A MODEL-INTEGRATED AUTHORING ENVIRONMENT FOR PRIVACY POLICIES

Tihamer Levendovszky, Andras Nadas, Istvan Madari, Janos Sztipanovits
Institute for Software Integrated Systems at Vanderbilt University

Ethan K. Jackson
Microsoft Research
Content

- Introduction
  - The problem of policy formalization
- PATRN: the MDE tool for formal policy models
- Introduction to CLP and open world reasoning
- Semantic Anchoring of Models
- Verification and Execution
- Future Work
Motivation

- Health care delivery requires the coordination of activities across many different service providers and organizations and generally requires the secure exchange of health information across organizations.

- Privacy and care management policies at the federal, state, and institutional level lead to a confusing array of potentially contradictory or subjective policy interpretations.
Goals

- Extend on previous efforts
  - Contextual Integrity
  - Stanford’s Policy Formalization in Prolog
- Provide a more generic toolkit
  - Versatile representation
    - Analysis (Simulation, Validation, Verification)
    - Enforcement
    - Audit
  - Involve more non-IT stakeholders
    - Easy understanding without programming background
- Provide a Formal Modeling Methodology
PATRN: the MDE tool for formal policy models
1. Recognize the common patterns used in the textual policy descriptions. These patterns will form the templates in PATRN.
2. Compile the object and actors of the policies and organize them into ontologies.
3. Formal policy models are composed from the templates and ontologies.
4. Using the semantic anchors, the formal policy models can be translated for analysis or enforcement.
Patterns

- **Patterns**: `<ontology> | | <logics> | | <structure>`

- **Example**: 
  
  **SUBJECT** that satisfies **ACCESS CONDITIONS** is allowed to perform **OPERATION** on **OBJECT** given that **RESTRICTIONS** are met

SUBJECT, OPERATION and OBJECT are defined by ontologies. ACCESS CONDITIONS are defined with first order logic expressions. RESTRICTIONS are defined with first order logic constraints.
Example Policy

- VUMC Policy OP 10-40.15 - Use and Disclosure of Protected Health Information:

  “Use of PHI by VUMC workforce members is limited to the activities described in the definitions of treatment, payment, and healthcare operations or to research activities that are in compliance with policies set by the Institutional Review Board (IRB) and the Office of Research.”
PERSON is allowed to use OBJECT on SUBJECT for a certain ACTIVITY when CONDITIONS are met.
Instantiated Policy Template Model (VUMC Authorized Use)
Introduction to CLP and open world reasoning
FORMULA 2.0

☐ A system for modeling with logic
☐ Generic; not specifically designed to model software.
☐ Specifications are written as “open-world” logic programs.
☐ FORMULA 2.0 can verify, synthesize, transform, compile and check models all with logic
☐ Z3 SMT Solver
Formula adds data types to logic programming.

Data types are “algebraic”, i.e. they are functions that create data.

A data constructor always constructs the same value when provided the same arguments.

Two values are the same if and only if they were constructed by the same constructor with the same arguments.

\[
E \ ::= \ (\text{src}: \text{Integer, dst}: \text{Integer}).
\]

\[
U \ ::= \ E + \text{Integer} + \text{String}.
\]
Open World Reasoning

- Facts and Rules
- Open World Queries
  - P?G: Find a closure of the program by ground facts where a goal is satisfied. E.g. “Is document id accessible by x?”
  - P[F]: Partially close P with facts F and remove “new” marking from all associated data types. E.g. Can x access any documents?
Solving and Search

Use state-of-the-art **satisfiability modulo theories** (SMT) solver Z3 to solve quantifier-free formulas.

- **FORMULA Specification**
- **Symbolic Execution**
- **SMT Formula**
- **Add symmetry breaking**
- **Z3 Solver**

- **Guess symbolic world**
- **Encode solution region**
- **Try something new**
- **Pick next region**
Semantic Anchoring
Semantic Anchoring

- Instance Model (Term Algebraic Representation)
  - Instantiates
  - Semantic Mapping (Execution of Logic Programs)
  - Defines
  - Semantic Models (Formal Structures in Term Algebra)
    - Specifies
    - Semantic Mapping Rules (Logic Programs)
    - Defines
  - Structural Constraints
    - Specifies
    - Semantic Domain (Term Algebraic Representation)
      - Defines
      - Structural Constraints
        - Defines
        - Operational Semantics
          - Specification for Operational Semantics (Logic Program)
            - Specification for Operational Semantics (Logic Program)
  - Testing and verification
    - Updates, Revisions, extensions
  - Intuitions, standards, and variations

Grounding
- Denotational Semantics in Mathematics
How we do things differently: Formalizing Policies with logic vs. Models and Semantics

Policy Text (HIPAA, States, Institutions, etc.) → Execution Domain
- Formal Logic Representation (Prolog, rules) → Execution

Logic programming

Models with semantics

Models (HIPAA, States, Institutions, etc.) → Execution Domain
- Models → Execution
- Semantics → Execution

Execution Domain 1.
- Models
- Semantics d1.

Execution 1.

Execution Domain n.
- Models
- Semantics dn.

Execution n.
Verification and Execution
Reasoning Module

- Policies are anchored to Formula
- Semantics are provided for state space exploration
- The SAT solver under Formula calculates the satisfying solution space for each policy.
- The solution spaces are compared to each other
Future Work
Future Work

- Move the tools to a more convenient Web based technology
- Create a policy and use case repository
- Adding more content
- … and more content
Thank You!

Questions?
Tools

- **PATRN Modeling Suite**
  - Developed in ISIS at Vanderbilt for SHARPs
  - Uses GME (Generic Modeling Environment) and Formula
  - [https://wiki.isis.vanderbilt.edu/sharps/index.php/PATRNIntro](https://wiki.isis.vanderbilt.edu/sharps/index.php/PATRNIntro)

- **Formula** (Formal Modeling Using Logic Programming and Analysis)
  - Developed by Microsoft Research
  - Uses Logic Programming over a SAT solver
  - Provides model based domain definitions
Iterative cycle of system design

1. Requirements and Policy specification
   Collect and create all requirements specification of the HIE system; including policies and architectures

2. Ontology extraction and modeling
   Analyze the specification and define the ontologies for both the system component and the policy component of the models

3. Information Flow and Policy modeling
   create the information flow and policy models using the ontologies

4. Verification, composition and simulation
   Verify the policies, compose the policies with the information flow and system models and simulate/verify the system behavior of the composition

5. System integration and executable policy code generation
   Export the models to be used in the final system
The gap

Policies

• Federal
  • HIPAA
• State regulations
  • General Health Information
  • Substance Abuse
  • Mental Health
  • Pharmacy records
  • HIV/STD records
  • Consent (in some cases)
• Institutional
  • Consent process
  • Employee access
  • Patient access
  • Disclosure process

HIE Requirements

• Document types
  • Legal Medical Record (LMR)
  • Designated Record Set (DRS)
  • Discharge Summary
  • Results (Lab/Imaging/etc.)
• Personnel requirements
  • Physicians
  • Nurses
  • Other Providers
  • Administrative
• Authorization
  • Locations (Physical and IP)
  • Admissions/Discharges
  • Certification
Code Generation with String Template

- Code Generator uses Code Templates with StringTemplate v4 Library.
- With only changing the templates it’s possible to generate code in different languages, like Formula, Drools, iLog IRL, etc.

Source Code Language Template:
- General Code Template to describe the language. Used to generate the ontologies and models that have static semantics.

Anchored Semantics Definition Template:
- Code Template that define the semantics of each Policy Template and Relation.
Health Information Exchange (HIE) Policy Simulation and Enforcement

Federal, State, Institutional Privacy Policies

Policy Authoring / Information Flow Modeling

Patterns | Semantics | System/Actor Models
---|---|---
Ontologies | Logic or Programming Language | Use Cases

HIE System Specification
Usage Scenarios

Authoring Enforcement

HIE System
Request Processor/Server

Request
Reply

Enforce. point

Executable Policy Code

Use Case Simulation Code

HIE System Simulator
Use Case Enactment (Policy Unit Testing)

Simulation Verification