Improving Timely Clinical Lab Test Result Management: A Generative XML Process Model to Support Medical Care

Sureyya Tarkan[1,2,3], Catherine Plaisant[2,3], Ben Shneiderman[1,2,3], and A. Zachary Hettinger[4,5]
[1]Department of Computer Science & [2]Human-Computer Interaction Lab
University of Maryland, College Park, MD 20742
+1 (301) 405-2768
[3]National Center for Cognitive Informatics and Decision Making in Healthcare
[4]National Center for Human Factors Engineering in Healthcare, MedStar Institute for Innovation, Washington, DC
[5]Washington Hospital Center, Department of Emergency Medicine, Washington, DC
{sureyya, plaisant, ben}@cs.umd.edu

ABSTRACT

This paper presents the innovative design and implementation of MSTART (Multi-Step Task Alerting, Reminding, and Tracking), which uses XML specifications of a workflow model. This model specifies a hierarchy of process definitions, which when combined with a database of actors and organizations, provides input for an Interface Generator. This novel software architecture produces a domain independent system that can be widely used and easily modified to generate MSTART applications for business, academic, or other processes. Our focus in this paper is on handling medical laboratory tests to reduce the currently dangerous number of missed laboratory reports. This paper expands on our initial work [31] by describing three approaches to improve test processes so as to ensure that results are returned and acted on: (1) a refined workflow definition of **agent temporal responsibilities** to model more complex processes, (2) a strategy to generate **actor action sheets** that offer appropriate choices at each step, and (3) a **configuration file mechanism** to more accurately predict process result times. While our examples are tied to medical laboratory tests, our design supports many multi-step processes.

Categories and Subject Descriptors

H.5.2. [**Information Interfaces and Presentation**]: User Interfaces – *Graphical user interfaces (GUI).*

General Terms

Design.

Keywords

Electronic health records (EHRs), workflow model, medical laboratory test results, process management, temporal responsibility, cognitive support, user interface.

1. INTRODUCTION

Researchers exploring common problems in outpatient care highlight the importance of timely test result management in primary care [9, 22, 25, 28, 33]. Researchers also suggest that electronic tools, if available, could help medical staff during this complex process [1, 13, 23]. However, other studies show that there is no standard, widely applied method for handling test results [7, 15, 33]. Thus, further research on the design of test result management and evaluation of their use in clinical settings could bring large benefits.

In a recent paper, we presented three ideas to reduce missed results with a system that aids clinical personnel in tracking laboratory tests from order to action completion [31]. First, we defined agent temporal responsibilities using a test process management model. Second, we derived a user interface from that model that could eventually be integrated into electronic health record (EHR) systems. Lastly, we provided retrospective analyses to identify common problems in past test orders.

This paper presents the innovative design and implementation of MSTART (Multi-Step Task Alerting, Reminding, and Tracking), which uses XML to specify a workflow model (Figure 1). The model specifies a hierarchy of process definitions, which provides input for an Interface Generator when combined with a database of actors and organizations. This paper also describes (1) a refined workflow definition of agent temporal responsibilities to model more complex processes, (2) a strategy to generate actor action sheets that offer appropriate choices at each step, and (3) a configuration file mechanism to more accurately predict process result times.

1.1 Background on Missed Results

Wahls [33] defined missed results as "mishandling of abnormal test results", which are then lost to follow-up. In a provider survey, 47% of respondents reported encountering one or more patients (a total of 312 cases in approximately 20,000 visits) with a missed result in the previous 2 weeks [33]. Clinical laboratory tests and imaging studies were the most commonly reported as having missed results. Researchers also found that 37% of providers indicated at least one patient (a total of 276 cases in approximately 20,000 visits) who experienced delays in diagnosis or treatment because of missed test results [33].

Besides self-response survey data from providers, recent studies based on an examination of patient medical records confirmed that errors and delays occurred during screenings for osteoporosis, breast cancer, and colorectal cancer. These results initiated a widespread discussion about the lack of adequate laboratory test result management systems. In one study, Cram et al. [4] reviewed bone density scans of 428 patients for 5 months. Of the 48 patients who were newly diagnosed with osteoporosis, 16 received no treatment recommendation. In 11 out of 16, the medical record showed no result review by providers. In another study, Poon et al. [22] showed that of 126 women with abnormal

IHI'12, January 28–30, 2012, Miami, Florida, USA.

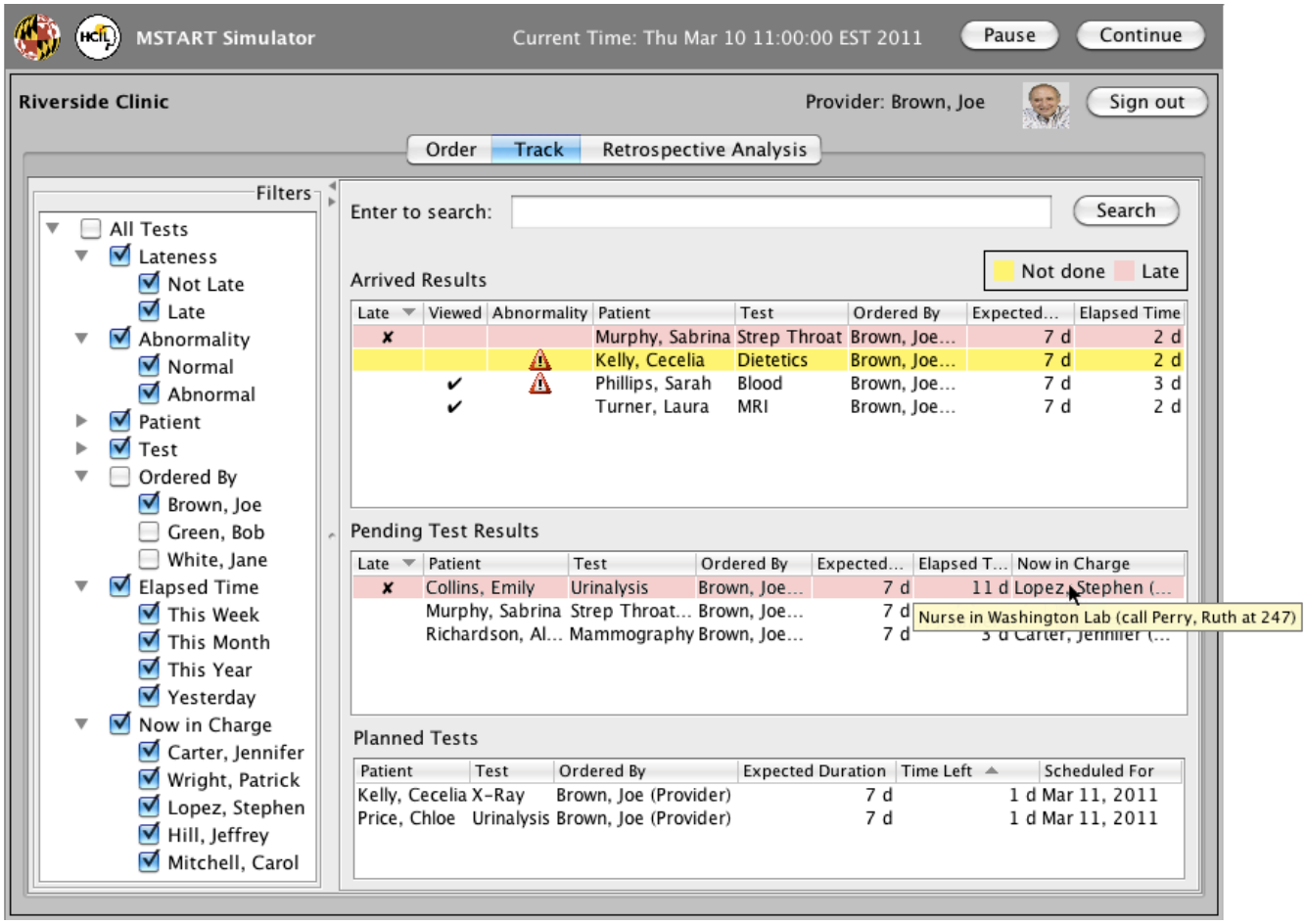

Figure 1. Tracking screen of the user interface of MSTART as seen by the ordering care provider, Joe Brown in Riverside Clinic (order and retrospective analysis tabs not shown). The tables show arrived, pending, and planned tests. By default users only see the work assigned to them. Tests that came back to the physician's office are listed in the table called "Arrived Results", while tests that are in progress but not returned to the physician are shown under "Pending Test Results". Tests that are ordered for the future are listed in the "Planned Tests" table. When the time is up for planned tests, the entries move to pending test results and are no longer listed in the planned list. Similarly, as the result of a pending test return back to the ordering physician, the entry in the pending table moves up to the arrived results. Color-coding indicates lateness: pink for lateness and yellow for incomplete status. Arrived results that are late to be acted upon are marked "x", viewed results are marked with a tick sign, and abnormal results are indicated by an exclamation mark. Tests are removed from the list only when the provider has signed off on the results. Hovering over actors with a mouse shows the role of the person, their institution as well as their manager's name and contact information as a tooltip. All tables are sorted by default so as to visually aid users see important results at the top. Filters on the left allow clinician to customize the display.

mammograms, 45 did not receive appropriate and/or timely follow-up care within 7 months. In addition, the physician did not adequately document discussions with patients in 29% of the cases with abnormal results, and did not document the follow-up plan and 27% of those cases. Furthermore, during a 4-month period using screening cards mailed by a managed care organization Baig et al. [1] identified 544 patients with abnormal fecal occult blood testing, a screening test used for reducing colorectal cancer mortality. A total of 248 of those patients did not undergo a complete diagnostic evaluation after the positive test. Only 50% of physicians were able to provide reasons, the remainder had no follow up due to a variety of causes, including physician, specialist, or practice-related decisions.

Apart from specific diseases, other researchers analyzed the occurrence of problems related to test result management in a broader setting. For example, Singh et al. [28] studied all critical imaging alert notifications in an outpatient setting with an advanced electronic medical record system for 8 months. Of the 123,638 imaging studies, 1,196 images generated alerts via the "View Alert" notification window; 217 (18.1%) of these were unacknowledged within 2 weeks of transmission. Timely follow-up at 4 weeks was lacking in 92 alerts (7.7%) and the occurrence was similar for both acknowledged and unacknowledged alerts. Nearly all missed abnormal results had a measurable clinical impact in terms of further diagnostic testing or treatment. In addition, Roy et al. [24] examined 2,644 patient discharges during 5 months. Of these, 1,095 patients collectively had 2,033 test results arrive *after* discharge. Of these results, 191 were *potentially* actionable, and surveys were sent to 155 primary care physicians. Of the 105 survey responses, physicians indicated they were unaware of 65 results, although 24 of these were actionable

and 8 were urgent. Finally, Schiff et al. [25] linked laboratory and pharmacy databases over a 2-year period to determine if patients with elevated thyroid-stimulating hormone (TSH) received appropriate treatment (the drug levothyroxine). Out of 36,760 unique patients tested for TSH levels, 982 had high levels, including 177 patients who had no associated recorded prescriptions. While 54 patients were lost to follow-up, the researchers contacted 123 patients and found 23 who were unaware of their abnormal results.

These studies all validate that delays and errors happen during the management of test results. Hickner et al. [9] propose that the complexity of the process causes such issues. This includes physicians ordering and reviewing a large number and variety of laboratory tests and imaging studies [4, 11, 23, 28], as well as the variable result arrival times and reporting processes associated with multiple testing locations [7, 9]. As can be seen in the studies cited above, patients can be - and sometimes were - seriously harmed by such errors in results management [9, 28]. As a consequence, failure to follow up on abnormal test results is a frequent cause of medical malpractice litigation [21].

1.2 Motivation

Mold [15] described four steps during which laboratory test result management errors were detected in a family practice setting: 1) Test Tracking: 15% of ordered laboratory tests were not recorded in the logbook and were missing in patients' charts, 2) Patient Notification: 92% of patients received their test results, 3) Documentation of Notification: 40% of charts were lacking sufficient documentation of patient notification (half were not initialed and half were not dated), 4) Follow-up Tracking: 40% of charts had poor documentation of follow-up tracking while 35% of patients did not follow up in 3 months, and 10% followed up late.

1. Pre-analytic phase
    a. Ordering the test
    b. Implementing the test
2. Analytic phase
    a. Performing the test
3. Post-analytic phase
    a. Reporting results to the clinician
    b. Responding to the results
    c. Notifying the patient of the results
    d. Following-up to ensure the patient took the appropriate action based on test results

Figure 2. Laboratory test management workflow

More recent research studies, however, used the workflow depicted in Figure 2 for the management of laboratory test results in an outpatient setting [2, 9]. This workflow captures the fundamental steps in the laboratory testing processes. The workflow begins with a test order by a medical provider, and ends when all the returned results are acted upon. One shortcoming of this workflow definition is that it does not include the interaction between multiple actors involved. For example, the patient and medical provider are both involved in ordering/implementing the test and following up results; implementing the test, performing the test and reporting results are often done outside the office; and the physician has to interpret and respond to the results while an assistant may actually notify and arrange for follow-up with the patient. While one can infer some of this information, explicitly designating a responsible agent is worthwhile since many errors might occur during this complex process, as reported in the background section.

In addition, Figure 2 does not define expected timeframes for each step in the workflow, which is critical when a delay might affect outcomes. We define **agent temporal responsibility** as "every actor involved in a process is assigned an associated duration during which they should complete their own task". We calculate an estimated result time for the overall process when each step of the process is set and a range of process time intervals is provided. At the time of order, the physician obtains a predicted time of completion, taking into account factors such as type of test, holidays, variations in workload, etc.

Using the model in Figure 2, medical researchers, clinicians, and healthcare system developers can group the laboratory tests in two to four categories using a combination of imaging studies (radiology) and laboratory tests (chemistry, hematology, and pathology) [4, 7, 9, 33]. In the simple case, the patient is asked to get a test done, and the provider analyzes the results that arrive back in her/his office. Further details are added based on test type (a radiology study includes a radiologist report, while a urinalysis will require a lab technician report). Other variations may occur depending on the test conducted; for example, a lipid panel order in blood work might necessitate the patient fasting overnight.

Researchers mention that multiple errors can and do occur at each step [7, 9, 15]. For instance, in the pre-analytic phase an order could be lost, or a specimen never drawn, lost or damaged during transport. During the post-analytic phase, the results may be misplaced, not documented in the patient's record, or not followed up with the patient. Using this knowledge, we devised a list of choices available to an actor while performing a task on the test process; we refer to these choices as "actor action sheets".

Carraro and Plebani [2] found that mistakes during laboratory testing occur primarily during the pre-analytic phase (68.2%), with 18.5% and 13.3% during the post-analytic and analytic phase respectively. They suggest the improvement of the total testing process, including pre- and post-analytic phases as a solution to the problem. We propose an advanced system for monitoring in-progress status throughout the laboratory test management workflow. This is similar in concept to popular online commercial services, such as those used to track shipments, check airline flight status, and even order food for delivery. These websites clear state expected arrival times, provide continuous feedback on progress and results, and explicitly identify who is currently responsible for a task, thereby, reducing the likelihood of lost packages, passengers, and merchandise.

The next section will introduce the reader to the relevant work in the area of **test result management** and **clinical workflow management** systems. The rest of the paper is organized as follows. We will first describe the language we use to model processes, from the simplest to the most complicated. Then we will explain the design and implementation of MSTART. Finally, we will illustrate the usability of MSTART via our initial evaluations and the extensibility of our approach to domains other than medicine. We will conclude with a proposal for future work.

2. RELATED WORK

In the past, researchers have approached the problem of missed results by implementing non-electronic solutions. Marcus et al. [14] evaluated two interventions for women with abnormal Pap smears: (i) an intensive follow-up protocol that depends on numerous attempts to contact the patient via mail or phone, (ii)

economic vouchers to compensate for the expenses of follow-up visits. Their study showed that both conditions improved the rate of follow-up as compared to a control group. Sung et al. [30] distributed a survey to providers, to learn their interest in direct reporting of laboratory test results to patients by mail. This research demonstrated that providers preferred direct reporting of normal, rather than abnormal, test results. Secondly, providers were more supportive in direct reporting of results of tests deemed to have less emotional impact. Both of these solutions suggest that including the patient in the process will facilitate test result management. While patient involvement is an important layer of safety, patients are not always able to interpret test results and a robust system should not rely on human vigilance alone to manage results.

Other researchers have attempted to address specific steps of the laboratory test management process. LabRespond [17] is a system that attempts to prevent errors during the shift from analytic to post-analytic phase. More specifically, it helps validate test results sent to clinicians at administrative, technical, sample, patient, and clinical levels so that the frequency of erroneous reporting is reduced. On the other hand, Laboratory Advisory System (LAS) [29] is an interface that assists clinicians in the pre-analytic phase with test selection and in the post-analytic phase with result interpretation. However, we were interested in a more comprehensive approach that takes into account *every* step of the laboratory testing process.

A clinical event monitor [10] is a system in which database operations and external events trigger the evaluation of a condition. The monitor determines whether an action should be performed based on the conditions, consisting of events and patient data. When certain conditions occur, the monitor generates alerts for clinician, patients, or other organizations. LabCheck [6], CLEM [32], ReNAP [23], and many other clinical event notification systems [13, 18, 19, 26] may remind a provider about following up on a test. Since these systems are implemented as rule-based engines accessing a database, they have various complex mechanisms. More importantly, such systems usually do not show the progress of the event, so the clinician has less control and understanding over the entire process. In fact, researchers indicated that they can generate many undesired alerts and cause **clinician alarm fatigue**; this may result in the clinician either bypassing or missing the truly important alarms [12, 28].

As opposed to rule-based architectures, some researchers have applied workflow management techniques to clinical situations. Instead of looking at specific events, these techniques seek to model the general case. For instance, Ling and Schmidt applied time workflow (Petri) nets to an example of a Patient Workflow Management System [5]. They define **firing time intervals**, and show that some transitions are reachable only after certain time periods have passed. This model is good for understanding workflow, however it does not constrain the total time.

Little-JIL [3] is a process definition language to model medical processes. Christov et al. first create a detailed and precisely defined model of a medical chemotherapy process with Little-JIL. Then, they identify process defects and vulnerabilities that pose safety risks, by applying rigorous automated analysis techniques to the original model. Finally, they reanalyze the improved process to show that the original defects are no longer present. The difference between Little-JIL and our approach is that the models created and proven in Little-JIL are the end product, while our focus is on using the model as a means to generate applications for different domains.

Recently, several computer-interpretable clinical-guideline modeling languages have been developed including machine-executable ones that support authoring, editing, and enactment [8, 20, 27]. A **medical guideline** is a document with the aim of guiding decisions and criteria regarding diagnosis, management, and treatment. We depart from these approaches because medical guidelines do not always stipulate a specific process or schedule for performing medical services. **Clinical pathways**, also called **care pathways**, have been introduced to overcome these drawbacks. They use medical guidelines to define and sequence different tasks of healthcare professionals. For instance, Noumeir describes the model of a radiology interpretation process [16] but his specification does not have the necessary elements to generalize to interpretation processes, or imaging study processes.

3. MODELING PROCESSES

This section introduces our process modeling language via the example of laboratory test result management workflow. A knowledgeable system administrator can write the workflow specifications in an eXtensible Markup Language (XML) file that is read by the running application, MSTART.

3.1 Processes

In a recent publication, we described a laboratory test process management workflow model that specifies responsible actors for different time periods during the lifetime of the process [31]. In that model, every test is a *process*.

Every process may have any number of *actor*s that indicate the roles of responsible agents (patient, nurse, physician, etc.) doing their assigned work in concert with other actors. Each actor can sequentially perform one or more *task*s of the main process. For a task, a duration range is set, and used to compute the expected lifetime of the entire process.

Each task can have zero or more *action*s associated with it. Actions are a list of appropriate choices during task execution. In this paper below, we also introduce the notion of *group*s and *option*s. Groups are used to categorize actions that have common properties together. Options on the other hand, enumerate all possible values of an action or a task.

3.1.1 Simple Linear Process

We start with a simple scenario where the process does not depend on other processes. The execution is linear, meaning that to proceed to the next task, the previous task has to be successfully fulfilled. Figure 3 shows an example XML specification for part of a process called "Test". In this simple situation, the physician hands the patient a lab test order and expects a result to arrive in his/her office. The actor "Patient" has a task named "Get Test Done" which has to be completed in 6 days (see *start*, *end*, and *unit* attributes in Figure 3). When the results come back to the "Provider", they must perform the "Analyze Result" task within 1 day.

```
<process id="100" name="Test">
    <actor role="Patient">
        <task id="1000" name="Get Test Done" start="0" end="6" unit="d" />
    </actor>

    <actor role="Provider">
        <task id="1001" name="Analyze Result" end="1" unit="d" />
    </actor>
</process>
```

Figure 3. XML definition of a simple linear process

The durations of the two tasks in Figure 3 allow the calculation of an expected duration for the overall process, "Test", which is

equal to 7 days. This information is later used to predict an estimated result time when the order is placed.

3.1.2 Extended Linear Process

An *extended process* inherits from another process, referred to as *super-process*, by using this super-process' specification. Besides being able to modify any task or action of its super-process, an extended process may extend its super-process by (i) adding more tasks or actions in addition to the ones already defined in its super-process, and (ii) breaking any *container* task of its super-process down into multiple sub-tasks. Each sub-task might be performed by different actors than the actor designated in the super-process specification. The execution is still linear, since tasks and sub-tasks still happen one after the other.

```
<process id="102" name="LaboratoryTest1" isa="100">
    <actor role="Patient">
        <task id="1000" hasa="1006" name="Schedule an Exam and Go to the Laboratory"
            end="4" unit="d" />
    </actor>

    <actor role="Nurse">
        <task id="1000" hasa="1007" name="Register Patient" end="1"
            unit="h" />
        <task id="1000" hasa="1008" name="Draw and Store Sample" end="1"
            unit="h" />
        <task id="1000" hasa="1009" name="Send Sample" end="22" unit="h" />
    </actor>

    <actor role="Laboratory Technician">
        <task id="1000" hasa="1010" name="Examine Specimen and Record Results"
            end="1" unit="d" />
    </actor>
</process>
```

Figure 4. XML definition of an extended linear process

The simple linear process model (Figure 3) can be extended to more concretely defined tests, such as imaging studies or laboratory tests. For instance, Figure 4 partially shows laboratory test tasks. First of all, the reader may notice from the first line in Figure 4 that a laboratory test is-a simple linear test (*isa* attribute refers to process id="100" defined in Figure 3). Then, it elaborates on the patient's task of getting the test done (task id="1000") by splitting this task into five tasks completed by three different actors. Has-a (*hasa*) attributes assign new *id*s to these tasks so that these tasks can be referenced elsewhere (Figure 4). More specifically, the patient schedules an exam with the laboratory and goes to his/her appointment within 4 days. As seen in Figure 4, the actor switches to a generic "nurse" role that could be one individual or a team made of a clerical staff member, nurse and/or a phlebotomist, and the same person does not have to complete each of the tasks. First, the patient is registered within one hour of arrival. The next task is to draw and store the blood sample within an hour, and the last "nurse" task is to send the sample for analysis. Finally, the laboratory technician, after receiving the sample, examines the specimen and records the results within one day. The last task of the provider analyzing results is inherited from Figure 3 without any changes.

Expected duration for the laboratory test is computed as follows. All task durations in Figure 4 are summed together to compute 6 days (equal to patient getting the test done in Figure 3). Because it inherits from simple linear test (process id="100"), the last task of provider analyzing the results from the simple linear test automatically appends at the end of laboratory test, which takes 1 day. Therefore, the total duration for a laboratory test is expected to be 7 days.

3.1.3 Parallel Process

A parallel process that consists of multiple other processes, named *sub-processes*, could not be captured with the aforementioned models. Unlike the previous two processes explained, sub-processes take place autonomously, so some tasks can be executed synchronously. For example, when evaluating a patient's sore throat, a clinician may order a rapid strep throat test and a throat culture at the same time to look for harmful bacteria. The rapid strep test takes approximately an hour, but is less accurate than the throat culture, which may take three or more days for final results (Figure 5). This structure allows different actors in parallel to carry out tests independently of each other, but initiated from the same previous action.

```
<process id="112" name="Strep Throat">
    <subprocess id="40000" name="Rapid Strep" isa="111" />
    <subprocess id="50000" name="Throat Culture" isa="102" />
</process>
```

Figure 5. XML definition of a parallel process

The XML model shown in Figure 5 also captures this. Sub-processes "Rapid Strep", which is-an (*isa*) office test (defined elsewhere in the XML file but shown here with process id="111") and "Throat Culture", which is-a (*isa*) laboratory test (id= "102") constitute a strep throat test. For such parallel tests, expected duration calculations are completed for each test separately.

3.2 Actor Actions

Our model helps physicians to review and take follow-up actions on results [31]. During result review, a physician could be guided to particular process-specific actions. This was accomplished by listing custom actions for tasks in the model.

Our new model allows specifying feasible actions for tasks of every actor, not just of the last actor of the process. If a necessary action is not given, a "write comment" action exists by default for each task. Since processes can extend other processes, it is also possible for tasks to inherit actions from super-processes, change some of those actions as appropriate, or add more process-specific actions. This will be explained in the subsequent subsections.

3.2.1 Actions during Process Result Review

```
<task id="1001" name="Analyze Result" end="1" unit="d">
    <action id="10" name="Access" object="Report" />
    <action id="11" name="No Follow-up">
        <option name="Baseline Value" />
        <option name="No longer under our care" />
        <option name="Opted out not to treat" />
        <option name="Other" />
    </action>
    <group type="Ask Assistant to">
        <action id="12" name="Inform" actor="Patient">
            <option name="Phone" />
            <option name="E-mail" />
            <option name="Mail" />
        </action>
        <action id="13" name="Schedule" object="Visit">
            <option min="1" max="30" name="d" />
            <option min="1" max="52" name="w" />
            <option value="1" min="1" max="30" name="mo" />
            <option min="1" max="5" name="y" />
        </action>
        <action id="160" name="Send" actor="Provider" />
    </group>
    <group type="Order">
        <action id="14" name="Repeat" process="100">
            <option min="1" max="30" name="d" />
            <option min="1" max="52" name="w" />
            <option value="1" min="1" max="30" name="mo" />
            <option min="1" max="5" name="y" />
        </action>
        <action id="15" name="New Test" />
```

Figure 6. XML definition of process result review actions

At the end of a process while reviewing the process results, the actor can be guided to particular process-specific actions, i.e.

common appropriate actions that actor would generally carry out. For instance, Figure 6 partly illustrates leading the physician through the analysis of test results, building from the simple linear process example (Figure 3). The first line shows that providers can "Access" the "Report" *object*, which is the report of the test result that comes back to the office for providers to review.

To improve the interface usability, we have also added support for meaningful grouping of actions, options of actions/tasks, and default values of options. In Figure 6, *group* specifies a set of actions that belong to the same category. The first group in this example is to "Ask Assistant" that encapsulates the following *action*s: (i) "Inform" the "Patient" actor via any of "Phone", "E-mail", or "Mail" *option*s, (ii) "Schedule" a "Visit" object in 1-30 days, 1-52 weeks, 1-30 months, or 1-5 years, with the default *value* being "1 month". Other actions presented to a provider in Figure 5 are grouped under "Order", which encloses actions such as "Repeat" current test (shown in the figure with its process *id*="100"), a "New Test", and so on.

```
<actor role="Provider">
    <task id="1001">
        <group type="Order">
            <action id="27" name="Consult" actor="Orthopedic Surgeon" />
            <action id="28" name="Consult" actor="Physical Therapy" />
        </group>
    </task>
</actor>
```

Figure 7. XML definition of adding an action to a process

Another recent feature is permission to adjust the actions of a super-process to the current process. For instance, Figure 7 indicates how an imaging study adds two new actions to the last task of simple linear test (since an imaging study inherits from a simple linear test). A possible action for the provider analyzing bone imaging study results is to order a consultation to an orthopedic surgeon, or to physical therapy. This action does not apply to other types of tests, so it is part of neither the simple linear nor the laboratory test processes.

3.2.2 Other Actor Actions

The model definition tolerates actions for not only the actor reviewing results as a final task of the process, but also different types of actors involved in the process.

```
<task id="1000" hasa="1007" name="Register Patient" end="1"
    unit="h">
    <action id="29" name="Access" object="Appointment" />
    <action id="30" name="Access" actor="Patient" />
    <action id="31" name="No Appointment">
        <option name="Appointment not found" />
        <option name="Appointment rescheduled" />
        <option name="Appointment canceled" />
        <option name="Patient did not show up for appointment" />
    </action>
</task>
```

Figure 8. XML definition of nurse actions during patient registration

One such example is given in Figure 8 for the individual registering the patient at the lab, i.e. the second task of a laboratory test (Figure 4). While doing this, possible actions could be confirming the appointment, or accessing and updating the patient record. One could likewise enumerate frequent actions (of other clinical personnel) for other tasks of a test.

3.3 Predicting Process Times

Section 3.1 explained how expected duration is calculated for all processes in a way that captures default durations. However, it overlooks the fact that there are weekends, holidays, etc. It does not take into account that some hours of the day, some weekdays, or some months might be busier than others. It also does not take into account past experiences. For this reason, we provide a module to take such special conditions into consideration.

```
<functions>
    <!-- federal holidays -->
    <function name="DATE">
        <argument factor="0">
            <value name="New Year's Day" month="1" date="1" />
            <value name="Martin Luther King, Jr. Day" rank="3" day="Monday"
                month="1" />
            <value name="Presidents Day" rank="3" day="Monday" month="2" />
            <value name="Memorial Day" rank="last" day="Monday" month="5" />
            <value name="Independence Day" month="7" date="4" />
            <value name="Labor Day" rank="1" day="Monday" month="9" />
            <value name="Columbus Day" rank="2" day="Monday" month="10" />
            <value name="Veterans Day" month="11" date="11" />
            <value name="Thanksgiving Day" rank="4" day="Thursday" month="11" />
            <value name="Christmas Day" month="12" date="25" />
        </argument>
    </function>
    <!-- some weekdays are slower/faster -->
    <function name="DAY_OF_WEEK" weight="1">
        <argument value="Saturday, Sunday" factor="0" />
        <argument value="Monday, Tuesday" factor="0.75" />
        <argument value="Wednesday, Thursday" factor="1" />
        <argument value="Friday" factor="1.5" />
    </function>
</functions>
```

Figure 9. XML configuration file to Predict Process Times

Figure 9 shows a simple system configuration file that lets a system administrator customize MSTART for such occurrences. For example, the second argument value of function DATE coincides with the 3rd Monday of January, which is a Federal holiday in the United States known as the Martin Luther King, Jr. Day. Here, the system administrator may list possible cases that could affect duration computations with the *function* tag. The *argument* of a function is an independent variable that takes specific inputs or argument *value*s. *Factor* implies how much that argument influences the time computation. In particular, given a date range, for each day, MSTART can check against every function in the file to identify the relevant factors. MSTART multiplies the factors altogether and the product is the speed of processing (in other words, how many units of work get done) so the higher it is, the faster things get done and vice versa. If no attribute matches for a given function, the factor is assumed to be 1 (default value, i.e. no effect). For instance, Friday, January 1st, 2010 (New Year's Day) results in no work (0 x 1.5 = 0). On the other hand, Monday, April 25, 2011 will end with 1 x 0.75 = 0.75 amount of work because certain tests may be difficult to obtain after the weekend if there is a high demand. To be able to finish 1 unit of work, a third of work time during Tuesday, April 26, 2011 is desired since 0.75 total amount of work could be fulfilled on Tuesday, April 26, 2011.

4. IMPLEMENTATION

This section describes our implementation of the techniques and algorithms for modeling workflow processes, generating actor action sheets, and predicting process times in MSTART. MSTART is an inspirational prototype developed as a Java application that consists of 50 classes, each between 100 and 1000 lines, to illustrate our ideas to domain experts. This novel software architecture produces a domain independent system that can be widely used and easily modified to generate MSTART applications for business, academic, or other processes. MSTART converts the workflow model into a hierarchy of process definitions, which provide input for an Interface Generator when combined with a database of actors and organizations.

4.1 Workflow Element Relationships

MSTART extracts the relationships between workflow processes, actors, tasks, actions, groups, and options shown in the class

hierarchy (Figure 10). This hierarchy is independent of the domain of the specification and is merely based on the XML specification elements (defined in Section 3).

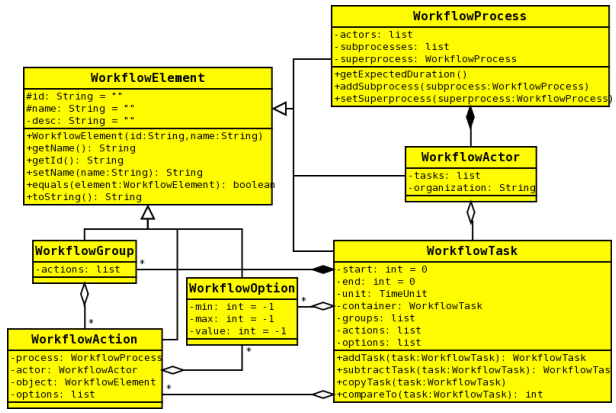

Figure 10. Class hierarchy of workflow element relationships built by MSTART

As seen in Figure 10, *WorkflowElement* is a superclass of all the other classes and it stores the *id* and *name*, along with a *desc*ription. *WorkflowProcess* descends from this class and has a *getExpectedDuration* method that returns the expected duration of a process. This class also saves its super-process as well as its sub-processes. *WorkflowTask* can contain another *WorkflowTask*, and addition and subtraction operations are available for this class. Although the associations are not shown in this diagram, *WorkflowAction* has three attributes referring to a *process*, an *actor*, and an *object*.

4.2 Process Hierarchy

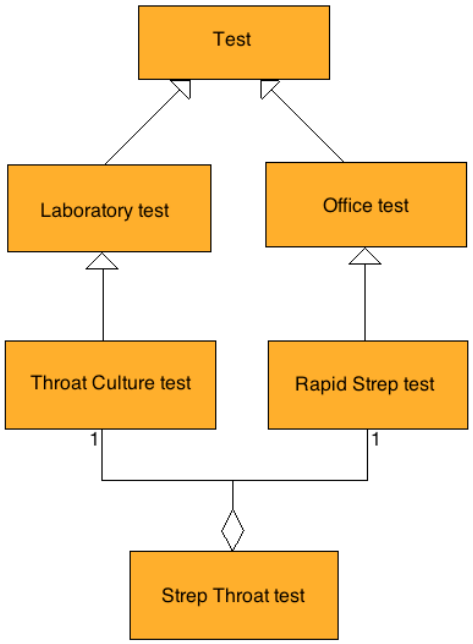

Figure 11. Class diagram of laboratory test processes

Given the relationships between workflow elements and the XML files containing actual process data, MSTART can now instantiate each process and construct a hierarchy of processes. Figure 11 depicts the class diagram of laboratory test processes described in Section 3.

After the workflow model file is read, our algorithm resolves "Generalization" (e.g. laboratory test inherits from simple linear test) and "Aggregation" (e.g. strep throat test contains a culture test) relationships shown in Figure 10 for test processes. This information is then used to determine the actual steps in each test. For example, although a laboratory test model definition does not own a "provider analyzes results" task (see Figure 4), because a laboratory test inherits from simple linear test and a simple linear test possesses this task after the "patient gets the test done" task (Figure 4), our algorithm determines that the "provider analyzes results" task should be the last task of a laboratory test and so on.

4.3 Data Objects

MSTART also reads two XML files that enclose all actors and organizations seen in Figure 12. For a laboratory test process, *organization* corresponds to a clinical facility and *actors* are medical workers. Critical to our simulation are the following facts: (i) organizations support some types of processes, and (ii) a test instance could be supported by multiple organizations. Actors supervise other actors so in case of a delay or error, supervisors can be notified.

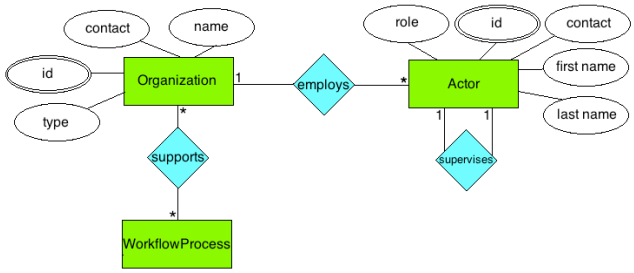

Figure 12. Entity-Relationship diagram of database Actor and Organization objects

Besides actor and organization objects, MSTART needs to retain events that happen at execution time. An *event* references an *order* and a *process* (Figure 13). Every event depicts an order's current snapshot. At creation time of an event, a time for the test result is predicted using the configuration file settings and the computations described in Section 3.3. Furthermore, *Order* class contains all the information about the lifecycle and status of a process (Figure 13). A *log* is created for every task completed during the actual processing of an *order*.

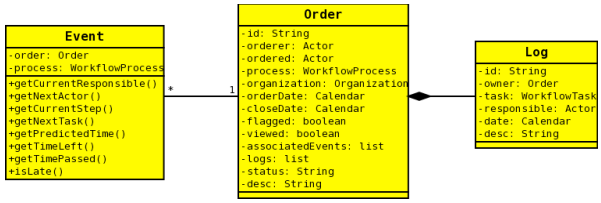

Figure 13. Relationships between events, orders, and logs

4.4 Interface Generator

Interface Generator is responsible from generating the user interface of MSTART (Figure 1) with the "Track" tab being selected. It creates three screens: (i) order, (ii) track, and (iii) retrospective analysis.

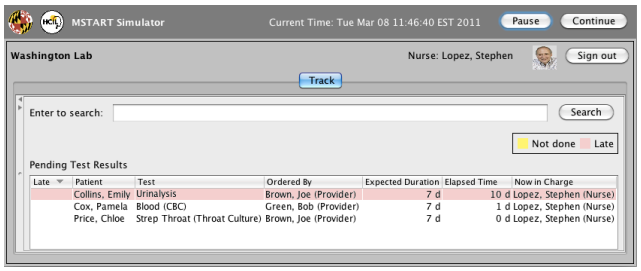

Figure 14. MSTART interface as seen by a nurse

Initially, actors are required to login and depending on the type of the actor, MSTART enables/disables certain screens. Figure 14 demonstrates how a logged in nurse would see MSTART interface. There is no ordering or retrospective analysis offered. Also, nurses only may access what is pending on their duties.

Figure 15. Pending strep throat test visualized in MSTART

The Interface Generator visualizes events described in the previous subsection. For instance, Figure 15 depicts two events as seen in MSTART. A strep throat test starts as one order, and then branches into two events. Each event is handled independently by different clinical staff, and finally, comes back to the ordering physician's office so that he or she can make decisions based either on individual reports or the overall result.

The user is guided with pop-up dialog boxes (such as Figure 16) to interact with these events. The interactive content of a dialog pane, called an *Actor Action Sheet*, is generated directly from the model definition, i.e. depending on what actions are specified in the XML file, actor action sheets are populated with widgets like checkboxes, dropdown boxes, buttons, and/or lists.

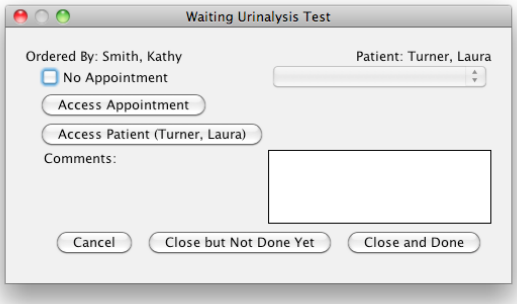

Figure 17. Nurse action sheet for registering a patient

Figure 16 introduces an "Act on Result of Strep Throat Test" dialog available to providers, with "Rapid Strep" tab currently chosen. Since a strep throat test comes as two separate results, different results are placed on separate tabs of the same dialog

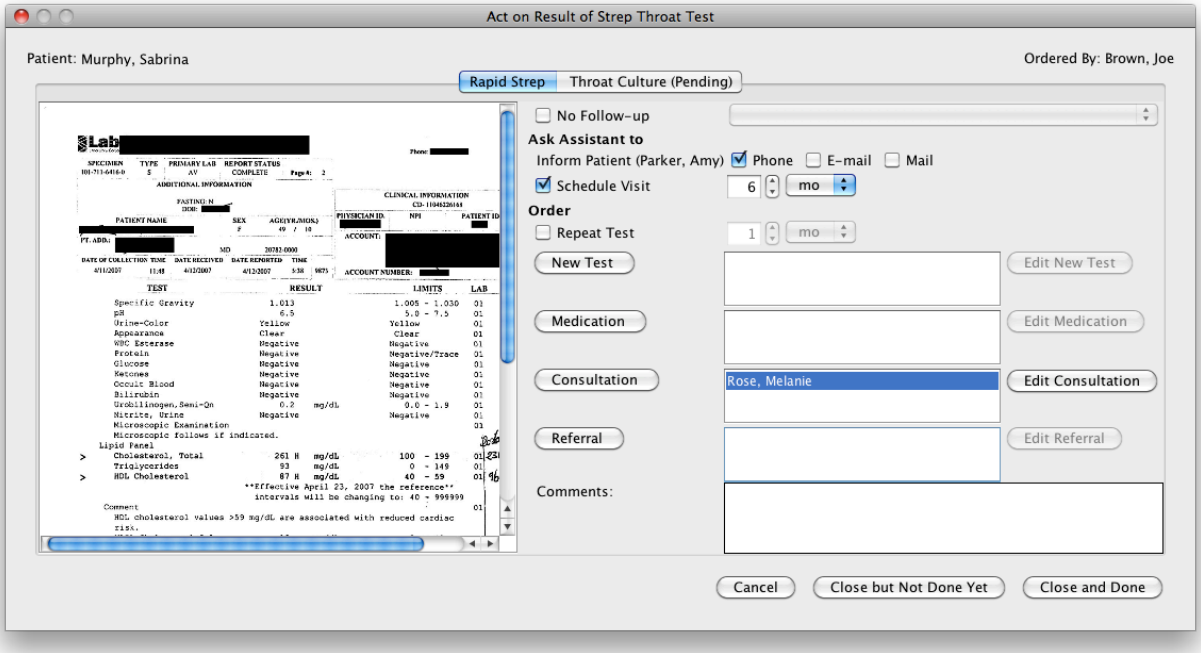

Figure 16. Dialog for provider to act on the result of a test. Results are presented (left of screen) with a menu of actor action sheet (right of screen). When the user double clicks on the rows in any of the tables in Track screen of MSTART, another window appears. Interacting with the main window of MSTART is disallowed until this window is closed. "Medication", "New Test", "Consultation", and "Referral" buttons take users to standard order screens implemented in EHR systems. Already taken actions are listed next to the corresponding buttons to remind the physician and allow them to edit or cancel those actions through standard screens. "Close and Done" finalizes follow-up so the result could be removed from "Arrived Results" table. On the other hand, "Close but Not Done Yet" button means the result was viewed and/or some actions may have been taken but the follow-up is not complete yet. Such a result is still kept in "Arrived Results" table for further processing but is marked as "viewed". "Cancel" button closes the window without affecting anything in cases when the clinician unintentionally opened an arrived result by mistake or wants it to remain as unviewed; the actions are not saved either.

frame. At this time, “Throat Culture” results have yet to come back so this is indicated with “Pending” flag on the tab text. XML definition in Figure 6 is used to layout the provider action sheet in Figure 16.

On the other hand, Figure 17 exemplifies possible actions of a nurse registering a patient at the lab. Since the result is not ready yet, there is no report. XML definition in Figure 8 is used directly to generate this screen. While the Cancel button allows the nurse to close the dialog without making any changes, “Close but Not Done Yet” button keeps the data in his/her track list, and “Close and Done” sends the test to a clinical staff person who will take care of the next task in the management of this lab test.

4.5 MSTART Simulator

To demonstrate MSTART in action, we simulate the placement of orders and actor actions. Our simulator advances time rapidly to show how the system would work in a real setting as orders become active. It accepts the desired number of orders and randomly creates orders at execution time. Logs related to that particular order occur at random during runtime as well. Because our example workflow model is from the medical domain, the orders created are for laboratory test processes.

5. DISCUSSION

Following a user-centered approach of iterative design and evaluation we conducted meetings with clinicians who provided feedback on our workflow model and MSTART. During a discussion with a medical doctor, a draft of improvements to the workflow model was prepared and the medical examples were elaborated. These yielded extended linear and parallel processes.

Approximately twenty clinicians provided feedback at three different events, for an estimated total of about five hours of review and discussions over the last three months. These were structured as either a demonstration of MSTART or a presentation to a group of people who are knowledgeable in medicine, human-computer interaction, or software engineering areas. All experts approved our workflow model and the calculation of expected durations, giving strong support to the explicit statement of responsible agents.

Most comments were related to the improvement to the model itself: (i) initially nurses in the clinic receive the results and if necessary, they are in charge of distributing results to physicians, (ii) an ordering physician can send the report to his resident or trainee within the clinic to check first, (iii) a clinic manager can check whether arrived results are acted upon by providers in a timely manner to ensure quality control. These all account for the routing within the clinic after the result arrived. By adding a couple of lines to the model and setting the preferences of the user interface, we accommodated these suggested improvements.

Another recommendation was regarding our process time predictions. One person recommended using artificial intelligence methods to learn from past results and adjust the times accordingly. We think that our current implementation is a starting approach and there is more work to be done. A more advanced configuration may consider factors for work hours (morning hours may get less work done compared to afternoon hours) or seasons (summer months may be slower because staff is on vacation). Different facilities may have different factors that influence how fast they return results, and this value might be adjusted over time to offer better predictions. Any attribute associated with an order could be used to adjust result times. Users can even adjust the factors through retrospective analysis.

There were many suggestions about our user interface generation. One reviewer encouraged us to improve the aesthetics of the interface. Terminology used on screens has been updated several times based on comments from medical experts.

Because MSTART is a domain-independent system, there are many other application domains that we can apply the idea of tracking multi-step processes. In academia, potential applications include monitoring the progress of undergraduate and graduate students in different departments, or reminding researchers of scientific publication acceptances. Business processes of enterprises can be tracked in a similar manner.

6. CONCLUSION

We described the design and implementation of MSTART, which uses a workflow model. The model specifies a hierarchy of process definitions for generating interfaces. MSTART produces a domain independent system that can be used to generate applications for multi-step processes. Our focus is on handling medical laboratory tests to reduce the currently dangerous number of missed laboratory reports. This paper also expanded our initial work by describing three approaches to improve laboratory test processes: (1) a refined workflow definition of agent temporal responsibilities to model more complex processes, (2) a strategy to generate actor action sheets that offer appropriate choices at each step, and (3) a configuration file mechanism to more accurately predict process result times.

Today, only a small number of comprehensive health providers can offer detailed tracking of laboratory test processes. Nevertheless, the simple linear test described in Section 3.1 can readily be implemented in medical clinics. We hope the description of our system will inspire system developers to introduce such tracking features in their own systems. We anticipate progressive laboratories would adopt such tracking methods to gain competitive advantages and improve their own performance. One of our goals is to encourage automated tracking implemented by methods such as barcodes and RFID tags to further reduce the clinical staff workload.

We continue gathering feedback from domain experts to refine our analysis of what is necessary. We will improve our process time prediction computations. We are also looking into design choices for a better retrospective analysis that could help the system administrator compare predicted with actual result times. Moreover, we want to work on the topic of actor responsibilities. Identifying who is responsible for a task in multi-step processes is a subject of discussion and requires a balance between individual accountability and the environment that the actors inhabit. Is performance the best, or only, way to tell whether someone is doing his or her job well? What are the preferred ways of identifying weak and strong actors in a multi-step process? Such issues also arise when one task needs to be collaboratively completed. What role do incentives play in helping to build awareness, and to analyze and improve performance of complex, multi-step processes? All of these questions warrant further study.

7. ACKNOWLEDGMENTS

This work was supported in part by Grant No. 10510592 for Patient-Centered Cognitive Support under the Strategic Health IT Advanced Research Projects Program (SHARP) from the Office of the National Coordinator for Health Information Technology. We also thank Dean Sittig, Daniel Murphy, Karen Guertler, Hardeep Singh, and Todd Johnson for their help with the paper.

8. REFERENCES

[1] Baig, N., Myers, R. E., Turner, B. J., Grana, J., Rothermel, T., Schlackman, N., and Weinberg, D. S. 2003. Physician-Reported Reasons for Limited Follow-Up of Patients with a Positive Fecal Occult Blood Test Screening Result. *Am. J. Gastroenterol.* 98, 9 (Sep. 2003), 2078-2081.

[2] Carraro P, Plebani M. 2007. Errors in a stat laboratory: types and frequencies 10 years later. *Clin. Chem.* 53, 7, 1338-42

[3] Christov, S., Chen, B., Avrunin, GS., Clarke, LA., Osterweil, LJ., Brown, D., Cassells, L., and Mertens, W. 2008. Formally Defining Medical Processes. *Methods Inf. Med.* 47, 5, 392-8.

[4] Cram, P., Rosenthal, G. E., Ohsfeldt, R., Wallace, R. B., Schlechte J., and Schiff, G.D. 2005. Failure to Recognize and Act on Abnormal Test Results: The Case of Screening Bone Densitometry. *Jt. Comm. J. Qual. Patient Saf.* 31, 2, 90-97.

[5] Dazzi, L., Fassino, C, Saracco, R, Quaglini, S, and Stefanelli, M. A. 1997. Patient Workflow Management System Built on Guidelines. In *Proc. AMIA Annu. Fall Symp.* 146-150.

[6] Dong, H., Greenest, D., Fleishert, G., and Kohane, I. 1998. An alert system for ED lab test results. *Proc AMIA Symp* 994

[7] Elder, N. C., McEwen, T. R., Flach, J. M., Gallimore, J. J., and Pallerla, H. 2010. The Management of Test Results in Primary Care: Does an Electronic Medical Record Make a Difference? Fam. Med. 42, 5 (May 2010), 327-333.

[8] Fox, J., Johns, N., Rahmanzadeh, A., Thomson R. 1996. PROforma: A Method and Language for Specifying Clinical Guidelines and Protocols. In *Proc. Med. Inform. Europe*.

[9] Hickner, J. M., Fernald, D. H., Harris, D. M., Poon, E. G., Elder, N. C., and Mold, J. W. 2005. Issues and Initiatives in the Testing Process in Primary Care Physician Offices. *Jt. Comm. J. Qual. Patient Saf.* 31, 2 (Feb. 2005), 81-89.

[10] Hripcsak, G., Clayton, P. D., Jenders, R. A., Cimino, J. J., and Johnson, S. B. 1996. Design of a Clinical Event Monitor. *Comput. Biomed. Res.* 29, 3 (Jun. 1996), 194-221.

[11] Hsiao, C. J., Cherry, D. K., Beatty, P. C., and Rechtsteiner, E. A. 2010. National Ambulatory Medical Care Survey: 2007 Summary. Natl. Health Stat. Report. 28 (Sep. 2010), 1-32.

[12] Krall, M. A. and Sittig, D. F. 2001. Subjective Assessment of Usefulness and Appropriate Presentation Mode of Alerts and reminders in the outpatient Setting. *Proc. AMIA Symp.* 334-8.

[13] Kuperman GJ, Teich JM, Bates DW, Hiltz FL, Hurley JM, Lee RY, and Paterno, MD. 1996. Detecting Alerts, Notifying the Physician, and Offering Action Items: A Comprehensive Alerting System. In *Proc. AMIA Annu. Fall Symp.* 704-708.

[14] Marcus AC, Kaplan CP, Crane LA, Berek JS, Bernstein G, Gunning JE, McClatchey MW. 1998. Reducing loss-to-follow-up among Women with abnormal Pap smears: Results from a Randomized Trial Testing an Intensive Follow-up protocol and economic incentives. *Med. Care.* 36, 3, 397-410

[15] Mold, J. W., Cacy, D. S., and Dalbir, D. K. 2000. Management of Laboratory Test Results in Family Practice: An OKPRN Study. *J. Fam. Pract.* 49, 8 (Aug. 2000), 716-7.

[16] Noumeir, R. 2006. Radiology Interpretation Process Modeling. *J. Biomed. Inform.* 39, 2 (Apr. 2006), 103-114.

[17] Oosterhuis, WP, Ulenkate, HJ, and Goldschmidt, HM. 2000. Evaluation of LabRespond, new automated validation system for clinical laboratory test results. *Clin. Chem.* 46, 11, 1811-7

[18] Oppenheim, M. I., Mintz, R. J., Boyer, A. G., and Frayer, W. W. 2000. Design of a Clinical Alert System to Facilitate Development, Testing, Maintenance, and User-Specific Notification. In *Proc. AMIA Symp.* 630-634.

[19] Payne TH, Savarino J. 1998. Development of a clinical Event Monitor for Use with the VA Computerized Patient Record System and Other Data Sources. In *Proc. AMIA Symp.* 145-9.

[20] Peleg M, Boxwala AA, Ogunyemi O, Zeng Q, Tu S, Lacson R, Bernstam E, Ash N, Mork P, Ohno-Machado L, Shortliffe, EH, Greenes RA. 2000. GLIF3: The Evolution of a guideline representation format. In *Proc AMIA Symp.* 645-9

[21] Phillips RL, Bartholomew LA, Dovey SM, Fryer GE, Miyoshi TJ, and Green LA. 2004. Learning from Malpractice Claims about Negligent, Adverse Events in Primary Care in the United States. *Qual. Saf. Health Care.* 13, 2, 121-126.

[22] Poon, E. G., Haas, J. S., Puopolo, A. L., Gandhi, T. K., Burdick, E., Bates, D.W., and Brennan, T. A. 2004. Communication Factors in the Follow-up of Abnormal Mammograms. *J. Gen. Intern Med.* 19, 4, 316-323.

[23] Poon, E. G., Kuperman, G. J., Fiskio, J., and Bates, D. W. 2002. Real-time Notification of Laboratory Data Requested by Users through Alphanumeric Pagers. *JAMIA* 9, 3, 217-22.

[24] Roy, C. L., Poon, E. G., Karson, A. S., Ladak-Merchant, Z., Johnson, R. E., Maviglia, S. M., Gandhi, T. K. 2005. Patient Safety Concerns Arising from Test Results that Return after Hospital Discharge. *Ann. Intern Med.* 143, 2, 121-128.

[25] Schiff, G. D., Kim, S., Krosnjar, N., Wisniewski, M. F., Bult, J., Fogelfeld, L., and McNutt, R. A. 2005. Missed Hypothyroidism Diagnosis Uncovered by Linking Laboratory and Pharmacy data. *Arch. Intern Med.* 165, 574-7

[26] Shabot, M. M., LoBue, M., and Chen, J. 2000. Wireless Clinical Alerts for Physiologic, Laboratory and Medication Data. In *Proc. AMIA Symp.* 789-793.

[27] Shahar Y, Miksch S, Johnson P. 1998. The asgaard project: a task-specific framework for application, critiquing of time-oriented clinical guidelines. *Artif. Intell. Med.* 14, 1-2, 29-51.

[28] Singh, H., Thomas, E. J., Mani, S., Sittig, D., Arora, H., Espadas, D., Khan, M. M., Petersen, L. A. 2009. Timely Follow-up of Abnormal Diagnostic Imaging Test Results in an Outpatient Setting: Are Electronic Medical Records achieving their potential? *Arch. Intern Med.* 169, 17, 1578-86

[29] Smith, B. J. and McNeely, M. D. 1999. The Influence of an Expert System for Test Ordering and Interpretation on Laboratory Investigations. *Clin. Chem.* 45, 8, 1168-1175.

[30] Sung S., Forman-Hoffman V., Wilson M. C., Cram P. 2006. Direct reporting of laboratory test results to patients by mail to enhance patient safety. *J. Gen Intern Med.* 21, 10, 1075-78

[31] Tarkan, S., Plaisant, C., Shneiderman, B., and Hettinger, A. Z. 2011. Reducing Missed Laboratory Results: Defining Temporal Responsibility, Generating User Interfaces for Test Process Tracking, and Retrospective Analyses to Identify Problems. To appear in *Proc. AMIA Symp.* In Press.

[32] Wagner MM, Pankaskie M, Hogan W, Tsui FC, Eisenstadt, SA, Rodriguez, E, Vries JK. 1997. Clinical Event Monitoring at the University of Pittsburgh. In *Proc. AMIA Symp.* 188-92.

[33] Wahls T, Haugen T, Cram P. 2007. The Continuing Problem of Missed Test Results in an integrated health system with an advanced EMR. *Jt Comm J Qual Patient Saf.* 33, 8, 485-492.

9. APPENDIX

The preferred review approach:

- The systems track
- The preferred allocation of reviewing expertise: the primary focus of the paper is Computer Science, and the secondary focus is Medicine.
- Three topics covered in the paper:
    - Health information system engineering: Health software architecture, framework, design, and engineering
    - Information technologies for healthcare delivery and management: Healthcare workflow management
    - Health information systems: Applications in electronic health records