TRANSFoRm

Translational Research and Patient Safety in Europe

The TRANSFoRm Project is partially funded by the European Commission under the 7th Framework Programme

Grant Agreement Number FP7-247787

D3.5
Software Integration Plan

Work Package Number: WP3
Work Package Title: Technical frameworks, policies and integration

Nature of Deliverable: Report
Dissemination Level: Public
Version: 1.0
Delivery Date (Annex 1): M36

Principal Authors: Vasa Curcin
Contributing Authors: Derek Corrigan, Cedric Vansuyt, James Rossiter, Lei Zhao, Theodoros Arvanitis, Adel Taweel, Eamonn O'Toole, Piotr Brodka
Partner Institutions: Imperial College London, Custodix, University of Birmingham, Royal College of Surgeons in Ireland, Trinity College Dublin, Wroclaw University of Technology, King's College London

7th Framework Programme http://cordis.europa.eu/fp7/ict/

European Commission http://ec.europa.eu/information_society/

eHealth
Better Healthcare for Europe

European Commission

Table of Contents

List of Figures

List of Tables

Executive Summary

TRANSFoRm is a digital infrastructure that shall support translational research across a number of domains, including epidemiological studies, randomized control trials and diagnostic support. Its software part is delivered as a component-based architecture, organized around three software configurations matching the three domains we are investigating.

This document provides an overview of the full software landscape of TRANSFoRm, focusing on the interactions between the components and what are the necessary integrations required to realize the three configurations. Relevant development practices are described, both on the team and project level, together with the standard integration process.

Due to the modular nature of the software, integrations between components are pair-wise, with minimal ordering dependencies present. Nonetheless, a detailed schedule of integrations has been developed covering years 3 and 4 of the project, and key critical dependencies have been singled out. These are given with the descriptions of each pair-wise integration. Finally, we show an overview of the current state of the integrations, and list the integration meetings held in Year 3.

The timelines in this document are discussed tri-monthly at each Face-to-Face meeting and potentially modified if deemed necessary.

1. Introduction

This document describes the plan and method for integrating software components produced in the TRANSFoRm project and ensuring their interoperability to deliver a software infrastructure capable of supporting the two use cases in the project, and related technologies that are not directly addressed in the use cases.

TRANSFoRm will produce a digital infrastructure to support translational research by facilitating the reuse of routinely collected primary care data in different types of clinical research and providing mechanisms for reusing data collected in trials in clinical data repositories. Associated with this is the decision support component that uses the same infrastructure to provide diagnostic recommendations at point of care based on the knowledge extracted from routinely collected data. This is shown in Figure 1.

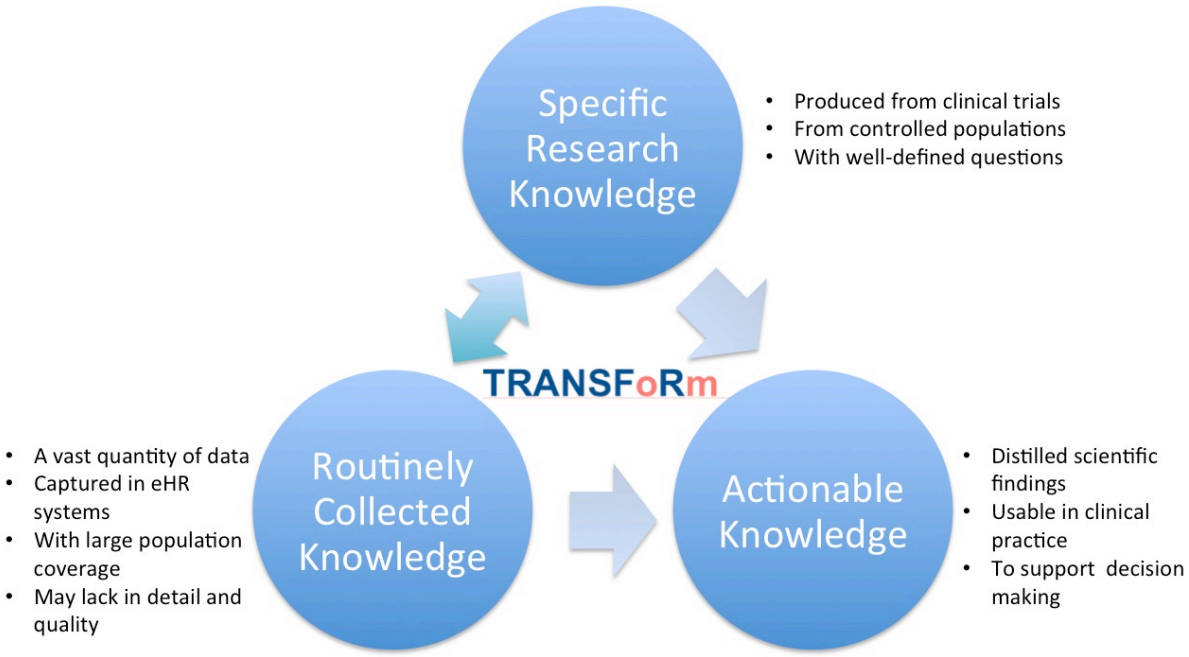

Figure 1: Data in TRANSFoRm

The components of the TRANSFoRm digital infrastructure consist of software tools and underlying models, which are easily combined in software stacks, akin to the LAMP model[1]. In such way, the users of TRANSFoRm technology will be able to pick and choose the components they require, possibly from multiple providers, safe in the knowledge that the resulting composite system (aka stack) will seamlessly work together. The two use cases [D1.1] demonstrate two possible approaches to the use of the infrastructure. The diabetes use case requires the TRANSFoRm technology to retrieve genotypic and phenotypic data from multiple data sources for an epidemiological study. The study is in silico, with no information obtained directly from the patients. The Gastroesophagal Reflux Disease use case uses the same technology to conduct a real clinical trial: recruit patients for the trial, based on their electronic health records, issue the electronic Case Report Forms and Patient Reported Outcome Measures, and collect the results that are then, together with some of the care data, fed back to the researcher.

With this vision in mind, it became evident that the software integration required in the project is based on a complex network of interacting software components and models, developed both sequentially and in parallel. This document, that is regularly reviewed at each three-monthly Face-to-face project meeting, presents the system components and development environments in section 2, the integration plan and process in section 3, and concludes in section 4 with the review of integration activities completed in year 3 (M24-M36).

[1] Originally standing for Linux-Apache-MySQL-PHP, term now denotes a family of interoperable open-source technologies.

2. TRANSFoRm software landscape

The TRANSFoRm software ecosystem is produced by multiple partners, and consists of executable software components and non-executable models that the software tools are based on. The high-level overview of the software components is shown in Figure 2.

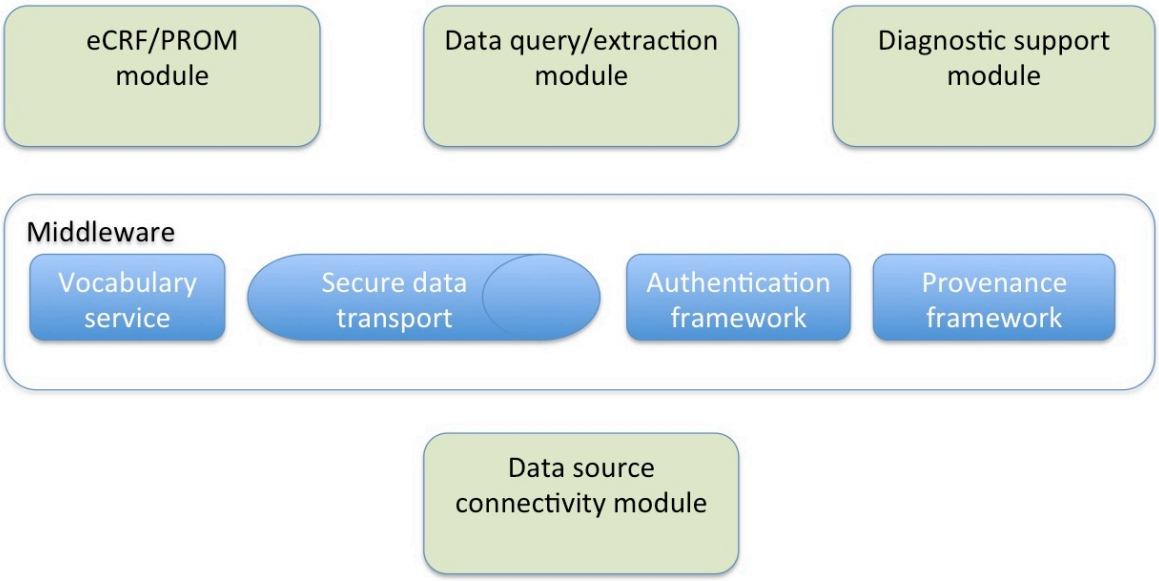

Figure 2: High-level software components

Three basic configurations of the tools will be evaluated in the project. Epidemiological Study configuration (Figure 3) is used in the Diabetes use case and consists of tools and frameworks for secure, provenance-enabled design and execution of epidemiological studies, from query design to phenotypic and genotypic data retrieval from heterogeneous data sources. Queries are formulated using standardized elements, and using information about suitable practices obtained from the Data quality tool and sent to the data sources, using a secure data transport mechanism that communicates with TRANSFoRm connector tools on the data source side.

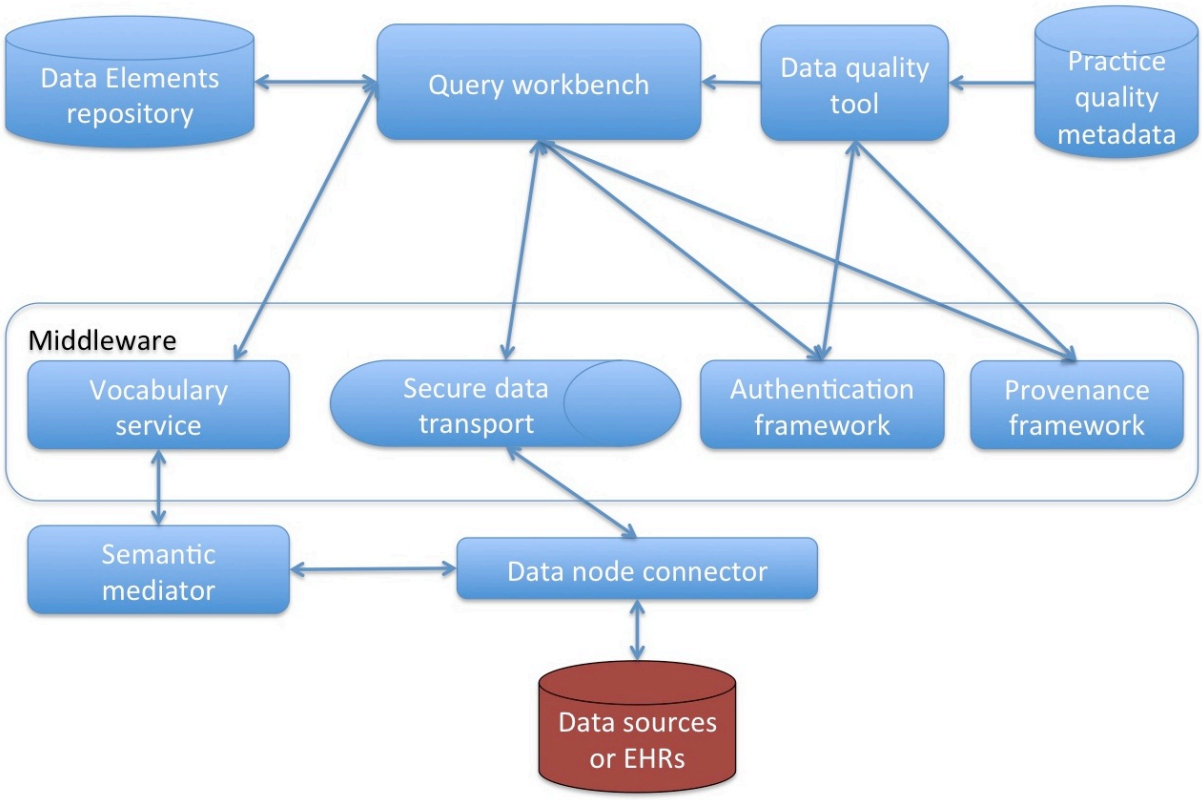

Figure 3: Epidemiological Study configuration

Clinical Trial software configuration (Figure 4) is used in the GORD use case, and consists of components needed for design, deployment, and collection of trial data, backed by provenance and secure authentication framework. Trial data consists of Patient-Reported Outcome Measures (PROMs) and electronic Case Report Forms (eCRFs) that are collected using the web and mobile devices as well as from the EHR systems in the clinical institutions where the trials are taking place.

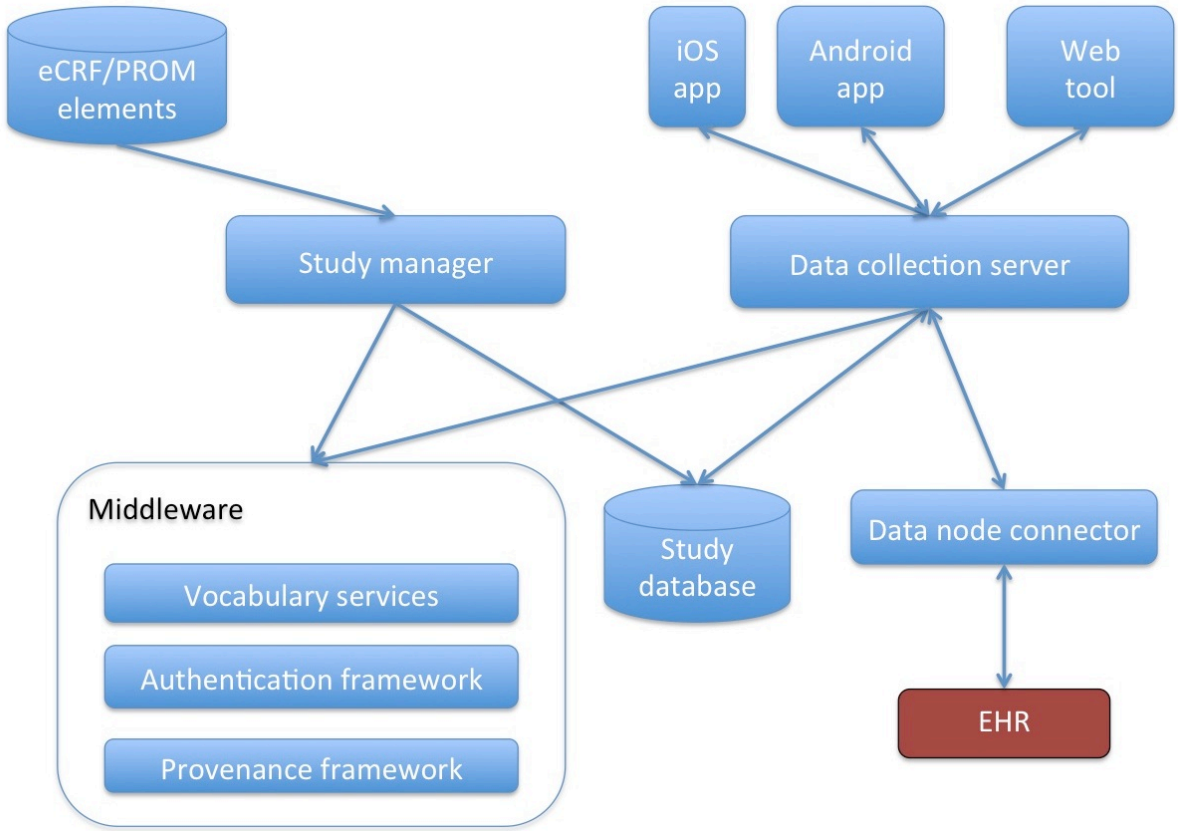

Figure 4: Clinical Trial configuration

Diagnostic Support configuration (Figure 5) is evaluated in collaboration with industrial partners. It consists of tools for mining rules from health data sources, and managing their deployment into the knowledge base upon which an evidence service is operating to support clinician diagnostic support tool embedded into a local EHR system.

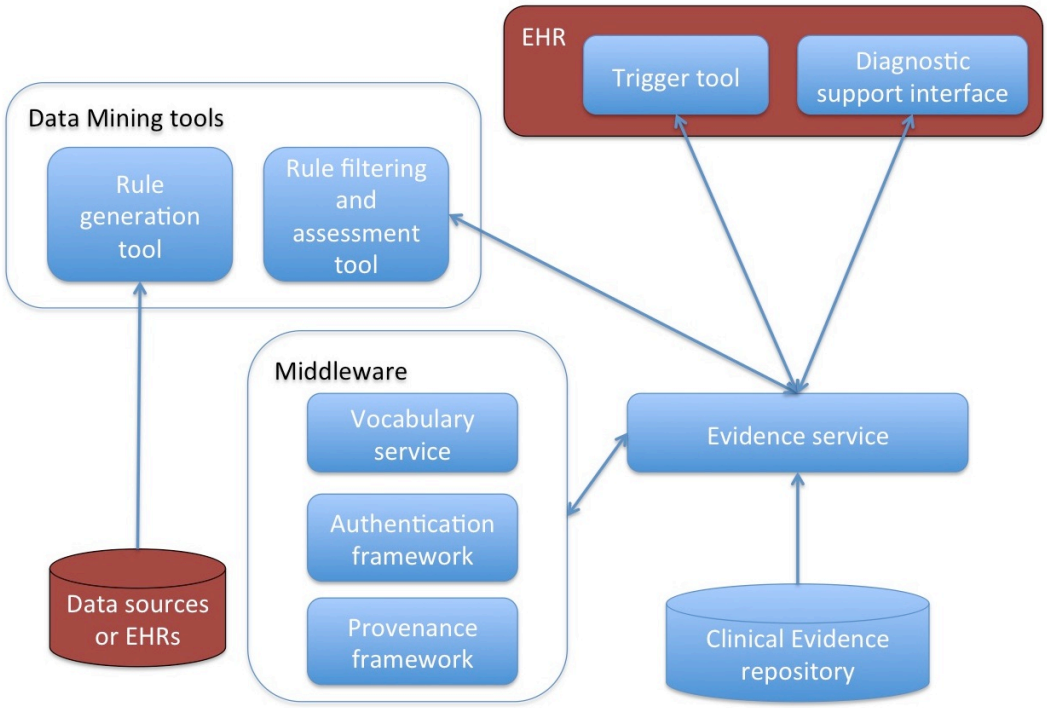

Figure 5: Diagnostic support configuration

In this section provided is an overview of all the software tools, together with the details of their development, and integrations required with other tools.

2.1 Software tools

Table 1: List of software tools

ID	Tool name	WT
1	*Authentication framework*	*3.3*
2	*Clinical evidence repository*	*4.3*
3	*Diagnostic evidence service*	*4.4*
4	*Rule data mining tools*	*4.5*
5	*Data quality tool*	*5.1*
6	*Diagnostic support interface for data collection*	*5.2b*
7	*Query workbench*	*5.3*
8	*Provenance framework*	*5.4*

9	*Semantic mediator*	*7.1*
10	*Vocabulary service*	*7.2*
11	*Data collection metadata repository*	*7.3*
12	*Infrastructure for study definition and deployment*	*7.4*
13	*Secure data transport*	*7.5*
14	*PROM and eCRF data collection*	*5.5, 7.6*
15	*Data source connector*	*7.5*

This section details the software tools developed as part of TRANSFoRm. For each, the development team is given, together with worktask(s) referenced in the Description of Work, timelines with any deviations, and list of other tools and models it integrates with. In case of tools with multiple versions, details of each are presented.

1. Authentication framework

<table>
<tr><td>Developed by: TCD</td><td>WT: 3.3</td><td>Timeline: M1-M48</td></tr>
<tr><td colspan="2">Integrates with:
Data quality tool
Query workbench
Provenance framework
Evidence service
Infrastructure for study definition and deployment</td><td rowspan="2">Versions:
v1.0 (M18) Initial version, security library.
v2.0 (M36) Single sign-on, user creation mechanism.
v3.0 (M48) Global role management.</td></tr>
<tr><td colspan="2">Models used:
N/A</td></tr>
</table>

Dynamic extensible authentication service for TRANSFoRm distributed middleware, distributed services/agents and software and user services. Based on Shibboleth, and described in D3.3: Security Solution Layer, this service provides authentication and authorization for all user-facing TRANSFoRm software.

2. Clinical evidence repository

Developed by: RCSI, TCD	WT: 4.3	Timeline: M6-M30
Integrates with: Diagnostic evidence service		Versions: v1.0 (M30)
Models used: CEM		

This component is responsible for providing a repository of clinical evidence, shown in Figure 5. The evidence is represented in an ontology-based computable format that supports defined clinical scenarios to be used to underpin the provision of a diagnostic decision support service based on evidence based clinical knowledge. Ontology is represented using web ontology language and resource description framework (OWL/RDF).

The main application programming language is Java, and Spring and Hibernate frameworks will be used. For the database MySQL[2] is being used. Protégé[3] is used for developing the ontology of clinical evidence.

The hosting of ontology is being done using Sesame triple store (using TomCat and MySQL).

3. Diagnostic evidence service

Developed by: RCSI, TCD	WT: 4.4	Timeline: M24-M42
Integrates with: Clinical evidence repository Rule data mining tools Diagnostic support interface for data collection		Versions: v1.0 (M36) v2.0 (M42)

[2] http://www.mysql.com/

[3] http://protege.stanford.edu/

Authentication framework Provenance framework	
Models used: CEM, PROV-CEM	

Diagnostic evidence service is a Web-based tool to support clinical decision making in primary care, on the basis of Clinical evidence repository. The system searches for relevant Clinical Prediction Rules, triggers their application, and recommends decision and/or safety support in an integrated clinical pathway.

The interfaces to this component will be provided by means of web services. There will be two main web service interfaces: a query interface to get clinical recommendations from the web service (used by the Diagnostic support interface) and an update interface that will be used to update the underlying clinical knowledge from research data (through Rule mining tools). It also uses the Authentication and Provenance frameworks. The interfaces will be implemented using standard web services technologies (such as Java Web services or REST based services).

v1.0 consists of a populated clinical evidence ontology, a technical hosting infrastructure and implementations of initial prototype methods to demonstrate access to query and update the ontology through web service based interface.

v2.0 comprises the complete set of web methods to support the defined diagnostic strategy and update of evidence from data mining tools.

4. Rule data mining tools

Developed by: IC, RCSI, WRUT	WT: 4.5	Timeline: M24-M36
Integrates with: Diagnostic evidence service		Versions: v1.0 (M36)
Models used: N/A		

This component consists of collaborative analytical tools for validation and improvement of generated rules and managing their deployment into the rule repository. These include visual model evaluation, together with drill-down methods for investigating the properties of the evidence sets used to construct rules. This functionality expands the ability of the system to collect and analyse the data on symptoms and signs in association with the reason for encounter and diagnostic outcome data to validate existing, and develop new clinical prediction rules (or likelihood ratios) for diagnosis by means of advanced machine learning methods like ensemble, hybrid and collective classification as well as structure prediction models.

The component uses Konstanz Information Miner (KNIME)[4], an open-source workflow-based data mining environment, and WEKA data mining algorithms, to define the analytical workflows.

5. Data quality tool

Developed by: NIVEL	WT: 5.1	Timeline: M18-M30
Integrates with: Authentication framework Query workbench Provenance framework		Versions: v1.0 (M24) v2.0 (M30)
Models used: PROV-CRIM		

The Data quality tool measures data quality of different care data repositories and other databases. Dependent on the type of research question, the researcher is able to decide what databases are suitable to participate in a clinical trial or a cohort study. The data quality tool runs on a periodically (monthly) updated database of the research metadata extracted from the practices, so that it can provide approximate figures needed for the researcher to make this decision.

[4] http://www.knime.org/

SQL Server database is used for calculating the quality measures developed for each use-case, while the web tool is using a separate MySQL database to store all the retrieved values for the quality indicators. These values are updated by the database owners, using Excel or CSV (Comma Separated Value) file formats. User queries are saved as XML and/or CSV files.

Data Quality Tool is implemented as a web application, with the lists of practices output from the tool into the Query workbench, and its actions captured through the Provenance framework. v1.0 constitutes an initial prototype for internal partners' feedback, with v2.0 offering full functionality.

6. Diagnostic support interface for data collection

Developed by: UB	WT: 5.2b	Timeline: M36-M48
Integrates with: Diagnostic evidence service *EHR systems*		Versions: v1.0 (M42) v2.0 (M48)
Models used: CEM		

This component provides the interfaces needed for collecting data from clinicians to provide diagnostic support together with a triggering component to flag patients with conditions present in the evidence service. Based on the information required by Diagnostic evidence service, it will dynamically construct data collection pages and send the data to the decision service to obtain a suggestion. The data collection pages are launched after the Trigger tool, based on LEPIS technology, discovers from the EHR system that the attending patient satisfies DSS criteria and alerts the clinician. V1.0 provides initial collection of pages for clinician verification and v2.0 fully implements all features.

7. Query workbench

Developed by: UB	WT: 5.3	Timeline: M12-M36
Integrates with: Authentication framework		Versions: v1.0 (M24)

Data quality tool Provenance framework Vocabulary service Secure data transport	v2.0 (M36)
Models used: CRIM, CDIM, PROV-CRIM	

Query workbench is used to create, manage, store and deploy queries of clinical data to identify subjects for clinical studies, evaluate trial feasibility, and to analyse the numbers of matching subjects in cohort studies. It is based on the CRIM research data model and uses the CDIM model for constructing queries, together with the Vocabulary service for coding query concepts in supported terminologies. The Authentication framework handles different user roles in the process, while the Secure data transport is used for sending queries to the data sources and receiving replies from them. Provenance of steps in the process, taken from CRIM, is captured in the Provenance framework using PROV-CRIM model. V1.0 provides the query generation capabilities, while v2.0 includes the response retrieval and data extraction.

8. Provenance service

Developed by: IC	WT: 5.4	Timeline: M12-M24
Integrates with: Authentication framework Data quality tool Query workbench eCRF/PROM data collection Diagnostic evidence service		Versions: v1.0 (M24)
Models used: PROV-CRIM, PROV-CEM		

The provenance framework manages the collection of and access to provenance data created during the operation of TRANSFoRm tools. The data store is using a MySQL database to store the information submitted by tools through a WSDL API. The data is represented using the Open Provenance Model (OPM) together with the

bridging provenance models that link OPM to TRANSFoRm's CRIM and DSM models.

Provenance service is integrated with the Query workbench, Infrastructure for study definition and deployment, Diagnostic evidence service, and the Data quality tool, all of which store provenance records of their tasks. The links are provided from the provenance login information into the Authentication framework to obtain user details related to particular actions. The integration with front-facing tools is provided through the API based on the bridging provenance models PROV-CRIM and PROV-CEM that are derived from CRIM and CEM, respectively.

9. Semantic mediator

Developed by: UB	WT: 7.1	Timeline: M18-M42
Integrates with: Vocabulary service Data node connector		Versions: v1.0 (M42)
Models used: CDIM, CDIM-DSM, DSM		

The semantic mediation service is the software component that translates TRANSFoRm user requests into real database statements for execution on each data source. One instance of the service is deployed into each data source site, which generates specific queries for that data source. TRANSFoRm develops archetypes to help describe eligibility criteria and data extraction requests. The archetypes are implemented using openEHR archetype definition language (ADL) and have bindings to TRANSFoRm CDIM ontology. The semantic mediation service provides Java API which accepts TRANSFoRm archetypes in ADL format and returns query statements in string format. The service refers to the local data source model (DSM) and CDIM-to-DSM mapping for query generation and the Vocabulary services for translating coding schemas. The individual mappings are encoded as XML files which are installed along with the semantic mediation service inside LexEVS server.

10. Vocabulary service

Developed by: UB	WT: 7.2	Timeline: M1-M18
Integrates with: Semantic mediator Query workbench Infrastructure for study design and deployment		Versions: v1.0 (M18)
Models used: N/A		

Integrated vocabulary service is designed to allow end users to search and retrieve clinical vocabulary contents through both a web interface and web services API. The vocabulary content of the service includes UMLS Metathesaurus subset of clinical concepts and associated terminologies relevant to primary care clinical practice and research, and other terminologies which are not included in UMLS, such as Read Codes v2, ATC, BNF, etc. The service uses LexEVS technology to access backend UMLS vocabulary database which is hosted in MySQL and uses direct JDBC connection to access the Read Codes v2 to SNOMED CT mapping table so that Read Codes v2 can be linked into UMLS vocabularies. For other standalone terminologies, the service employs Apache Solr to build and host Lucene index in order to support full-text search. The service exposes a common interface to integrate vocabularies from different backend connections. The web interface is implemented as an Ajax application using Google Web Toolkit (GWT). The web services interface supports both SOAP and RESTful API. The SOAP API is implemented using Metro, the reference implementation of JAX-WS web service stack. The RESTful API is implemented using Jersey, the JAX-RS (JSR 311) reference Implementation for building RESTful Web services.

Semantic mediator uses the service to perform translations between coding schemas used in queries and the ones supported by the local data source. Query workbench uses the Vocabulary services to search for appropriate terms in various coding schemas.

11. Data collection metadata repository

Developed by: UB	WT: 7.3	Timeline: M18-M36
Integrates with: Infrastructure for study definition and deployment		Versions: v1.0 (M36)
Models used: CRIM		

The repository will store and provide vocabulary controlled CDEs and eCRFs contributed by the primary care community to enable clinical researchers to reuse or construct and design new case report forms for capturing and collecting studies' data for the purpose of the TRANSFoRm use-cases. The eCRF metadata standard adopted is ODM 1.3.1. However, questionnaires created using the ODM model do not contain enough information to generate forms in mobile / web applications. There are several ways that specific question can be displayed (e.g. multiple answers might be presented as drop down list, radio buttons list, slider), thus ODM model has to be extended to provide information about GUI. Therefore, the repository needs to store separate rendering elements for use in web and mobile applications

12. Infrastructure for study definition and deployment

Developed by: UB	WT: 7.4	Timeline: M30-M42
Integrates with: Authentication framework Provenance framework Data collection metadata repository eCRF/PROM data collection		Versions: v1.0 (M36) v2.0 (M42)
Models used: CRIM, PROV-CRIM		

This component manages the design and deployment of functional eCRF definitions using eCRF/PROM elements from Data collection metadata repository, consisting of CDEs, ODM elements, and representational GUI widgets. These computable representations are stored in the Study database and used by eCRF/PROM data

collection to generate graphical interfaces, capture, and store patient data. The study participants are assigned user credentials by the Authentication framework.

13. Secure data transport

Developed by: TCD	WT: 7.5	Timeline: M12-M48
Integrates with: Query workbench Data node connector		Versions: v1.0 (M24) v2.0 (M36) v3.0 (M48)
Models used: N/A		

Service-based infrastructure that enables communication between research and clinical systems, providing federated secure data access and query of electronic health record systems and research databanks. Asynchronous messages are used to receive the queries from Query workbench and send them to Data node connector. Integration at both the Query workbench and the Data node connector ends is managed by a middleware proxy library. This library runs local to both components and is used to invoke the set of services and workflows provided by the distributed infrastructure, whilst abstracting the implementation details from each application. V1.0 delivers the basic functionality for retrieving count information from the data sources, v2.0 provides the data extraction, while v3.0 fully links with the Data connector tool.

14. eCRF/PROM data collection

Developed by: WrUT	WT: 5.5, 7.6(1)	Timeline: M24-M48
Integrates with: Authentication framework Infrastructure for study definition and deployment Provenance framework Data node connector		Versions: v1.0 (M39) v2.0 (M42) v3.0 (M45) v4.0 (M48)
Models used: CRIM		

The component that manages the study-specific event-triggered data collection activities. The computable models from Data collection metadata repositories are retrieved inside study definitions and used to construct interfaces for data entry by patients and clinicians on mobile and web platforms. Study participants can then use either the mobile or the web tools to fill in the PROM forms. The clinician can either enter the eCRFs manually in a similar manner, or populate the fields from the EHR system via the Data node connector component. Unlike the Epidemiological study configuration, in this scenario Secure data transport is not used, since the EHR retrieval happens at the clinician's site. The eCRFs and PROMs are embedded into the EHR system by the means of a trigger tool, such as LEPIS. The collected data is persisted inside the study database. Version 1.0 will provide support for basic data collection, version 2.0 will support user information and data storage, and version 3.0 will support data extraction from EHR systems and version 4.0 will integrate with the EHR's triggering mechanism.

15. Data node connector

<table>
<tr><td>Developed by: UNIVDUN</td><td>WT: 7.5</td><td>Timeline: M36-M48</td></tr>
<tr><td colspan="2">Integrates with:
Semantic mediator
Secure data transport
PROM and eCRF data collection
Data sources and EHR systems</td><td rowspan="2">Versions:
v1.0 (M42)
v2.0 (M48)</td></tr>
<tr><td colspan="2">Models used:
CDIM, CDIM-DSM, DSM</td></tr>
</table>

The component acts as the interface between the queries arriving via Secure data transport (or Data collection server in case of the RCT configuration) and the local data source from which data needs to be extracted. This involves the translation of a query posed in CDIM, with some terminology, into an executable form that can be processed by the local data source. Semantic mediator is used in this translation process. The tool also comprises a user-facing console tool residing at the data provider site that displays arriving data queries in a form meaningful to the data controller at the site, allowing them to perform additional scrutiny of each query, if so

desired. In addition to the query formulation in local coding, each arriving entry contains context information for the query: study agreement details, approved person attached to that study, approved organisation attached to that study, and explanation and purpose of the query in natural language. Version 1.0 will support data extraction and integrate with Semantic mediator, while version 2.0 will have a graphical console for the data controller.

2.2 TRANSFoRm Models

In addition to the software tools, the models in TRANSFoRm are also an essential part of the software design and specification tasks. The models form the backbone of the tools and ensure their interoperability both on the conceptual level, and for concrete data exchange tasks. For example, the Query Workbench tool is using the CDIM model to generate the query parameters, and the CRIM model for the query structure. Thus, while there is no integration involved, the models are versioned and tracked together with the tools to facilitate development. Presented here are the models, together with the listing of the components which are based on them or are using them.

CRIM (Clinical Research Information model) underpins all software tools developed in the project and is necessary to integrate the clinical research workflow with TRANSFoRm software platform. CRIM extends the Primary Care Research Object Model (PCROM) [Spe08] to include new information objects, episode of care related objects and high-ranking concepts (areas), while satisfying all the requirements from the two TRANSFoRm research use cases. It is implemented as a set of UML models that are instantiated in the features of TRANSFoRm software tools, such as the Query Workbench and the eCRF Manager. The workflows of user tools, and the data structures involved, are all specified inside CRIM.

CDIM (Clinical Data Integration Model) is a global mediation model using the local-as-view approach, expressed as an ontology and taking a realist approach. It underpins the unified structural/terminological interoperability framework for integrating heterogeneous data sources within the clinical and research settings of primary care. Domain coverage was explored through discussions with primary care physicians in the TRANSFoRm project; by review of research papers and database

metadata; and review of other ontologies. Use-case scenarios were employed to ensure the ontology was fit for purpose. The ontology includes concepts that are especially important to primary care (e.g. episode of care or reason for encounter), but also others to handle temporality in queries (e.g. start and beginning of processes). Other ontologies were imported or specialized to give deeper definition to the concepts in the domain. These included BFO 1.1, OGMS and VSO. The ontology is stored as OWL files and managed using Protégé 4.2. It is deployed using the LexEVS 6 platform.

DSM (Data Source Model) is a UML model that describes the internal structure of data sources such as RDBMS, XML documents and HL7 messages. It is designed to be a target for mapping CDIM concepts to structural elements within the sources and is used by the middleware for query translation. During the design, a range of existing data sources were investigated for their data models and user-defined data types covering both clinical data repositories and genetic data sources. This was done through a review of specifications for various data models and also with the assistance of staff familiar with the data source. Models were instantiated for the data sources at NIVEL, GPRD and Biomina. The instantiated source models are stored as XML files, which can be viewed with any XML reader, such as a web browser. They were generated semi-automatically using a tool for this purpose. The files are managed with an XML editor and will be deployed via the LexEVS 6 platform using a custom loader. The models and tools are versioned and tracked for integration purposes.

CDIM-DSM mapping model is required to express how concepts from the CDIM are related to one or more structural elements of the DSM using a variety of operators. Operators can be assembled in parallel or in series and support numerical and string operations as well as built in functions. Knowledgeable staff at the data sources must comprehend the meaning of the CDIM concept and establish the best match to the structural elements in the data source model and combine these in the appropriate way. The concepts of the CDIM are expressed independently of the data sources so such mappings are not guaranteed to exist. Mappings were instantiated for the data sources at NIVEL, GPRD and Biomina. The mapping models are stored as XML files, which can be viewed with any XML reader, such as a web browser. They are

generated semi-automatically using a dedicated tool and are deployed using the LexEVS 6 platform, via a custom loader. The models and tools are versioned and tracked for integration purposes.

CEM (Clinical Evidence Model) provides an ontology of clinical evidence to represent the clinical knowledge needed to provide diagnostic decision support for three defined patient safety use cases: abdominal pain, chest pain and dyspnea. The evidence model contains two core models, a general model of evidence, and a model to represent clinical prediction rules. The general model of evidence allows us to represent relationships between presenting patient complaints, formal diagnoses, and the diagnostic cues that support those diagnoses. The clinical prediction rule model allows us to represent formal clinical prediction rules that implement formal scoring schemes and clinical interpretations. The evidence for these models is obtained from review of clinical evidence from literature and also from a data-mining module that allows population of the models from clinical evidence derived from analysis of electronic sources of primary care data.

PROV-CRIM is the bridging model that maps the key concepts from CRIM to the Open Provenance Model (OPM) to support provenance capture in tools that are based on CRIM, such as Query Workbench, Data Quality tool, and eCRF components. The core OPM concepts such as entities, actions, and agents are extended, or annotated by a subset of CRIM elements. The model is deployed as a web service inside the Provenance Framework that allows the instantiation of the model templates. The approach has been published in [Cur11].

PROV-CEM is the bridging model that maps the key concepts from CEM to the OPM for provenance capture in diagnostic support tools and clinical evidence service. Similarly to PROV-CRIM, the basic OPM elements are extended or annotated by CEM elements.

2.3 Development environments

TRANSFoRm software is developed at several sites, each with its own development practices, depending on the size of the team at the site and the nature of the

technical outputs, but conforming to the common Software Quality Assurance standards described in [MS4].

Broadly speaking, larger sites (UB, TCD, and WRUT) use agile development methodologies, while smaller ones, with a single developer, rely on iterative prototyping with frequent feedback from partners and internal milestones based on the schedule of technical meetings.

The basic documentation produced across all technical teams consists of:

- **Technical design documentation** is created at various stages of software development. Initial requirements analysis is performed with end users (clinical or technical) at the start of any technical component to help build a list of requirements suitable for an overall project specification document. The resulting document enables collaborative design decisions before any code is written. As part of this specification diagrams of the data models may also be constructed together with the overall architecture of the software, possible workflows and detailed use-cases. Standard UML is used where appropriate. As software requirements are likely to change over time based on user feedback or any technical issues, the specification is updated regularly. The specification itself may be made up of several design documents depending on the design tools used.
- **Software documentation** is made throughout development in the form of comments and standardised comment blocks within the source code. In typical development scenarios a JavaDoc style system of annotation is used for all code. Inline comments are used to explain algorithms and other complex code blocks. All source code is annotated with the author, name of the code block (class or otherwise), description of the functionality of the code and all input/output fields.
- **User manuals** are typically created at the completion of a staging version of any software under development to allow users to become familiar with any software before it is pushed to the production environment. In general, user manuals are in human readable HTML or PDF format with suitable structure to

allow easy navigation of core usage scenarios. All user manuals are tailored to the intended audience, thus any manual for software for integration with work by other developers is different in style and complexity to that of documentation for end users.

Major code versions and associated documents are submitted to a central source code versioning system, implemented via a Subversion installation hosted by KCL (svn://mdmi.kcl.ac.uk:51516/transform). Subversion[5] (SVN) is a version control system that allows ordered backups to be made. Multiple users can commit changes they make to files as well as roll back any changes made, all the while maintaining a single consistent version.

The specificities of each development teams' internal procedures are described below.

2.3.1 University of Birmingham (UB)

University of Birmingham (UB) TRANSFoRm software development follows a simple agile development philosophy with a standard structured release and deployment workflow. Software is developed as part of an iterative process, with rapid prototyping of new features independently of core software products. Research into new software techniques, especially UI development, is performed separately to development of core technologies used by other TRANSFoRm members.

UB uses a Development - Staging - Production workflow for all software and hosting of web-based applications. Separate servers are used for each stage of development, with integration between developers happening at the staging level using source repository frameworks such as Subversion.

Initial research by individuals takes place at the development level, where virtual machines or other development architectures are used to produce early versions of

[5] http://subversion.apache.org

software. The collaborative work environment at UB encourages developers to discuss their ideas and make changes to their own development software without affecting staging or production versions of UB's TRANSFoRm software output. At development level it is expected that changes can break existing code, which is allowed as all software developers are working on their own codebase.

All software (web based or otherwise) is systematically tested at the development stage before it is pushed to SVN, including integration testing. Comments on each commit to SVN are required for all changes, no matter how small.

After confirming the quality of newly developed software in a development environment, the SVN is used to push all changes to a staging environment. This is considered the first "public" showcase of any work and is considered a repository for all stable code. No changes are made to software at the staging level but continued development of software components is allowed at the development level.

User acceptance testing is performed on the codebase at the staging level. Extra audit and debugging overheads are allowed as these can be used to profile and further improve any code for eventual production release.

On acceptance of software at the staging level by end users, the SVN is used to push the same code to a production environment. This is unlikely to take place without further changes at a development level based on user feedback. In all cases any changes based on user requirements must first be tested successfully at a development level before pushing to the staging environment for more user acceptance testing.

In addition to the standard documentation the following is also produced:

- **Information governance** documents describe the accepted and implemented policy regarding the different dimensions of access to data, specifically patient identifiable data in any software involving real patient records. Compliance with institutional policy and current legislation are described.

- **Mockups** are produced on completion of requirements analysis and are used as a starting point for development of user interfaces and software workflows. Software such as Balsamiq is used to build simple diagrams of future software interfaces in a format suitable for end users. As well as enabling design decisions for refining user interface requirements, mockups are used to enable discussion of integration of separate software components. In this case non-UI diagrams such as block diagrams are used to describe any required interactions.

2.3.2 Royal College of Surgeons in Ireland (RCSI)

The software development carried out by Royal College of Surgeons uses a rapid prototyping approach for development. Models or software components are prototyped at an early stage and demonstrated to the wider TRANSFoRm teams for feedback and update accordingly. This iterative process is repeated in cycles as required.

Because RCSI has a small development team (one RCSI developer), local development is done on a single dedicated development server (no virtual machines are used). Unit testing is carried out on local development code bases. Inputs from other contributing teams (one other TCD developer), such as Trinity College Dublin (TCD), are consolidated from their own local development environment manually into a single local development environment.

Multiple development components are versioned and brought together in an integration environment and system testing is carried out on the integrated code base. Subversion is used as a repository management tool locally with separate areas for development, integration, staging and production code bases. User acceptance testing takes place in the staging area, before the code is transferred to a production environment. At this point it is considered ready for testing/integration with the wider TRANSFoRm code base.

2.3.3 University of Dundee

The University of Dundee and INSeRM are jointly responsible for the CDIM model, and the two developers, one from each institution, are communicating regularly via email, and conference calls. While CDIM is not a software product per se, its stable versions are getting regularly committed to the central SVN. In addition to this, UNIVDUN produces the data source models and mappings of CDIM to these models. All models have been developed iteratively using either Protégé 4.2 or Enterprise Architect 9. The development of instances of source and mapping models has been through prototype tools developed at Dundee.

The UNIVDUN developers are also responsible for the Data Node Connector and they are based at the Health Informatics Centre, University of Dundee as part of a multi-skilled development team operating within an agile environment separating out software development, integration testing and deployment. The developers attached to TRANSFoRm, prototype and test software components before release to the TRANSFoRm SVN source repository where integration efforts are supported. Tools utilised to support prototyping and testing include Trac & JIRA for tracking software changes via a ticket system, SVN and Git for code version control, unit testing facilities, and a development Team Wiki for information sharing. Regular meetings are held with more senior local TRANSFoRm members to specify and manage development needs. The team uses a variety of tools, development languages and RDBMS, both Java and Microsoft oriented. On acceptance of software after integration testing with other members of the TRANSFoRm team the software is deployed. Test failure results in code going through the cycle again.

All HIC projects are fully supported by documentation, from initial requirements specification to a management plan and user guides for clients, conforming to general TRANSFoRm documentation requirements.

2.3.4 NIVEL

The software development carried out by Netherlands Institute for Health Services Research (NIVEL) follows agile development philosophy with a structured release and deployment workflow, adjusted to a single developer environment. Software

components are prototyped at an early stage and demonstrated to the TRANSFoRm researchers for feedback and updated accordingly. All the development is done on a single development server.

2.3.5 Trinity College Dublin (TCD)

Trinity College Dublin (TCD) TRANSFoRm software development follows a simple agile development process with a standard structured release and deployment workflow. Software is developed as part of an iterative process, with prototyping of new features independently of core software products. This process uses a Development - Staging - Production workflow for all software and hosting of services. Separate machines are used by each developer, with integration between developers happening at the staging level using source repository frameworks such as Subversion.

At the development level, virtual machines or other development architectures are used by individuals to produce early versions and prototypes of software. At this stage, all developers on the team work on their own codebase, allowing individuals to make changes without impacting the staging or production versions of TCD's output. All software is tested with existing software at the development stage before it is pushed to SVN, particularly integration testing, with comments required on each commit.

After confirming the quality of newly developed software in a development environment, the SVN is used to push all changes to a staging environment. This is considered the first "public" showcase of any work acts as repository for all stable code. No changes are made to software at the staging level but continued development of software components is allowed at the development level.

Integration testing is performed on the codebase at the staging level as the nature of the software produced necessitates integration with a number of other TRANSFoRm development teams.

On acceptance of software at the staging level by partner teams, the SVN is used to push the same code to a production environment. This is unlikely to take place

without further changes at a development level based on feedback received during integration. In all cases any changes based on requirements must first be tested successfully at a development level before pushing to the staging environment for more integration testing.

2.3.6 Imperial College London (ICL)

The team at Imperial College consists of one developer, and it uses an iterative prototyping approach, whereby early functional specifications are discussed with the project partners before producing initial technical designs and implementations that are then incrementally enriched with both technical functionality and additional models. The functional specifications are mapped onto existing TRANSFoRm models where appropriate, such as CRIM and Clinical Evidence Model, and every increment is tested with respect to the other TRANSFoRm components affected by the change in the increment, or all in case of fundamental changes. A staging area is used for other components to connect to the software and perform integrative tests where required, before moving stable versions into the production area.

An internal Subversion repository is used for tracking code versions, with stable releases committed to the central Subversion repository at KCL.

2.3.7 Wroclaw University of Technology (WRUT)

Wroclaw University of Technology follows an agile development methodology, with six developers on the team. A constant exchange of information is taking place within the team and progress and short term plans are established during regular weekly meetings. These meetings are also used to discuss team-level issues. Each individual software component worked on by WRUT has a separate team associated with it, which meets at least two times a week.

Subversion is used as the local version control system, and a SharePoint instance has been configured for the management of project documentation. A backup strategy has been introduced for the most crucial elements of the infrastructure (for instance, for WT 7.6 full backup is made every 12h – encrypted and stored on external server).

3. Integration plan

In this section, we present the integrations that are required in the project, together with the overall strategy for managing the process. All software integrations in the project are happening in years 3 and 4, and a three-month window is assigned to each. Any shifts in the timetable are discussed at the monthly Technical Committee meetings and agreed with the Scientific Project Manager, who is overseeing the whole process.

3.1 Integration process

TRANSFoRm is a stack-based software infrastructure, where components can be combined to create solutions for a particular purpose, such as the epidemiological study of the diabetes use case, or the clinical trial of the GORD use case. Therefore, not all software components described above need to be integrated, since there is no monolithic TRANSFoRm software package. The majority of integrations happen between the infrastructure components (vocabulary services, provenance framework, authentication framework, and the secure data transport layer) and user-facing applications.

Each integration is pair-wise, performed on two components, by the two teams responsible. One team is designated as the lead, and provides the Software Integration (SI) staging environment on their site – in most cases, this is the development site of the more complex component.

Three months are normally assigned to each integration task, intended to run in parallel to other development happening. The technical details of the task are specified in an initial kick-off meeting, which is typically face-to-face. The integration points and internal versions of the software to be integrated are agreed on and described in detail, with 2-3 milestones set with target dates. The integration manager and technical leads from both sides agree on the plan, and tasks are assigned to named individuals. Depending on the scale of the meeting, an optional hack-a-thon can also take place. In the follow-up period, the teams work in continuous contact (email/phone/webex as required), overseen by the integration manager, until an integrated version is produced.

The integration tests are also agreed at the kick-off meeting and included in the specification. These are typically based on the representative use case scenarios. The site in charge of the staging environment is responsible for running integration tests, and the finalized version, together with the integration report, is submitted to the central SVN repository at KCL.

3.2 Integration tasks

The list of integrations required is presented below, with a technical description of what is involved in each case. The timing of integration tasks for the next 12 months is reviewed by the Scientific Project Manager every three months, at the Face-to-face meetings, and agreed with the teams.

I1: Diagnostic evidence service (3) and Authentication framework (1)

Partners: RCSI, TCD	Host site: RCSI	Timeline: M36-M39

The evidence service relies on the authentication framework for user login and to determine the access rights for accessing and updating the rules. This is done via the user roles, as defined in the Operational Security Policy document. The roles currently defined for the Evidence Service include: Decision support consumer with read-only access to evidence repository, Ontology curator with read/write access to ontology and Analyst with read/write access to rule repository.

I2: Diagnostic evidence service (3) and Rule data mining tools (4)

Partners: RCSI, WRUT, IC	Host site: RCSI	Timeline: M38-M40

The Diagnostic evidence service will integrate with data mining tools to discover and update new clinical evidence. The integration will provide a REST based web service update interface that will be included in the version 2.0 of the service. Rule data mining tools, implemented using the KNIME workflow system will be invoked through a custom scripting extension. The data mining interface will allow for clinical review and manual curation of rules before they are deployed to evidence service.

I3: Diagnostic evidence service (3) and Provenance framework (8)

Partners: RCSI, IC	Host site: RCSI	Timeline: M40-M42

The actions in the evidence service need to be captured in the provenance framework. This will be done using the diagnostic support provenance template, with individual actions mapped to API calls in the provenance framework that construct fragments of provenance logs, compliant with the template.

I4: Diagnostic evidence service (3) and Diagnostic support interface (6)

Partners: UB, RCSI	Host site: UB	Timeline: M42-M45

UB's front-end for diagnostic support will communicate with the evidence service using an API that will be provided in the version 2.0 of the Evidence Service. The evidence service will offer the rule model that will be used by the diagnostic support interface to construct data collection elements and feed the collected information back to the evidence service.

I5: Diagnostic support interface (6) and EHR systems

Partners: UB, EHR vendors	Host site: EHR vendors	Timeline: M45-M48

The diagnostic support front-end will need to be integrated into the EHR systems. CSC's iSoft in Netherlands will be one of the initial pilot projects, with a dedicated site at which to perform the initial integration. The work consists of using LEPIS version 2, or an alternative triggering tool within the EHR system, to notify the TRANSFoRm diagnostic support component that a suitable patient has presented to the user.

I6: Query workbench (7) and Provenance framework (8)

Partners: UB, IC	Host site: UB	Timeline: M30-M33

Query workbench user actions, creation and editing of studies and query criteria, are captured by the Provenance framework, according to a dedicated provenance template. The integration work consists of mapping the individual actions to API calls on the provenance framework that construct provenance traces from the template.

I7: Query workbench (7) and Authentication framework (1)

Partners: UB, TCD	Host site: UB	Timeline: M33-M36

Query Workbench is using the Authentication Framework to authorize its users, according to the roles specified in the Operational Security Policy, namely Chief

Investigator, Principal Investigator, and Authorized Researcher. This is done via a series of calls to the Web Service API provided by the Authentication Framework.

I8: Data quality tool (5) and Query workbench (7)

Partners: NIVEL, UB	Host site: N/A	Timeline: M36-M38

Data quality tool applies various criteria to score individual practices with a data quality score, reflecting the percentage of patients (or count) that meet the quality criteria and other information, such as availability of linked genetic data. This information is passed on to the Query workbench as the list of practices, with associated criteria collection expressed in CDIM, and is used to inform the query constructed in the Query workbench. The integration consists of loading this information into the Query workbench interface by copying-and-pasting or using an XML file. Since there is no direct link between the two pieces of software, no host site is needed.

I9: Data quality tool (5) and Authentication framework (1)

Partners: NIVEL, TCD	Host site: NIVEL	Timeline: M38-M39

The Data Quality tool relies on the Authentication Framework to identify the users logging into the system. The access to the Data Quality tool is an atomic role, according to the Operational Security Policy.

I10: Data quality tool (5) and Provenance framework (8)

Partners: NIVEL, IC	Host site: NIVEL	Timeline: M39-M41

The actions in the Data quality tool are captured by the Provenance framework, according to the provenance templates defined in [D5.2]. The integration work consists of mapping the individual actions to API calls on the provenance framework that construct provenance traces from those templates.

I11: Query workbench (7) and Secure data transport (13)

Partners: UB, TCD	Host site: UB	Timeline: M37-M40

The data that the Query Workbench is retrieving from the data sources is transported via the Secure Data Transport mechanism. The queries are encrypted and passed

on to the data transport, which delivers them to the data source endpoint. Once the reply has been received from the data source, asynchronously, the Query Workbench is notified and retrieves the data.

I12: Provenance framework (8) and Authentication framework (1)

Partners: TCD, IC	Host site: IC	Timeline: M35-M38

The Provenance Framework stores the SAML identifiers used to authenticate actions in various TRANSFoRm tools. No security information beyond these certificates is stored, so for any further lookup and analysis, the data is extracted from the Authentication Framework. This is done via a secure authenticated Web Service API provided by the Authentication Framework.

I13: Secure data transport (13) and Data node connector (15)

Partners: TCD, UNIVDUN	Host site: UNIVDUN	Timeline: M41-M44

Secure data transport component passes the queries to the local instance of the Data node connector sitting on the data source site. Once the response has been formulated, the Data node connector sends it to the originating host using the Secure data transport messaging framework.

I14: Study definition and deployment infrastructure (12) and Authentication framework (1)

Partners: UB, TCD	Host site: UB	Timeline: M39-M41

The study definition and deployment infrastructure will link to the authentication framework both to authenticate research users of the system and to allow clinicians to create users for patients who are going to enter their PROM data into the data collection framework.

I15: Study definition/deployment infrastructure (12) and eCRF/PROM data collection (14)

Partners: WRUT, UB	Host site: WRUT	Timeline: M42-M45

The study definition and deployment infrastructure will use the information in the eCRF metadata repository to create new studies and allow clinicians to assign individual patients to studies. Once the studies and data elements have been defined and deployed, the data collection server will publish the patient and clinician

interfaces and start collecting the data.

I16: Data node connector (15) and eCRF/PROM data collection (9)

Partners: UNIVDUN, WRUT	Host site: WRUT	Timeline: M42-M45

Some parts of eCRF data collection will happen by extracting data from the local EHR system. Since this is happening on the clinician's site, secure data transport is not used, but a direct query is sent from the data collection server to the Data node connector using the same API that the Secure data transport uses.

I17: Study definition and deployment infrastructure (12) and Provenance framework (8)

Partners: UB, IC	Host site: UB	Timeline: M41-M42

The actions performed in the study definition and deployment will be captured in the provenance framework using the defined provenance template. Specific actions will invoke the API calls on the provenance framework to create provenance graph fragments compliant with the template.

I18: eCRF/PROM data collection (14) and Authentication framework (1)

Partners: WRUT, TCD	Host site: WRUT	Timeline: M42-M43

The data collection component will link to the authentication framework to authenticate patients entering their PROM data into the data collection framework.

I19: eCRF/PROM data collection (14) and Provenance framework (8)

Partners: WRUT, IC	Host site: WRUT	Timeline: M43-M44

The data collection actions will be captured in the provenance framework using the defined provenance template. Similarly to other provenance capture scenarios, data collection actions will invoke the API calls on the provenance framework to create provenance graph fragments compliant with the template.

I20: Data node connector (15) and Semantic mediator (9)

Partners: UNIVDUN, UB	Host site: UNIVDUN	Timeline: M39-M42

Data node connector will use the semantic mediator to translate structural and semantic aspects of queries into the local representation language, effectively

mapping CDIM concepts into the local DSM, while relying on the Semantic mediator to take care of any terminology translations by invoking Vocabulary services.

I21: eCRF/PROM data collection (14) and EHR systems

Partners: WRUT, EHR vendors	Host site: EHR vendors	Timeline: M45-M48

Data collection module will need to be integrated into the EHR systems on which the GORD RCT use case will be conducted. CSC's iSoft in Netherlands and Germany will be one such system, with others to follow. The work consists of using LEPIS or an alternative triggering tool in the EHR system to notify the TRANSFoRm component that a suitable patient has presented to the user.

In integration tasks, we separately consider the single-site integrations that occur between software components developed by the same team. Such integrations are managed by the local teams as part of their development process and treated as internal milestones rather than processes. These include:

I22: eCRF Metadata Repository (11) and Study definition and deployment infrastructure (12) – UB, M36

Ensuring that study elements are compatible with the needs of the infrastructure.

I23: Query workbench (7) and Vocabulary services (10) – UB, M24

Providing the functionality inside the Query workbench to lookup terms from the Vocabulary services using its API.

I24: Semantic mediator (9) and Vocabulary services (10) – UB, M39

Allowing the Semantic mediator to use Vocabulary services for terminology translations via the latter's API.

I25: Diagnostic evidence service (3) and Clinical evidence repository (2) – RCSI, M30

Ensuring the Diagnostic evidence service can read and write entries from and to the clinical evidence repository.

I26: Infrastructure for study design and deployment (12) and Vocabulary services (10) – UB, M36

Enabling the study creators to lookup terms in various terminologies using the Vocabulary services.

Integration timeline

The overview of the integrations is shown in Figure 6. Blue bars denote integrations related to the diagnostic support configuration, yellow bars are specific to the epidemiological study (Diabetes use case), and red bars denote integrations relevant for the randomized clinical trial (GORD use case). Grey bars are relevant for multiple configurations.

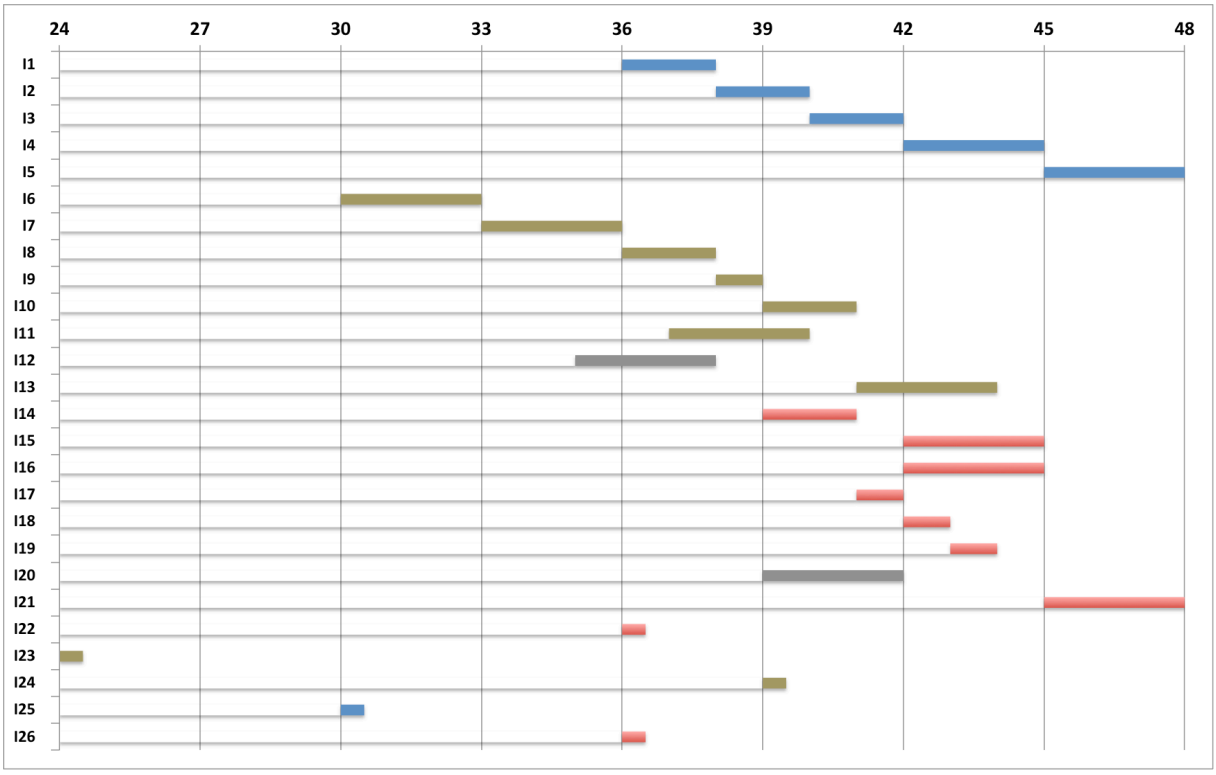

Figure 6: Integration timeline

Due to the modular design of the overall TRANSFoRm system, there are few ordering dependencies between the integrations. Within the diagnostic support configuration, **I5**, the embedding of Diagnostic support interface into the EHR system, is dependent on **I4**, which is the successful integration of the interface with the underlying Evidence service.

In the epidemiological study configuration, used in the Diabetes use case, **I13** depends on **I11**, to ensure the correct format of queries arriving to the Data node connector.

The clinical trial configuration, used in the GORD use case, has a slightly more complex dependencies, since the integration of Study definition and deployment with the Authentication framework (**I14**), has to precede the integration of Study definition and deployment with the eCRF/PROM data collection (**I15**), which in turn has to precede the integration of the latter with the EHR systems (**I21**).

4. Integration activities update for year 3

The following integrations have been completed in year 3 (M24-M36) of the project:

- I6: Query workbench and Provenance framework
- I7: Query workbench and Authentication framework
- I22: eCRF metadata repository and Study definition and deployment infrastructure
- I23: Query workbench and Vocabulary services
- I25: Diagnostic evidence service and Clinical evidence repository
- I26: Infrastructure for study design and deployment and Vocabulary services

The following integrations are either in progress or due to start at M36:

- I1: Diagnostic evidence service and Authentication framework
- I8: Data quality tool and Query workbench
- I12: Provenance framework and Authentication framework

The list of integration design meetings and hackathons is given in Table 2. Some integrations have been discussed before their due start date to ensure teams are kept abreast of the latest developments.

Table 2: Integration meetings in Year 3

Date	Place	Integrations	Participants
13/06/2012	*Utrecht F2F*	*I6*	*UB, TCD*
02/08/2012	*Webex*	*I6, I7*	*UB, TCD, IC, UNIVDUN*
20/08/2012	*Webex*	*I6, I7*	*UB, TCD, IC, UNIVDUN*
21/09/2012	*Malta F2F*	*I1*	*TCD, RCSI*
21/09/2012	*Malta F2F*	*I11*	*UB, TCD*
05/04/2013	*London F2F*	*I15*	*UB, WRUT*
08/04/2013	*Webex*	*I8*	*UB, NIVEL*
16/04/2013	*Webex*	*I15*	*UB, WRUT*

5. Summary

This document presented the software integration plan for TRANSFoRm software components, reflecting the current status of the plan as of M36 of the project.

TRANSFoRm is a digital infrastructure organised around three configurations: Epidemiological Study (supporting the Diabetes use case), Randomized Control Trial (supporting the GORD use case), and the Diagnostic Support (evaluated with the industrial partners). These three configurations were described, together with the updated list of all planned software components and descriptions of how they fit into those configurations. Since TRANSFoRm is a model-based system, the underlying models were also described.

The development practices relevant to the integration were described, both across all teams, and on the levels of individual teams. The latter tend to differ based on the size of the teams and the nature of their tasks.

The integrations in the system are pair-wise, between individual components, since they typically consist of service-based communications, and there is no monolithical software towards which we are heading. Each of these was described in detail, together with the overview timeline that includes the critical dependencies.

The Software Integration Plan is edited by the Scientific Project Manager and shall be reviewed tri-monthly, at each project Face-to-Face meeting, and updated as deemed necessary.

References

[Cur11] Provenance Model for Randomized Clinical Trials. Curcin V, Danger R, Kuchinke W, Miles S, Taweel A, Ohmann C. in Data Provenance and Data Management in eScience, Springer Berlin Heidelberg 2013. DOI: 10.1007/978-3-642-29931-5_1

[D1.1] D1.1: Detailed Use Cases, TRANSFoRm WP1, 2011. Available from: https://transform.kcl.ac.uk/groups/finaldeliverableandmilestonedocsfirstreportingperiod/

[D6.2] D6.2: Clinical Research Information Model, TRANSFoRm WP6, 2012. Available from: https://transform.kcl.ac.uk/groups/reportingperiod2draftdeliverablesandmilestones/

[D6.4] D6.4: Integration Models, TRANSFoRm WP6, 2013. Available from: https://transform.kcl.ac.uk/groups/period3deliverablesandmilestone/

[MS4] MS4: Software Quality Assurance Policy, 2011. Available from: https://transform.kcl.ac.uk/groups/finaldeliverableandmilestonedocsfirstreportingperiod/

[Spe08] The Primary Care Research Object Model (PCROM): a computable information model for practice-based primary care research. Speedie SM, Taweel A, Sim I, Arvanitis TN, Delaney B, Peterson KA. J Am Med Inform Assoc. 2008 Sep-Oct;15(5):661-70. doi: 10.1197/jamia.M2745.

Abbreviations

CDIM	Clinical Data Integration Model
CDIM-DSM	Clinical Data Integration Model – Data Source Model bridge
CEM	Clinical Evidence Model
CRIM	Clinical Research Information Model
DSM	Data Source Model
DSS	Diagnostic Support System
eCRF	Electronic Case Report Form
EHR	Electronic Health Record
IC	Imperial College London
KCL	King's College London
PROM	Patient Reported Outcome Measure
RCSI	Royal College of Surgeons in Ireland
RCT	Randomized Clinical Trial
TCD	Trinity College Dublin
UNIVDUN	University of Dundee
WRUT	Wroclaw Institute of Technology

Diagram notation