that had experience in open source development and began porting the application to an open architecture (the original application's dependence on proprietary libraries necessitated this port). Development of the open-source implementation began in 2003, and the first version was posted to Source Forge under a GPL license in 2007.

Project Description: Ultimate EMR has a very different architecture than the other open-source projects discussed in this report. The system is built on the Zope content management framework using the Plone content management system. Zope (Z object publishing environment) is an open source web application

server written in the Python programming language (equivalent to the Apache tier of the LAMP stack). Plone is a content management web application that runs on Zope. Plone provides content management, user management, site navigation, content searching, and page formatting functionality. Ultimate EMR uses the features of Plone to build views of patient data. Ultimate EMR uses the Zope object oriented database called ZODB for data persistence.

Ultimate EMR has been designed explicitly for the very small practice. Development has focused on ease of use and simplified data entry, rather than on providing a complex and feature rich EMR solution. The product offers the scheduling and registration features of a typical practice management system, but does not include a billing capability (relying instead on integration with existing billing systems). For EMR functionality, the product currently offers features problem lists, prescription printing, SOAP notes (free-text only), and a document manager. A notable feature of the system is a transcription desktop view into the record system, which allows a transcriptionist to enter transcribed dictations into the appropriate patient record and post the documents for electronic review and sign-off by the physician.

The application lacks a reporting module, coded medication lists, and an HL7 interface. Laboratory test information may be stored as scanned images in the document manager, with no provisions currently for storing or processing structured lab results. All data entry in the system is currently free-text only, and there is no support in the current version for ICD-9 or CPT4 coding or any other standardized medical vocabularies.

Ultimate EMR is available as either a hosted service or as an onsite installation, and the company offers a service contract for either type of implementation. Ultimate EMR prefers to focus on the hosted model and is strongly promoting this approach.

Ultimate EMR does not intend to offer direct implementation support for the application. Instead, the firm working to establish a distribution channel whereby its partners will provide installation services to interested practice sites. These third-party technicians will be licensed by Ultimate EMR to perform basic implementation (including set-up of hardware and software and basic system configuration) for those practice sites interested in the on-site solution. However, all content customization, data integration, and feature enhancements for the hosted software will be managed and performed by the Ultimate EMR Company as a fee-based service. The company is still in the beginning stages of its business planning and has not yet engaged any third-party partners. Ultimate EMR is currently implemented at two small practice sites in Florida -- a cardiology practice with two physicians and an internal medicine practice with three physicians.

Assessment: Ultimate EMR is appropriate for the small physician practice with very basic EMR needs. The system focuses on ease of use and intuitive user interfaces at the expense of robust and automated data-management features. The lack of coded data or basic reporting features, in particular, will prove to be a significant barrier for any practice that wants to do more than schedule patient appointments, generate electronic patient notes, and review problems lists, medication lists, and scanned documents. In particular, the system cannot, at present, support most of the decision-support or reporting functions required for clinical quality improvement. It is possible that, as the number of practice site implementations grows, Ultimate EMR will begin to address these existing gaps in functionality.