Introduction

Representation and execution of research alerts is an important challenge in clinical research informatics. Many institutions address this with locally developed tools which are often designed specifically for the healthcare domain. Despite initiatives such as BRIDG or Arden syntax, there is no widespread use of a single standard or toolkit and no significant alert logic sharing across EHR or CTMS vendors. We have addressed the problem of research alerts representation and execution using workflow technology (WT) [1,2]. WT has been used outside healthcare in many industries (e.g., banking, manufacturing or shipping) and has a well established set of standards, including the XML process definition language (XPDL; see Figure 2 for XPDL schema). We demonstrate its use for representation, sharing and execution of research alerts. In contrast to BRIDG or Arden Syntax standards, there are several commercial workflow software packages available on the market today, as well as open source options.

Methods

Our framework, called RetroGuide, at its core utilizes the open source Enhydra workflow suite (Figure 1). The use of RetroGuide involves three phases:

1. **Editing**: A user authors a research alert in a graphical workflow editor (in WT terms such an alert module would be called a process definition). It uses a step-based paradigm to represent logical steps and operates on a single patient level.

2. **Retrospective testing**: A user executes the flowchart on a testing cohort of patients. A set of audit trail reports generated by each node in the flowchart offers insight into how well the alert performs on retrospective data. This alert testing step is optional and may not be required for simple alerts. For complex alerts, it allows fine tuning of trigger or interim alert logic and gives the user the ability to explore and fully utilize available coded EHR data related to the alert logic. Users can go through several iterations of alert authoring and testing.

3. **Deployment (prospective use)**: A user loads the module into the engine and registers all trigger or relevant EHR events with an event listener component. The deployed modules will trigger, respond to further relevant events and depending on the logic eventually perform the user-authored intervention action (active or passive alert).

Results

We linked a workflow engine with an EHR system first in July 2009. Since then, we deployed in pilot feasibility testing 2 processes: LDL check in adults and a surgical procedure monitoring process (Figures 3 and 4). In the month of September, 2009, the engine processed 17,464 EHR events (Figure 5).

Discussion

We have successfully piloted implementation of a workflow engine within a healthcare institution. A flowchart-based modeling paradigm is used for representing clinical logic. Our methodology uses standard-based workflow technology and can be adopted by other healthcare institutions (e.g., CTSA sites). The RetroGuide graphical modeling paradigm has been evaluated in a prior study at Intermountain Healthcare (retrospective execution) with favorable results indicating high user friendliness of the flowchart-based logic modeling [3]. We have demonstrated the same use with EHR data at Marshfield Clinic and fully implemented the prospective mode of functionality [4].

Acknowledgment

We would like to thank numerous individuals from Marshfield Clinic IT department and Biomedical Informatics Research Center for their support during this project. VH was supported by the Morgridge Institute for Research.

References


Contact

Vojtech Huser, MD, PhD
huser.vojtech@marshfieldclinic.org
http://www.marshfieldclinic.org/birc

Fig. 1: System architecture: Based on the mode (retrospective vs. prospective), the system works with a data warehouse or directly with the host EHR system.

Fig. 2: Graphical depiction of the XML Process Definition Language (XPDL) schema. Key components of the ‘Process’ and ‘Activity’ elements are shown.

Fig. 3a: Web-based admin console of the Enhydra Shark workflow engine (later renamed Together Workflow Server).

Fig. 3b: Thick client version of the admin console. This Java-based client offers better display capabilities. For each process instance, current status can be viewed as a process flowchart.

Fig. 4: Enhydra Jawe workflow editor (later renamed Together Workflow Editor). LDL control module is shown in the main window.

Fig. 5: Graph showing number of EHR events processed by the workflow engine. Presented counts combine trigger events (events which trigger a new process execution) and listen events (events relevant to an existing process instance).