Supporting State-wide Immunization Tracking using Multi-Paradigm Workflow Technology

Amit Sheth, Krys Kochut, John Miller
Devashish Worah, Souvik Das, Chenye Lin
Large Scale Distributed Information Systems Lab
The University of Georgia
Athens, GA 30602-7404
email: *amit@cs.uga.edu*
URL: *http://LSDIS.cs.uga.edu*

Devanand Palaniswami, John Lynch
Ivan Shevchenko
Connecticut Healthcare Research
and Education Foundation
Wallingford, CT 06492
email: *lynch@chime.org*
URL: *http://www.chime.org*

Abstract

The rapidly evolving managed health-care industry requires efficient coordination of human and automated tasks as well as information-flow across multiple enterprises. One of the most critical applications in managed care is state-wide immunization tracking, which if supported by appropriate workflow technology can achieve substantial near term impact. In this paper, we discuss a *comprehensive* and *real-world* application to support child immunization tracking for the state of Connecticut in close collaboration with CHREF[1]. The application system uses UGA-LSDIS's[2] multi-paradigm transactional workflow management system METEOR_2. It utilizes the World Wide Web either exclusively, or in conjunction with CORBA-based infrastructures.

Proceedings of the 22nd VLDB Conference
Mumbai(Bombay), India, 1996

[1] Connecticut Healthcare Research and Education Foundation

[2] Large Scale Distributed Information Systems Lab., The University of Georgia

1 Introduction

Workflow management systems (WFMS) have been used to automate organizational processes in various application domains. However, most of the existing WFMS are limited in terms of their applicability across heterogeneous software and hardware platforms, or in terms of their support for coordinating processes across multiple organizations, or in their ability to deal with large-scale real-world applications. Among approximately 250 products claiming or aiming to support workflow management, a large majority are primarily relevant to office automation (including e-mail routing and imaging/document/management applications) that usually involve tasks (also termed steps or activities) performed by humans.

The METEOR_2 effort[3] at UGA-LSDIS builds upon the earlier experiences involving prototype WFMS and telecommunication application development, in the METEOR project at Bellcore [ANRS92, ASSR93, JNRS93, KS95]. While METEOR_2 shares many of the principles and the gross architecture approach taken in METEOR, it goes substantially beyond it in research and technology development for the WFMS, and more importantly, in application and technology transfer aspects.

[3] This research was partially done under a cooperative agreement between the National Institute of Standards and Technology Advanced Technology Program (under the HIIT contract, number 70NANB5H1011) and the Healthcare Open Systems and Trials, Inc. consortium. See URL:http://www.scra.org/hiit.html. Additional partial support and donations are provided by Post Modern Computing, Illustra Information Technology, and Hewlett-Packard Labs.

An important aspect of this project is that the technology and system development effort at UGA-LSDIS has occurred in close collaboration with CHREF. The collaboration involves a detailed study of applications that can benefit from workflow technology with the help of healthcare industry experts at CHREF and end-users at healthcare providers. It also involves application development utilizing schemas of real (production) databases on heterogeneous computing environments at multiple locations. These aspects are reported in this paper.

The application is designed and implemented using the prototype METEOR$_2$ WFMS that supports, what we call, *multi-paradigm transactional* workflows. The "multi-paradigm" aspect discussed in this paper refers to the support for intra- and inter-enterprise workflows over a variety of distributed and heterogeneous processing infrastructures. In particular, METEOR$_2$ supports workflow automation with centralized and distributed architectures utilizing the Web and CORBA. The "multi-paradigm" also refers to support for different notions of transactions as found in database management, distributed transaction monitors, as well as in Electronic Data Interchange (EDI), and in specific application domains (e.g., HL7 in healthcare) [WS96].

The rest of this paper is organized as follows. Section 2 defines the immunization tracking application. This is followed by a discussion of related work. Section 4 reviews the workflow model supported by METEOR$_2$. Sections 5 and 6 discuss two WFMS prototype implementations. All aspects of the system discussed in these sections have been implemented. Finally, in section 7 we discuss some of our experiences in developing this application. This paper is a substantially abridged version of the report available at http://lsdis.cs.uga.edu/publications.

2 Application Description and Requirements

With managed healthcare coming of age, monitoring and tracking the performance of the different players involved, compulsory performance reporting, immunization tracking, child birth reporting, etc. have become important. In fact, the first item listed under Quality of Care in the Health Plan Employer Data and Information Set (HEDIS) [Ass95] is Childhood Immunization Rate. Healthcare resources have to be used efficiently to lower costs while improving the quality of care provided and processes in the managed healthcare industry need to be computerized and automated.

Figure 1 shows a schematic and the scope of the

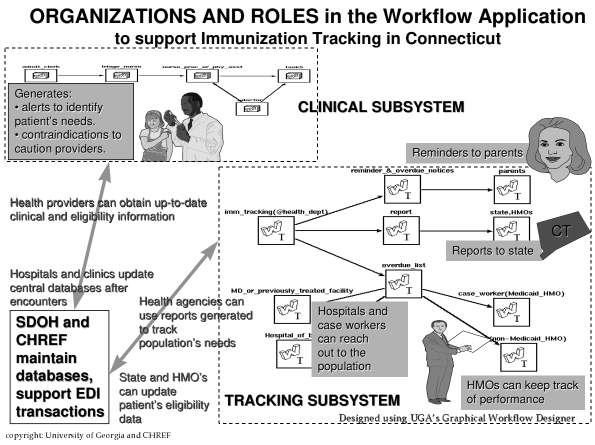

Figure 1: Application Schematic

application we have developed. This includes online interactions for the workflow application between CHREF (as the central location), healthcare providers (Hospitals, Clinics, Home Healthcare Providers) and user organizations (SDOH, Schools, Department of Social Services-DSS).

The system can be best explained in terms of the following three components.

Databases

The workflow application system supports the maintenance of the following central databases:

- The Master Patient Index (MPI) to record the personal information and medical history of patients,
- The Master Encounter Index (MEI) to record brief information pertaining to each encounter of patients at any hospital or clinic in the state,
- An Immunization Database (IMM) to record information regarding each immunization performed for a person in the MPI,
- Eligibility Databases (ELG) that provides an insurance company's patient eligibility data, and
- Detailed Encounter Databases (ENC) that provide detailed encounter information as well as relevant data for patients served by that provider.

The current application design and implementation calls for CHREF to manage the MPI, MEI, and IMM centralized databases for the state of Connecticut. The ELG database containing data provided by insurance companies is used in this application for EDI based eligibility verification using the ANSI X12 standard. In particular, Web-based access is used to submit eligibility inquiries using ANSI 270, and responses are received using ANSI 271 "transactions". Each participating provider organization (hospital or clinic) has its own ENC database. When a structured database is used at CHREF, we promote the use of ODBC compliant DBMSs. The DBMSs (Illustra and Oracle) used in the application are ODBC compliant.

The Clinical Subsystem

The clinical subsystem has been designed to provide the following features:

- roles for Admit Clerk, Triage Nurse, Nurse Practitioner and Doctor,
- worklists for streamlining hospital and clinic operations,
- automatic generation of Medical Alerts (e.g., delinquent immunizations) and Insurance Eligibility Verification by the Admit Clerk, and
- generation of contraindications for patients visiting a hospital or clinic to caution medical personnel regarding procedures that may be performed on the patient.

The Tracking Subsystem

Health agencies can use the data available to generate reports (for submission to authorities like the DSS, State Government, etc.), and for determining the health needs of the state. More importantly, immunization tracking involves reminding parents and guardians about shots that are due or overdue and informing field workers about children who have not been receiving their immunizations.

2.1 Requirements of the Application

Some of the important requirements for this application, as determined by CHREF, include:

- Support for a distributed client/server based architecture in a heterogeneous computing environment. At the level of any user of the system, this distribution should be transparent.
- Support for inter- and intra-enterprise wide coordination of tasks.
- Provision of a standard user-friendly interface to all users of the system.
- Support for a variety of tasks: transactional and non-transactional, user and application.
- Capability of using existing DBMS infrastructure across organizations.
- Low cost of system for the providers and user organizations.
- Ease of modification (re-design), scalability, extensibility and fast design-to-implementation.
- Use of standards, including EDI for interactions between autonomous organizations where possible.
- Security authorization for users and secure communication (required as patient data is typically confidential).

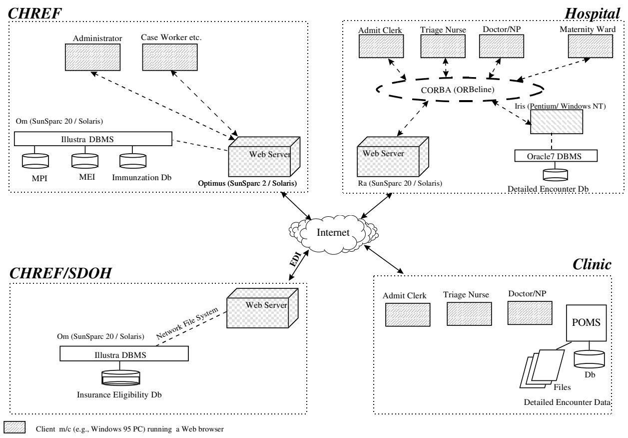

Figure 2: System Schematic

Based on these requirements, we have created a system testbed on which the application is implemented (See Figure 2). This includes heterogeneous server systems (Solaris and NT) spread across CHREF and LSDIS-UGA, use of heterogeneous communication infrastructure (**multiple** Web servers, CORBA), and multiple databases.

The development of the application was done keeping the end-user and system requirements in mind. The Web-based user-interfaces were designed based on the needs of healthcare personnel who would eventually be the end-users.

3 Related Work

A lot of general literature is available on workflow research [Elm95, Dog96, She95]. In this section we review only the work that is relevant to the application being discussed.

The Web has been used in other workflow related work. It has been used in workflows dealing with information processes, such as an office-wide calendar system, and in business processes such as a library book ordering system [WF94]. A Virtual Electronic Medical Record [VEM95] was developed "to provide clinicians at the Virginia Neurological Institute with a means of accessing complete, up-to-the-minute patient information on-line".

A hospital care planning system, using a Web interface and the Oz Collaborative Workflow Environment, has been implemented to deal with such things as patient admissions, assigning staff for patients, and planning patient care events [Lee95]. ActionWorkflow Metro [Tec95] allows a standard Web browser to become a client for their WFMS. All these products have a major disadvantage of using a single server with a primary forms-based interface. Typically, all relevant data are kept on a single central server. This approach does not scale up to large and widely distributed environments with autonomous organizations. Fault tolerance issues are largely ignored.

A lot of research has been done using distributed middleware technology like CORBA, OLE, OpenDoc, or DCE. These infrastructures provide security to varying degrees and are in general better in failure recovery and exception handling than current Web-based technology alone. Digital's ObjectBroker is one such product in which multiple applications work in a seamless manner by interoperating and exchanging data. The Mentor project [WWWD95] uses a rigorous workflow specification method and tries to reconcile it with a distributed middleware architecture as a step towards enterprise-wide integration. GTE's TSME project has aimed at integrating object-oriented programming, distributed databases, and operating system technologies, to provide general mechanisms for efficient interoperability among heterogeneous computing resources [GHKM94].

The Exotica project [MAGK95] aims to enhance the FlowMark product by addressing scalability and availability issues. The ABS architecture proposed in [SJKB94] introduces an implementation architecture for a WFMS that addresses issues like transparency of underlying computing infrastructure and scalability with respect to heterogeneous and distributed computing environments. [ANRS92] discusses in detail the design and implementation of a prototype system that supports the execution of Flexible Transactions and its use to develop a service order provisioning application. Using the term *transactional workflows*, use of transaction concepts in workflow management were introduced in [SR93] and subsequently discussed in several papers including [BDSS93, GH94, RS95, GHS95, MAGK95, TV95].

4 Workflow Management System METEOR$_2$

The METEOR$_2$ model is an extension of the METEOR model [KS95] in terms of both the logical and run-time models.

The logical workflow model has been enhanced in terms of defining additional task structures for tasks involving database 2PC and 2PC coordinator tasks for adding transactional features to groups of tasks [Wan95], worklist manager tasks for managing worklists of humans interacting with the system [Mur95], and the use of *roles* for mapping the organizational structure of the real-world.

The run-time issues that have been addressed in METEOR$_2$ relate to the use of the Graphical Workflow Designer for defining the workflow process [Lin96], automatic translation from the design to run-time code, implementation of different scheduling paradigms [Wan95, Das96, Pal96], support for exception handling and failure recovery [Wor96], and integration of heterogeneous communication infrastructures and processing environments (such as CORBA, Web, and Lotus Notes) into the run-time model.

4.1 The Meteor$_2$ Run-Time Model

The main components in the execution environment are the Workflow Scheduler, Task Managers and Tasks (see Figure 3). To establish global control, as well as facilitate recovery and monitoring, the task managers communicate with a scheduler. It is possible for the scheduler to be either centralized or distributed, or even some hybrid between the two [Wan95, MSKW96].

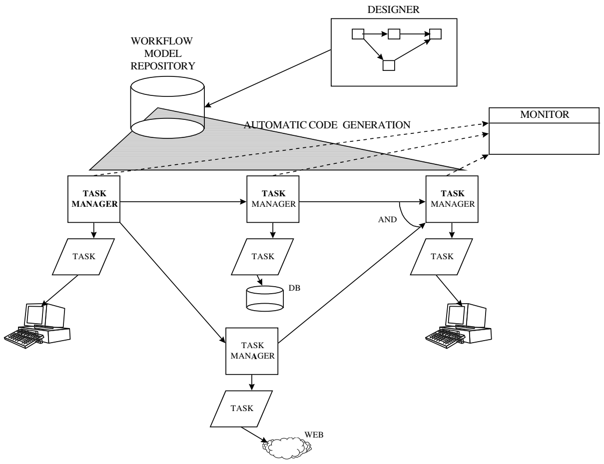

Figure 3: Meteor$_2$ Run-Time Model

To facilitate monitoring and control of tasks, each task is modeled using a well-defined task structure at the design time [ASSR93, RS95, KS95]. A task structure indicates the generic form of a task. A structure simply identifies a set of states and the permissible transitions between those states. Several different task structures have been developed - transactional, non-transactional, compound, twophase commit (2PC) [KS95, Wan95].

To more effectively manage complexity, workflows may be composed hierarchically. The top level is the overall workflow (see Figure 4), below this are sub-workflows (or compound tasks), which themselves may be made up of subworkflows or simple tasks. Coordination between tasks is accomplished by specifying intertask dependencies using enable clauses. An enable clause may have any number of predicates. Each predicate is either a task-state vector or a boolean expression. When one task leaves a given state it may enable another task to enter its next state.

4.2 Graphical Workflow Designer and Automatic Translation

A number of commercial workflow management tools provide adequate to good workflow designers, as far as graphics is concerned, and our tool does not claim any distinction in this respect. However, it does support a more comprehensive workflow model as described above. The designer has three components: the map designer, the data designer, and the task designer. These components can model (in the same order):

- the workflow map expressing the ordering of tasks and dependencies among them,
- the data objects manipulated and transmitted by the tasks, and
- the structure of the tasks.

Figure 4 shows a screen snapshot of the map designer with part of the design of the Immunization Tracking application.

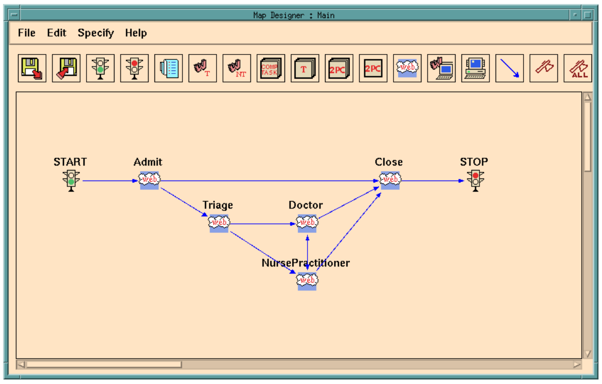

Figure 4: Main Workflow Map

The user starts by creating a *workflow map*, incorporating a number of *tasks* (both simple and compound). The sub-workflow abstraction enables the user to design the overall workflow at a high level (in particular, the whole workflow can be regarded as a single compound task). The map of the Admit Clerk is presented in Figure 5. The Immunization Tracking application contains several such compound tasks.

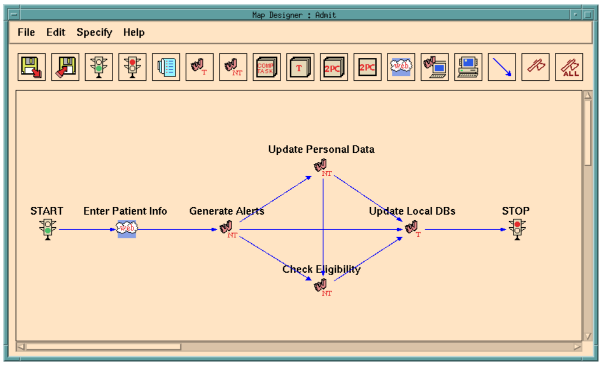

Figure 5: Admit Sub-Workflow Map

A typical workflow incorporates a number of data types which represent objects available to the workflow tasks. We have decided to utilize an object-oriented data model (Object Modeling Technique [RBP+91]) as the basis for the data designer.

The designer of a workflow can create a number of classes, encapsulating both the data representation (attributes) and operations on the data (methods) for various data elements manipulated by the tasks in the workflow. As in OMT, various relationships (binary, ternary, one-to-one, one-to-many, etc.) may be defined on groups of existing classes to reflect data type dependencies. In addition, similar classes can be grouped together into hierarchies.

A workflow design is output in an intermediate form, Workflow Intermediate Language (WIL). A WIL specification file is converted into code stubs for the desired runtime architecture as part of the automatic translation process. The runtime code stubs can be used to easily incorporate hand-written task code or even existing applications.

4.3 The Role of World-Wide Web in METEOR$_2$

Two principal communication infrastructures are used to build Meteor$_2$ prototype implementations. These infrastructures were carefully chosen for their functionality, availability, popularity and reasonable cost structures. We believe that our choices of Web and CORBA were fundamentally right. Web technology is needed in two distinct and important ways. First, the ubiquitous nature of Web browsers such as Netscape's Navigator make it a natural user interface. One of the application requirements was that users with any of the popular platforms should be able to participate in a workflow without additional new hardware. A multitude of users (many of whom are not computer sophisticated) are already familiar with these browsers' easy-to-use interface (see Figure 6). They presently see the interface as a way to access all sorts of information and perform simple tasks such as filling out forms. The uniformity, wide availability and simplicity of the interface makes it ideal for the healthcare domain where there is little time or inclination for special-purpose training.

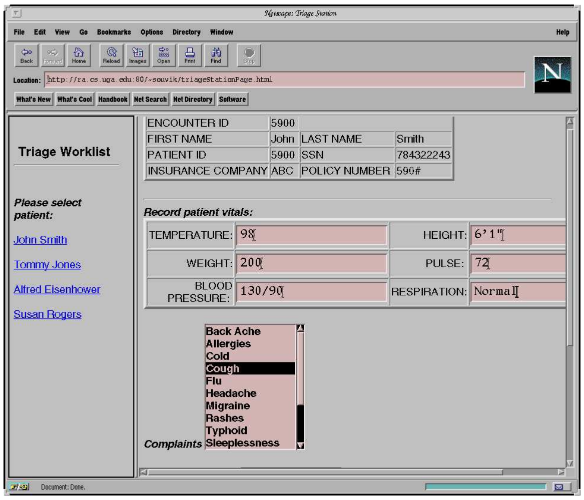

Figure 6: Triage Nurse's Interface

Another important role for the Web is in providing a communication infrastructure that supplements and/or replaces CORBA in that role. We view CORBA as a key element for building robust transactional workflow systems (see the next section for more details). However, we believe that it is unlikely for all organizations participating in a workflow (e.g., clinics and small hospitals) to be able to afford a CORBA product or, more importantly, to have the trained personell to manage this relatively complex system. It is more likely that they will have a Web server or access to a Web server (e.g., at CHREF) that they can use from their Web browser. For this reason, METEOR$_2$ can support workflows in a Web-only (CORBA-free) mode. It relies solely on Web browsers, Web servers, the HyperText Transfer Protocol (HTTP) and Common Gateway Interface (CGI) scripts coded in C/C++ and Perl to handle all of the communications. While HTML files are used on the server side to provide static documents on the Web, CGI programs allow data entry into forms and data manipulation, including database accesses, thus allowing multiple persons and organizations to interact easily over the Web. This capability is used in developing distributed Web-based workflows.

Any number of organizations can be easily incorporated into a workflow - a Web server at an organization accesses its local database(s) using CGI programs that can be accessed using Web browsers at other organizations - thus facilitating a truly distributed client-server implementation. Multiple Web servers are needed to integrate the various autonomous organizations (CHREF, hospitals, clinics, and insurance companies) and databases (MPI,MEI, Immunizations, Insurance/Eligibility, Detailed Encounter) into the workflow process and to allow for task distribution across these organizations.

Most requirements of the workflow application system (mentioned in the previous section) can be met using a Web-based implementation. The Web-only approach has its limitations in terms of workflow scheduling and data transfer and requires higher user involvement in the workflow process. Many of these limitations have been addressed in the integrated (CORBA and Web) version of our application system.

4.4 The Role of CORBA in METEOR$_2$

Following the advocacy for distributed object management based infrastructure for workflow management systems in [GHS95], we extensively utilize the specific services provided by PostModern Computing's ORBeline 2.0 [Inc94], a CORBA 2.0 implementation.

We view CORBA as fundamentally important for building a robust WFMS capable of supporting transactional workflows. CORBA provides an infrastructure for facilitating the development of reusable, portable, robust and interoperable object-based software in a distributed and heterogeneous environment. In addition, CORBA provides an infrastructure that addresses many of the limitations that are present in the communication model of the Web.

In the Web-based paradigm, passing of information between tasks (in this case CGI scripts) is primarily limited to URL-encodings and hidden fields. On the other hand, CORBA allows object types to be specified so that any type of information can be straightforwardly passed and automatically checked for compliance with the type. In addition, typed objects can be passed between tasks simply by reference, thereby requiring only necessary changes at the task level (leaving the IDL-interface unchanged) if there is a change in data type of the objects exchanged.

Error handling is a critical requirement for WFMSs. CORBA provides an `Exception` base class to be used to handle *standard exceptions* and *user exceptions* [Inc94]. This makes it possible to define error handlers at the interface level to *catch* exceptions that might occur at various levels of interaction (e.g., task and processing entity, task manager and task, scheduler and task manager) within the workflow process. The Web model does not provide specific constructs for error handling. To be able to deal with errors that might occur at run-time (e.g., due to relocation or unavailability of documents on a Web server, communication failures, and server failures), additional knowledge (in the form of routing tables, alternate routing mechanism, etc.) would have to be coded into the CGI scripts themselves, or would require user interaction through the browser to decide an alternate course of action.

One of the key requirements for developing robust transactional workflows is fault tolerance and task-level recovery. Exception handling and workflow recovery is modeled in a hierarchical manner at three levels of the METEOR_2 model: the task, the task manager, and the workflow level. The support for these features is being incorporated into the existing workflow designer and the distributed run-time architecture. CORBA provides a fault tolerant environment to guarantee availability of object implementations throughout the lifetime of an object. This inherent fault-tolerance of CORBA makes it superior to the Web for designing robust distributed applications.

5 CORBA-based Implementation of the Workflow System

We have designed and implemented several CORBA-based WFMS [MSKW96]. Here we discuss the fully distributed implementation that is used for the application under consideration.

5.1 Distributed Scheduler Architecture

The idea behind the creation of the fully distributed runtime for a WFMS is simple. The main principle is that every task manager should participate equally in the scheduling duties. Therefore, there is no need for a single process (possibly even including a number of threads) to carry out all of the task and data maintenance. All of the communication (including task manager/task activation and data object management and transfer) is handled by the CORBA infrastructure. We utilized Post Modern Computing's ORBeline 2.0 product [Inc94], implementing CORBA 2.0 standard.

As seen in Figure 7, the layout of the workflow task managers resembles very closely the workflow as it was created by the map designer of our graphical designer (shown in the Appendix).

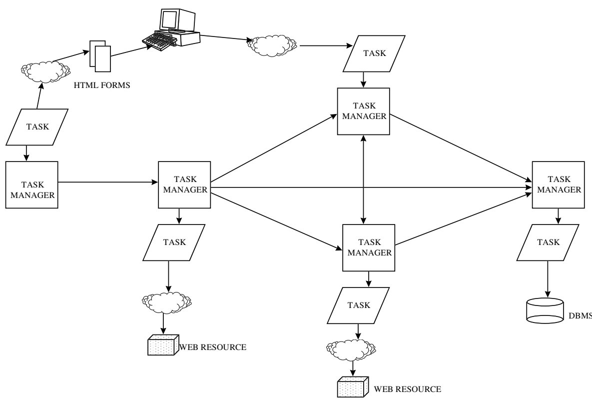

Figure 7: Distributed Architecture

In the fully distributed version of our runtime system, the task scheduling as well as handling of data for task inputs and outputs is controlled locally by individual task managers. Unlike in the centralized versions of the runtime, no single process is responsible for controlling the overall workflow. Instead, each of the task managers comprising the whole workflow is responsible for activating its own dependent (successor) task manager(s). Each task manager contains scheduling code required to spawn off the next task manager(s) as specified by the workflow designer. This scheduling code is automatically generated and included as a part of the run-time code for the task. In addition, each task manager must pass the necessary data objects as inputs to the dependent task, most likely after updating or creating the data. Communication between task managers is achieved using CORBA Interface Definition Language (IDL) interfaces.

Each task manager is equipped with a fragment of per task manager scheduling code which is subdivided into pre-activation, task activation, and post-activation parts. The pre-activation part must determine if and when the associated task should start execution. Since a task in a workflow may depend on a number of other tasks (with the control part either AND-ed or OR-ed), such an AND-OR tree is normalized to a disjunction of conjunctions and compiled into a fragment of C++ code, essentially acting as part of the scheduler of the overall workflow.

The task activation part handles the execution of the associated task. After preparing the task inputs, which might involve "unpacking" of the input data objects, and verifying that the task program may be executed by the specified processing entity, the task itself is started. After its completion, the final state of the workflow task is determined, and (possibly depending on the produced results) the output data objects are prepared.

In the post activation part, which corresponds to the workflow task being in one of its terminating states, the task manager's scheduling part attempts to initiate processing by the task manager of the dependent task.

The monitoring service is used to oversee the workflow execution. Individual task managers communicate their internal observable states, as well as data object references to it by asynchronous messages.

The distributed architecture matches the inherent distributed character of the workflow very well. Also, it eliminates the bottleneck of task managers having to communicate with a remote centralized scheduler during the execution of the workflow. Another advantage of this architecture is that it avoids single point of failure (we have not discussed details of the recovery managers due to limitations of space).

5.2 Implementation of the Distributed Architecture

Task managers in our distributed architecture based on CORBA are implemented as CORBA objects. In addition, data elements manipulated by tasks in the workflow are also represented as CORBA objects; task managers transfer data simply by exchanging references to them.

Human tasks (implemented as HTML forms) interact with the CORBA-based workflow through CGI scripts (associated with the HTML forms). Such a script is implemented as a CORBA client of the controlling task manager object (regarded as the server). The script, after accessing the user-provided data, calls one of the task manager methods to transfer the information to the task manager. The task manager can then activate dependent task manager(s) and transfer the data via the ORB.

Synchronization of the human-oriented flow of control (based on form-to-form thread) and the ORB-based flow of control and data between task managers is quite involved. For example, consider a scenario in which a user interacting with the workflow system (via a Web browser) needs to perform two tasks with a few application tasks (scheduled by the user to retrieve some data objects) in between. Control and data flow between the application tasks is managed by the ORB, while the transition from one user task to the next is controlled by the user and implemented as an HTML form thread. The synchronization is achieved by specifying two incoming transitions to the task manager at the synchronization point, one coming from the previous task manager and one from the CGI script activated from an HTML form. These two transitions are connected by an AND operator and thus either one of them must wait for the other before the task manager can activate its associated task.

6 Web-Based Implementation of the Workflow System

In this implementation, tasks are implemented as CGI scripts coded in C/C++ or Perl. They are responsible for controlling access to patient data in their respective organizations. For example, the Web server at CHREF provides data contained in the MPI, MEI and Immunization databases to various tasks involved in the workflow application. The tasks associated with the admit clerks (at a hospital) are implemented as CGI scripts running on a Web server at the hospital and are responsible for retrieving patient-specific information (demographic data, encounter data, and immunization alerts) from the MPI, MEI, and Immunization databases.

Output data and state information is transferred between tasks using `hidden` fields within HTML documents, and Universal Resource Locator (URL) encodings.

Control and data information is communicated between the tasks through HTML documents. The output (standard output) of a task is written as an HTML page (typically containing a form). Workflow execution is coordinated by using inlined task scheduling information in the HTML form using the `action` and `method` attributes of the `FORM` tag. This page includes a submit button that, once clicked, will cause the next CGI script (task), denoted by the URL in the value of the `action` tag to execute. Hence, control flow between tasks is indirect (instigated by the user).

Following the approach described above, tasks interfaced to a common browser may be executed one after the other. After a few such steps, the workflow will need to move on to the next participant (user). This is accomplished using a worklist mechanism as described below. The final task of the first user (e.g., an admit clerk) simply writes a work entry into a database (or specially protected file). Using the client-pull method, a worklist task for the next user (e.g., a nurse) periodically polls the database and rewrites the worklist window (or frame) based on the currently available work entries. An entire workflow is started by running the initial CGI script.

In some cases it is desirable to allow two CGI scripts to communicate more directly by opening up a socket connection (following the http protocol) from the first CGI script to a Web server which then invokes the second CGI script. Parameters are passed through the socket connection. The intermediate step of using the button on a form to initiate the second CGI script is bypassed. This bypass is particularly useful for accessing databases (local or remote) when less human interaction is desired or when it is necessary to connect to a database inside a firewall.

Although not as fully distributed as the distributed CORBA architecture, the Web-based implementation is nevertheless distributed in the sense that multiple clients (browsers), servers and DBMSs are involved. Consider for example the admit clerk. The first CGI script which is run to display the initial form is likely to be run from the local (hospital) Web server. The next CGI script would be run at CHREF to enable access to the MPI, MEI and Immunization databases available at CHREF. This would in-turn generate patient specific information (demographic, identification, medical history, medical alerts). This information presented as output on a HTML form would then be used as an input to the task accessing the Insurance/Eligibility databases accessed through the Web server at the Insurance company using EDI across the Internet. Eligibility information returned from the previous task is used as an input for the next task (triage nurse task) in the workflow process. On completion of the admit clerk task, it adds the patient's identifier to the worklist of the triage nurse. Coordination of the tasks in the workflow process and exchange of data between the various tasks is thereby achieved using this mechanism.

7 Discussion and Conclusions

In this paper, we discuss the immunization and tracking application. It is an important, real and complex healthcare workflow application. The application requirements were collected through interactions with potential end-users from provider organizations and the healthcare experts at CHREF. The application is implemented on a testbed distributed over machines at both CHREF and UGA-LSDIS. It involves a realistic multi-enterprise heterogeneous computing environment consisting of multiple hardware, operating systems, communication infrastructure, as well as multiple databases with complete schemas.

The application posed many interesting and demanding requirements on the UGA prototype workflow management system METEOR$_2$. One particular aspect we discussed in this paper involved the support for both the Web-only workflow implementation (with multiple Web servers) as well as the CORBA (and Web) based workflow implementation for the to meet the requirements and capabilities corresponding to different participating healthcare organizations. Pros and cons for using the Web-only and the CORBA (and Web) based infrastructure have been discussed in some detail in the paper. The principal advantages of the Web-only implementation are its relatively low cost, high availability and popularity. The CORBA (and Web) implementation is evolving into a complete robust implementation capable of supporting transactional workflows. This implementation is capable of supporting more general and flexible control and data dependencies, increased distribution and load balancing, better support for recovery and exception handling, as well as transactional capabilities (in future using CORBA's Object Transaction Service).

Other novel aspects that have not been discussed in detail in this paper include the automatic code generation related to the coordination and information flow aspects of the workflow runtime code, support for fully distributed scheduling, support for two X.12 based EDI transactions, and some aspects of security. A number of other aspects including support for multimedia objects, authorization, error handling, recovery, performance issues (beyond back-of-the-envelope calculations), inter-workflow coordination, collaboration, help facilities using in-line multimedia, etc. require more work on our part or are not reported due to space limitations.

The application if deployed can provide significant benefits to the state's healthcare needs. Both at the clinical level, as well as at the level of the tracking subsystem, up-to-date and accurate information on patients can be immediately obtained from the appropriate databases. The system has the potential of reducing societal healthcare costs due to the monetary benefits of immunization tracking and by facilitating monitoring of the health needs of the state and catering to them immediately. System generated Medical Alerts inform providers about patient needs, and also assist in identification of contraindicators applicable for a patient. State health agencies can generate monthly reminders and overdue notices for parents and guardians regarding immunizations. Demographic reports can be generated for the state regarding child immunizations and HMO performances. Children delinquent on immunizations can be identified and can be brought to the attention of HMOs, field workers and other responsible personnel. The effectiveness and benefits of the system can also be monitored. Internet resources are increasingly available at hospitals, clinics and other health agencies which is a big advantage in today's information age. Several health-related Web sites (e.g., Centers for Disease Control) are now available, and could be of potential use in these organizations.

Demonstrating the advantages of using a workflow management system such as METEOR as compared to using one-at-a-time application development by hard-coding the task coordinations has not been an easy process. This has primarily to do with the application driven business decision making process and stove-pipe mentality. It has also been difficult to emphasize the need for a CORBA-based solution to the technically naive viewer (who often make decisions) when the user interfaces and the end-user interactions in the Web-only and CORBA-based implementations look similar. Moreover, the implications of distribution, scalability, robustness and error handling which are essential in any large-scale real-world application system are not all too apparent when the application prototypes are demonstrated to the decision makers. This information exchange process between UGA (the technology developer and prototype application developer) and CHREF (the production application developer and application system deployer) has involved many meetings and has been further facilitated by internships of two UGA graduate students at CHREF. They participated in both the development of $METEOR_2$ and the workflow application with CHREF.

The testbed is continually being expanded and refined as the application requirements are better understood. No modern complex application is likely to be static over a long period of time, and we do not expect immunization tracking to be the same as time progresses. As the application is being demonstrated to healthcare and government organizations by CHREF, additional enhancements are being suggested (e.g., for providing on-line reporting capabilities for schools). The application development and demonstrations have already led CHREF to undertake an additional project for deploying a part of the immunization tracking application with its client in the city of Bridgeport. Furthermore, testing of the $METEOR_2$ software at CHREF and planning for trials involving end-users (CHREF's clients) have also begun. Additional workflow technology enabled applications have also been identified by CHREF and issue of licensing $METEOR_2$ technology is being addressed.

References

[ANRS92] M. Ansari, L. Ness, M. Rusinkiewicz, and A. Sheth. Using flexible transactions to support multi-system telecommunication applications. In *Proc. of the 18th Intl. Conference on Very Large Data Bases*, pages 65–76, August 1992.

[Ass95] National Committee For Quality Assurance. *Health Plan Employer Data and Information Set (HEDIS)*, 1995.

[ASSR93] P. Attie, M. Singh, A. Sheth, and M. Rusinkiewicz. Specifying and enforcing intertask dependencies. In *Proc. of the 19th Intl. Conference on Very Large Data Bases*, pages 134–145, Dublin, Ireland, 1993.

[BDSS93] Y. Breitbart, A. Deacon, H. Schek, and A. Sheth. Merging application-centric and data-centric approaches to support transaction-oriented multi-system workflows. *SIGMOD Record*, 22(3):23–30, September 1993.

[Das96] S. Das. A distributed runtime for the $METEOR_2$ workflow management system. Master's thesis, University of Georgia, Athens, GA, August 1996.

[Dog96] A. Dogac. Special theme issue: Multidatabases. In *Journal of Database Management*. Winter 1996. 7(1).

[Elm95] A. K. Elmagarmid. Special issue on software support for work flow management. In *International Journal of Distributed and Parallel Databases*. Kluwer Academic Publishers, April 1995. 3(2).

[GH94] D. Georgakopoulos and M.F. Hornick. A framework for enforceable specification of extended transaction models and transactional workflows. *International Journal of Intelligent and Cooperative Information Systems*, 3(3):599–617, 1994.

[GHKM94] D. Georgakopoulos, M. Hornick, P. Krychniak, and F. Manola. Specification and management of extended transactions in a programmable transaction environment. In *Proc. of the Tenth International Conference on Data Engineering*, Houston, TX, February 1994.

[GHS95] D. Georgakopoulos, M. Hornick, and A. Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases*, 3(2):119–154, April 1995.

[Inc94] PostModern Computing Technologies Inc. *ORBeline User's Guide, The SMART Object Request Broker.* PostModern Computing Technologies Inc., 1994.

[JNRS93] W. Jin, L. Ness, M. Rusinkiewicz, and A. Sheth. Concurrency control and recovery of multidatabase work flows in telecommunication applications. In *Proc. of ACM SIGMOD Conference*, May 1993.

[KS95] N. Krishnakumar and A. Sheth. Managing heterogeneous multi-system tasks to support enterprise-wide operations. *Distributed and Parallel Databases*, 3(2):155–186, April 1995.

[Lee95] W. Lee. Hospital care planning. Technical report, Columbia University, 1995. http://americas.cs.columbia.edu.

[Lin96] C. Lin. A graphical workflow designer for the $METEOR_2$ workflow management system. Master's thesis, University of Georgia, Athens, GA, August 1996.

[MAGK95] C. Mohan, G. Alonso, R. Guenthoer, and M. Kamath. Exotica: A research perspective on workflow management systems. *Data Engineering Bulletin, Special Issue on Infrastructure for Business Process Management*, 18(1), March 1995.

[MSKW96] J. A. Miller, A. P. Sheth, K. J. Kochut, and X. Wang. Corba-based run time architectures for workflow management systems. *Journal of Database Management, Special Issue on Multidatases*, 7(1), 1996.

[Mur95] A. Murugan. Graphical workflow designer. Master's thesis, University of Georgia, 1995.

[Pal96] D. Palaniswami. A web-based runtime for the $METEOR_2$ workflow management system. Master's thesis, University of Georgia, Athens, GA, August 1996.

[RBP+91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. *Object-Oriented Modeling and Design.* Prentice Hall, Englewood Cliffs, NJ, 1991.

[RS95] M. Rusinkiewicz and A. Sheth. Specification and execution of transactional workflows. In W. Kim, editor, *Modern Database Systems: The Object Model, Interoperability and Beyond*, pages 592–620. ACM Press, New York, NY, 1995.

[She95] A. Sheth. Tutorial notes on workflow automation: Application, technology and research. Technical report, University of Georgia, May 1995. presented at ACM SIGMOD, San Jose, CA, http://lsdis.cs.uga.edu/publications.

[SJKB94] H. Schuster, S. Jablonski, T. Kirsche, and C. Bussler. A client/server architecture for distributed workflow management systems. In *Proc. of the Third International Conf. on Parallel and Distributed Information Systems*, pages 253–256, Austin, TX, September 1994.

[SR93] A. Sheth and M. Rusinkiewicz. On transacational workflows. *IEEE Data Engineering Bulletin*, 16(2):1–4, June 1993.

[Tec95] Action Technologies. Metro tour. Technical report, Action Technologies, Inc., 1995. http://www.actiontech.com.

[TV95] J. Tang and J. Veijalainen. Enforcing inter-task dependencies in transactional workflows. Technical Report J-2/95, VTT Information Technology, Espoo, Finland, January 1995.

[VEM95] VEMR. The virtual electronic medical record. Technical report, University of Virginia, 1995. http://custer.neuro.virginia.edu/niims/information/projects/VEMR.html.

[Wan95] X. Wang. Implementation and performance evaluation of CORBA-based centralized workflow schedulers. Master's thesis, University of Georgia, August 1995.

[WF94] T. White and L. Fischer. *The Workflow Paradigm - The Impact of Information Technology on Business Process Reengineering.* Future Strategies, Inc., Alameda, CA, 1994.

[Wor96] D. Worah. Design and implementation of exception handling and recovery in the $METEOR_2$ workflow management system. Master's thesis, University of Georgia, Athens, GA, August 1996.

[WS96] D. Worah and A. Sheth. What advanced transaction models have to offer for workflows ? In *Proc. of Intl. Workshop on Advanced Transaction Models and Architectures*, Goa, India, May 1996.

[WWWD95] D. Wodtke, J. Weissenfels, G. Weikum, and A. K. Dittrich. The Mentor project: Steps towards enterprise-wide workflow management. Technical report, University of Saarland, Department of Computer Science, Saarbrucken, Germany, November 1995.