Experimental Platform for Model-Integrated Clinical Information Systems

Janos Mathe‡, Jan Werner‡, Yonghwan Lee‡, Akos Ledeczi‡, Bradley Malin‡#, Janos Sztipanovits‡

‡Department of Electrical Engineering and Computer Science
#Department of Biomedical Informatics
Vanderbilt University
Electronic Medical Records (EMR) is an integrative project with three main goals:

- Build a credible testbed for EMR research
- Contribute to solving privacy and security challenges of EMR systems applications
- Use EMR application testbeds for the integration, testing, and evaluation of new technologies on core TRUST research areas, including:
  - Model-based design for security and privacy
  - Formal modeling, verification, enforcement of privacy & security policies
  - Data mining & representation of real clinical workflows
  - Security & privacy technologies for sensor networks
  - Public policy to technology interactions
1. Experimental platform for Model-Integrated Clinical Information Systems (MICIS)
   - Provide a common integration testbed for security and privacy aware Clinical Information Systems (CIS).

2. Component integration platform
   - Based on a standard Service-Oriented Architecture framework (SOA)
   - Extended Prolog-based Policy Evaluation Point & Policy Enforcement Point components (MICIS-PROPER)
     - Reusable
     - Platform-Independent
   - Integrated with the Apache Orchestration Director Engine (ODE)
3. Model integration platform
   - Built on Vanderbilt's metaprogrammable Model-Integrated Computing (MIC) tool suite
   - System models capture environment
     - Workflows
     - Services
     - Deployment
     - Messages
     - Message Attributes
     - Organizations
     - Roles
     - Access control policies
     - Security policies
   - Privacy modeling language based on Stanford's work on contextual integrity
     - Enables formal representation of permitted communications
     - Considers past, as well as future, communication instances
3. Model integration platform
   - Experimental platform has several components:
     - Set of domain-specific modeling languages
       - Captures relevant architectural components
       - Captures policy modeling aspects of selected CIS applications
     - Model transformations
       - Map domain-specific models on the MICIS component integration platform
     - Example application models
     - Running experiments for analytic analysis
Architecture: Applications

Component Integration Platform (MICIS-CIP)

User Interface Layer
- Front End (Webpage)
- Enterprise Service BUS
- Application A
  - Execution Engine
  - Orchestration Logic
  - CIS Orchestrator (CIS-O)
  - Execution Environment

Application B
- Web Services Container
- Web Service

Application C
- Front End (GUI)

Modeling Integration Platform (MICIS-MIP)
- Modeling Environment
  - Metamodel Editor
  - Metamodels
  - Metamodel Translator
- Model Transformation Layer
  - Front End Skeleton Translator
  - Model Editor
  - User Models
  - Model Editor
  - Execution Environment Translator
  - Verification Tool Translator
  - Verification Tool
- Execution Environment
  - Policy Translator
  - Policy Translator
  - Policy Set (Prolog Rules)
  - Policy Decision Point (PDP)
  - MICIS-PROPER

Enterprise Service BUS

Web Services
- Web Service

CIS Orchestrator (CIS-O)
- Execution Environment
- CIS-DB

Application A
- Web Services
  - Workflows
  - Workflows

Application B
- Web Services
  - Workflows

Application C
- Front End (GUI)
Architecture: Model Transforms
Architecture: Model Transforms

MICIS-PROPER a.k.a. Specification & Enforcement
MICIS-PROPER architecture

Modeling Integration Platform (MICIS-MIP)

Modeling Environment

- Model Editor
  - Workflow Models
  - Data Models
  - Policy Models

Model Transformation Layer

Prolog-based Policy Evaluation Point and Policy Enforcement Point (MICIS-PROPER)
MICIS-PROPER architecture

Web Service Container (Axis2 running on Tomcat)

Policy Engine

Policy Store

Policy Description

Policy Decision Point (PDP)

Policy Enforcement Point (PEP)

Context Handler

Web Service

Prolog-based Policy Evaluation Point and Policy Enforcement Point (MICIS-PROPER)
MICIS-PROPER architecture

Modeling Integration Platform (MICIS-MIP)
- Modeling Environment
  - Model Editor
    - Workflow Models
    - Data Models
    - Policy Models
  - Model Transformation Layer
    - Policy Translator
  - Policy Set
  - Policy Decision Point (PDP)
  - Policy Engine
    - Policy Store
    - Policy Description
  - Web Service Container (Axis2 running on Tomcat)
    - Policy Enforcement Point (PEP) Context Handler
      - Web Service

TROST
Team for Research in Ubiquitous Secure Technology
MICIS-PROPER architecture

- Integrated with Apache Orchestration Director Engine (ODE)

- Enabler
  - construct rigorous specification via privacy & security languages
  - experimental analysis of specification in complex system
  - description of security and privacy constraints with temporal aspects
  - rich user-defined contextual dependence
Tying it Together: An Example Scenario

Outpatient monitoring system

Wearable sensors, video capture, wireless networking

TRUST Project:
Berkeley
Cornell
Vanderbilt
Tying it Together: An Example Scenario

1. send AlertMessage

Outpatient monitoring system

Clinical Information System

CIS Orchestrator (CIS-O)
Tying it Together: An Example Scenario

Clinical Information System

Outpatient monitoring system

2. Log AlertMessage

EMR System

CIS DB

Clinical Information System

CIS Orchestrator (CIS-O)
Tying it Together: An Example Scenario

Outpatient monitoring system

3. Pull PatientRecord

CIS Orchestrator (CIS-O)

Clinical Information System

EMR System
Tying it Together: An Example Scenario

CIS Orchestrator (CIS-O)

Clinical Information System

Outpatient monitoring system

3. Show AlertMessage

Alert Monitor System
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Alert Monitor System

Outpatient monitoring system

I need to see this patient’s vitals!

Nurse

Monitors & verifies alerts
Tying it Together: An Example Scenario

CIS Orchestrator (CIS-O)

Clinical Information System

Alert Monitor System

Outpatient monitoring system

Nurse

Monitors & verifies alerts
Tying it Together: An Example Scenario

Outpatient monitoring system

4. Request PatientRecord

CIS Orchestrator (CIS-O)

Clinical Information System

Alert Monitor System
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Outpatient monitoring system

5. Request Patient Record

EMR System

Clinical Information System
Tying it Together: An Example Scenario

6. Pull Patient Record

EMR System

CIS DB

Clinical Information System

Outpatient monitoring system

CIS Orchestrator (CIS-O)
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Outpatient monitoring system

7. Send Patient Record

Alert Monitor System

Clinical Information System
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Outpatient monitoring system

Alert Monitor System

Nurse

Monitors & verifies alerts

Bob (the patient) needs help!
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Alert Monitor System

Outpatient monitoring system

Nurse

Monitors & verifies alerts

Validate Alert
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Outpatient monitoring system

Alert Monitor System

Nurse

Monitors & verifies alerts
Tying it Together: An Example Scenario

Clinical Information System

CIS Orchestrator (CIS-O)

Outpatient monitoring system

Alert Monitor System

Nurse
Monitors & verifies alerts
Tying it Together: An Example Scenario

Clinical Information System

Outpatient monitoring system

8. Relay Message

CIS Orchestrator (CIS-O)

Message Delivery System

Clinical Information System
Tying it Together: An Example Scenario

Outpatient monitoring system

9. Transform & Notify

Message Delivery System

Pager

Clinical Information System
Tying it Together: An Example Scenario

Clinical Information System

Outpatient monitoring system

Verified alert indicates a serious condition

Message Delivery System

Pager

Doctor

Many options:
- e.g., Call → provide directions to patient
- e.g., Alert EMS → bring patient to hospital
- e.g., …
• Clinical information system services, workflows, policies, roles are all captured in the models

• The system is automatically generated and deployed
Example: A Little Deeper

OPMAAlertMain

- Receive OPMAAlert
- Invoke LogService
- Invoke AlertMessage
- Reply

OPMAAlertMonitor

- Receive AlertMessage
- Invoke MessageRender
- Receive EMRRequest
- Invoke EMR
- Reply EMR
- Invoke EMRStore
- Receive EMRStore

- Monitor

EMR WS

Synchronous Message exchange
Asynchronous Message exchange
Access Control Policy Execution Point

Alert Message Render WS

Message Sender WS
Example Scenario

- When an anomaly is detected, the outpatient monitoring service issues an alert
- The clinical information system orchestrator (CIS-O) receives the alert message
- After logging alarm status in the EMR system, CIS-O sends the message to Alert Monitor System to render it on a monitoring station
- When the nurse checks the message → requests the patient’s medical record to evaluate the situation
Example Scenario

- Patient information includes medical history & contact information which can be used by the nurse to validate the alert.
- If the alert is deemed important, she writes the status to the patient medical record.
- Finally, CIS-O forwards the alert message to the designated doctors by using the Message Delivery System.
- Otherwise, the alert message is stored in the EMR system and the process is terminated.
Example: Sample Workflow Model

Workflow: OPMAlertStore process

Step 1: receive EMRStore
Step 2: invoke EMRStore
Step 3: receive AlertMessage
Step 4: invoke MessageSender
Step 5: receive
Example: Sample Workflow Model

OPMAAlertStore Process

Goal: store the result of nurse’s alert validation

Steps:

1. Alert status is assigned to the OPMAAlert data type

2. **Invoke EMRStore** activity invokes the PatientInformation web service
   a) Store the validation results in the EMR System
   b) Privacy policies applied when `invokeEMRStore` activity invokes Patient Information web service

Workflow: OPMAAlertStore process
Example: Sample Workflow Model

**OPMAalertStore Process**

**Goal:** store the result of nurse’s alert validation

**Steps:**

3. After the *receive* activity receives the acknowledge message from the web service, it assigns it to the *AlertMessage* variable

4. The *InvokeMessageSender* activity invokes the *MessageSender* web service to forward the alert message to the designated doctors via the Message Delivery System
Example: Sample Workflow Model

Workflow: OPMAlertStore process

**OPMAlertStore Process**

**Goal:** store the result of nurse’s alert validation

**Steps:**

5. After the MessageSender web service is completed,

6. The OPMAlertstore process returns.
Example: Policy Models
Policies Defined for Scenario

• Only medical staff is allowed to access alert messages

• Only primary care physicians are allowed to access patient’s medical record

• The nurse is allowed to access the records of patients monitored by the OPM system

• Medical staff is allowed to access patient’s record in emergency situation triggering the Break Glass policy
Example: Policy Models

- **Policy description includes**
  - Definition of incoming & outgoing data
  - Evaluation point
  - Obligations
  - Additional datasets for policy evaluation

- **Model contains information required to generate the policy:**
  - Query evaluated to determine access rights
  - Attribute relations used for policy evaluation
  - Textual policy description
Example: Policy Models

- Example query:
  - `retrievedata(PatientID, staffID)` after the service has been executed
  - Use a redefined set of predicates and attribute relations
    - `(is_critical(), treats(staffID, MRN))`

- These are generated from
  - incoming data
  - outgoing data
  by the Policy Enforcement Point (PEP)
Code Generation
Code Generation

Execution Environment
Translator

Deploy.xml

BPELDocument
(OPMAAlertMain Process)
Conclusions

- **Experimental Platform for EMR research**
  - Helping to solve privacy and security challenges of EMR systems applications
  - Usable for the integration, testing and evaluation of new technologies

- **Ongoing technology transition: Experimental Sepsis Management System for ICUs:**
  - Sepsis management protocol is formally defined: evidence-based medicine
  - Sepsis Management System is mapped on SOA platform
  - Model-Integrated systems approach
Acknowledgements

- NSF TRUST (CCF-0424422)

- Research Team
  - Yonghwan Lee
  - Akos Ledeczi, Ph.D.
  - Janos Mathe
  - Brad Malin, Ph.D.
  - Jan Werner
  - Janos Sztipanovits, Ph.D.