

RDF

Quad structure a minimum

Towards Literature Survey

April 07 2009

Ankesh

Outline

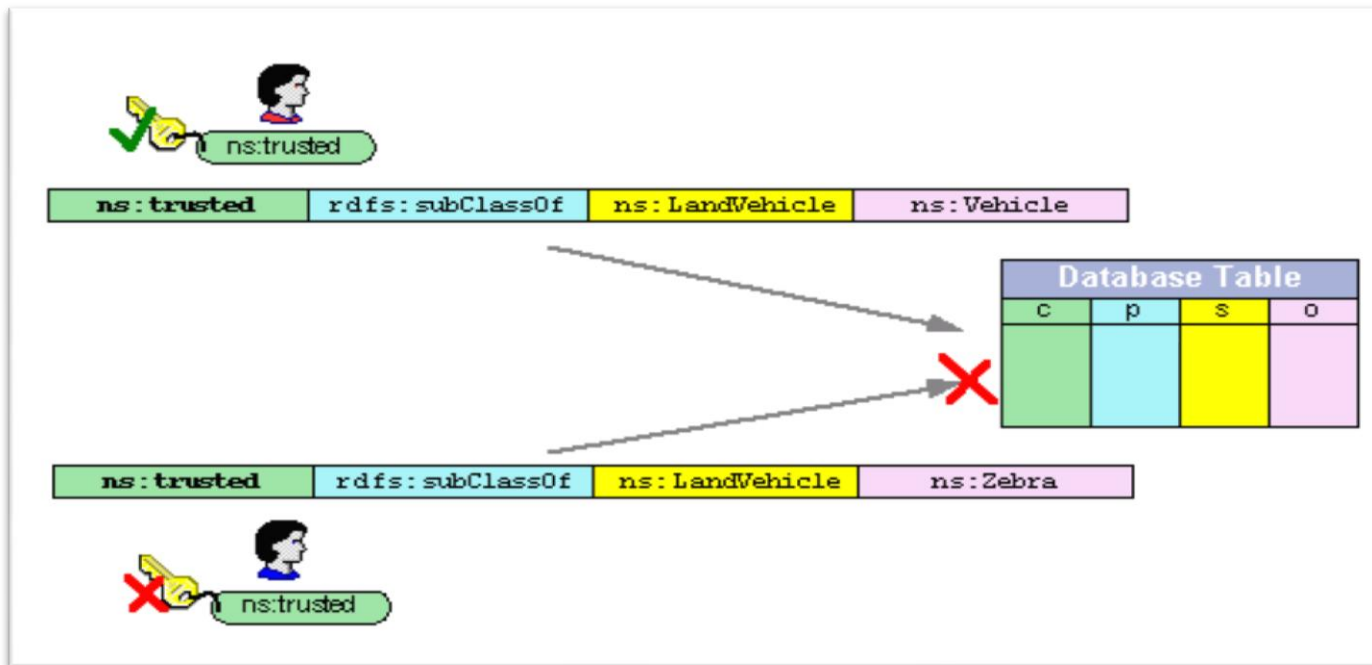
- General discussion on policies [3- 13]
- Overview of Surveyed Papers [14 – 42]
- Summary/ Conclusion [43 – 45]

RDF Triples

- RDF is based on the idea of **identifying things using Web identifiers (URIs)**, and **describing resources in terms of simple properties and property values** [*Introduction of RDF-Primer*]
 - Triple structure was sufficient for <URI> <property> <value>
1. Provenance needs crept up – **<Alice> says <X> <Y> <Z>**. (API specific against RDF principle)
 2. **Named graphs** – <graph-name> <X> <Y> <Z>. RDF statements that describe graphs (Eg. can evaluate specific graphs using task-specific trust policies)
 - **[Named Graphs, Provenance and Trust. Carroll. Hayes et. Al. www2005]:**
 - **Data syndication** – provenance information and provenance chains
 - **Restricting information usage** - intellectual property rights, privacy preferences (CC, P3P)
 - **Access control** - fine-grain access control (Intellidimension. RDF Gateway - Database Fundamentals.)
 - **Signing RDF Graphs** - it is necessary to keep graph that has been signed distinct from the signature. [Signing RDF Graphs. Carroll. ISWC 2003]
 - **Stating propositional attitudes** such as modalities and beliefs
 - **Scoping assertions and logic** where logical relationships between graphs have to be captured
 - **Ontology versioning and evolution**

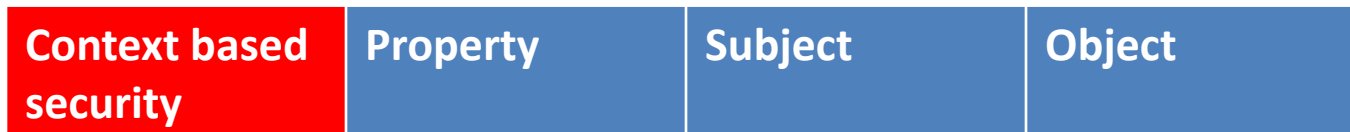
4th (/ ≥ 4) element

- RDF Gateway uses **context based security** to control access to groups of statements in a database table.



Source: Intelli dimension: A SW infrastructure company
(<http://archive.intellidimension.com/default.rsp?topic=/pages/rdfgateway/dev-guide/db/db.rsp>)

Policy – the 4th element



- Semantic mark up of web entities with policy languages (not new, e.g. Rei)

RDF document

```
<rdf:Description rdf:about="">
  <pol:policy rdf:resource="#general-policy"/>
  <rdf:comment>
    The general-policy applies to all the triples in this RDF document
  </rdf:comment>
</rdf:Description>
```

Triple

```
< rdf:Statement >
  <rdf:subject rdf:resource=" #sub"/>
  <rdf:predicate rdf:resource=" #prop"/>
  <rdf:object rdf:resource=" #value"/>
  <pol:policy rdf:resource="#triple-policy"/>
  <rdf:comment>
    The triple-policy applies to the "sub" "prop" "value" triple.
  </rdf:comment>
</rdf:Statement>
```

Resource

```
<AcPol:AccessibleResource rdf:about="AccRes">
  <rdf:policy rdf:resource="#Access-control-policy">
  <rdf:comment>
    This AC-policy defines restrictions on access
  </rdf:comment>
</AcPol:AccessibleResource>
```

Policy types, use of semantic technologies, Rein and PAW

POLICIES SO FAR

Policies in general

- **Access control, security**, protection, model **systems behaviour**
- Traditionally: **identity based** (password protected, public key certificates) and **role based** (credentials- member, team, public).
Some disadvantages:
 - classes must be defined in advance, creating temporary is difficult.
 - Difficult to set up a fine grained way in a directory-file based protection. (eg. access to page or particular data)
- **Rule based**: (Eg. if affiliated with RPI and its not rainy) Requesters for data provide a demonstration that they satisfy the policy encoded in the rules.
 - KeyNote, Ponder, Trust-X, Protune, Rei
- MAC, DAC (owner has a say)

Policies + Semantic Technology

- Ontology based policy representation
 - DL Based : KAoS
 - Rule based : KAoS, Rei, Protune
 - Deontic Logic Based (Rei) – rights, prohibitions, obligations, dispensations
- Define policy classes (Positive Authorization), action types (Retrieval) and actor types (Doctor). Actions and actors may have property values.
- Adv: Can easily be integrated with domain specific ontologies.
- Semantic markup with policy related information

Harmonizing policies : Common framework

- Challenge: Formalizations are based on different derivation strategies and reasoning mechanisms
- No Single Standard Policy Language
- **Rein Framework** for Policy Management on the Web – supporting heterogeneous policy systems [Using Semantic Web Technologies for Policy Management on the Web. L. Kagal, T. Berners-Lee et. al. (AAAI 2006)]
 - Over policy languages and domains described in RDFS, OWL or N3 rules
- Resource → Policy → Policy Language → Meta-policies
- **Access control** : reasoning engine decides whether requests for *resources* is valid. (+ delegation, no obligations).
- Cwm based implementation. Builtins for cryptography, string functionality and math operators.

Policy Aware Web

- Rule based DAC
- Rules can be published, searched, browsed and shared using HTTP.
- Burden of proving that the requester has right to access on requester. She provides proof to the server which verifies the proof to grant access.
- Proofs represented using ontologies (PML)

```
:PF8 a Proof:Step;  
Proof: StepAssertion {:Auth :Kari :URI1};  
Proof:StepRule log:ModusPonens;  
Proof:StepDependsOn (:PF6 :PF7).
```

- Share policies (transparent) and proofs (accountable).

[Weitzner, Hendler, Berners-Lee, Connolly, Creating the Policy-Aware Web: Discretionary, Rules-based Access for the World Wide Web]

Challenges for PAW

1. The system receiving the proof can check its correctness with respect to only those rules of logic that it accepts. Often security languages combine basic inference rules with application-specific inference rules to make application-specific logics.
 - System should be able to ascertain the correctness of proofs based on different security logics. (Hint PCA)
2. Coherent representation and operation with inconsistencies of the web. (FOL \leftarrow Annotations Logic)
3. Developing protocols and standards for proof generation.

Proof-Carrying authentication (PCA) is a “non-standard” logic on the Web designed and implemented for a general and powerful **distributed authentication framework** *based on a higher-order logic*. It enables authentication frameworks based on different logics to interact and share resources.

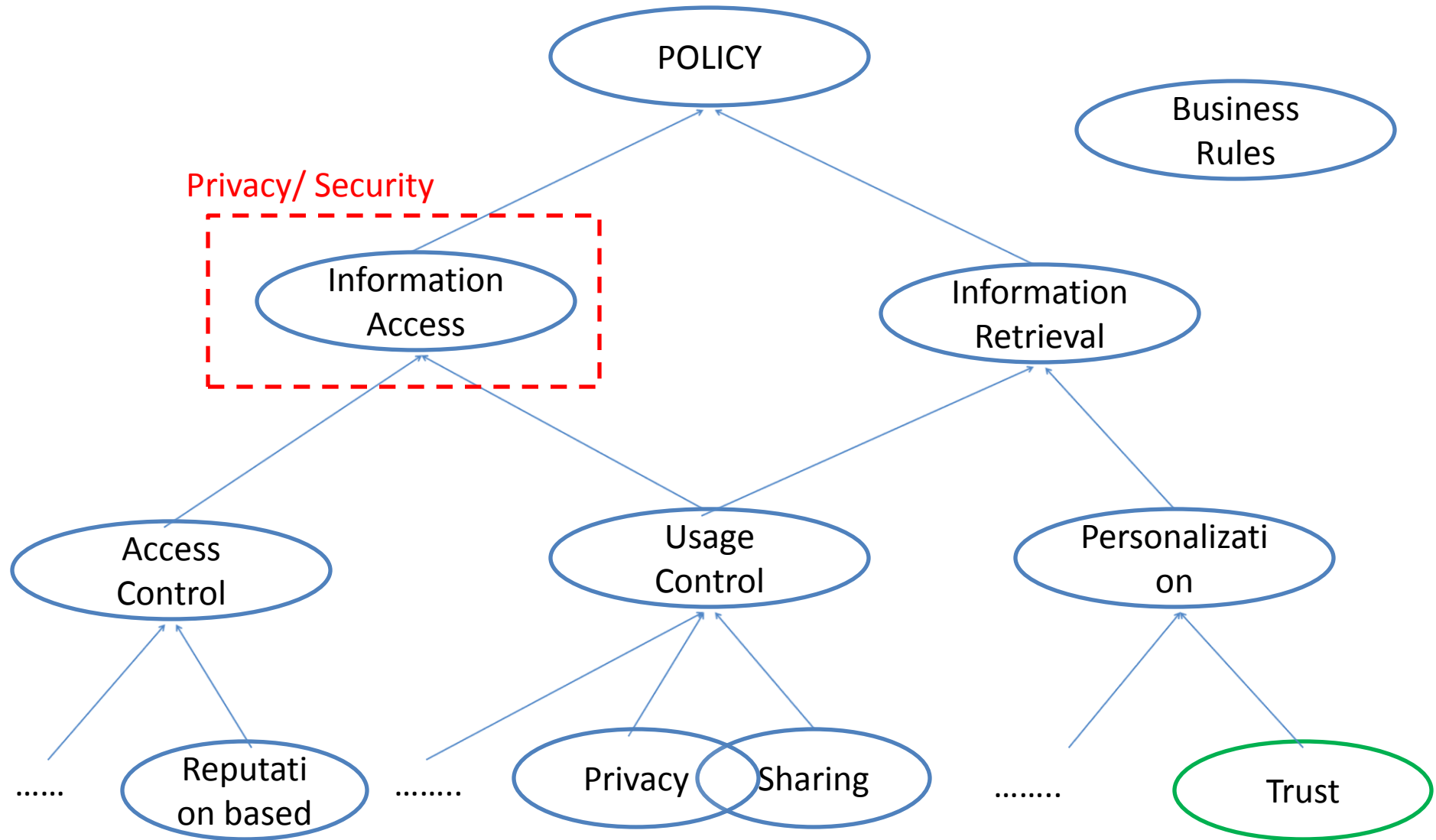
Proof-Carrying Authentication

- Their logic has **no decision procedure**. Users submit proofs with their requests. **Proof checking is simple**.
- Therefore distributed authentication is possible.
- It defines a core logic, an application-specific logic (operators and rules) can be defined on top of it.
- Main abstractions it supports:
 - Name-to-key-bindings
 - Access control
 - Delegation/ Revocation
- Statements are represented as (digitally signed) data-structures.

Policy Classification, Notion of data ownership, Policy for usage control

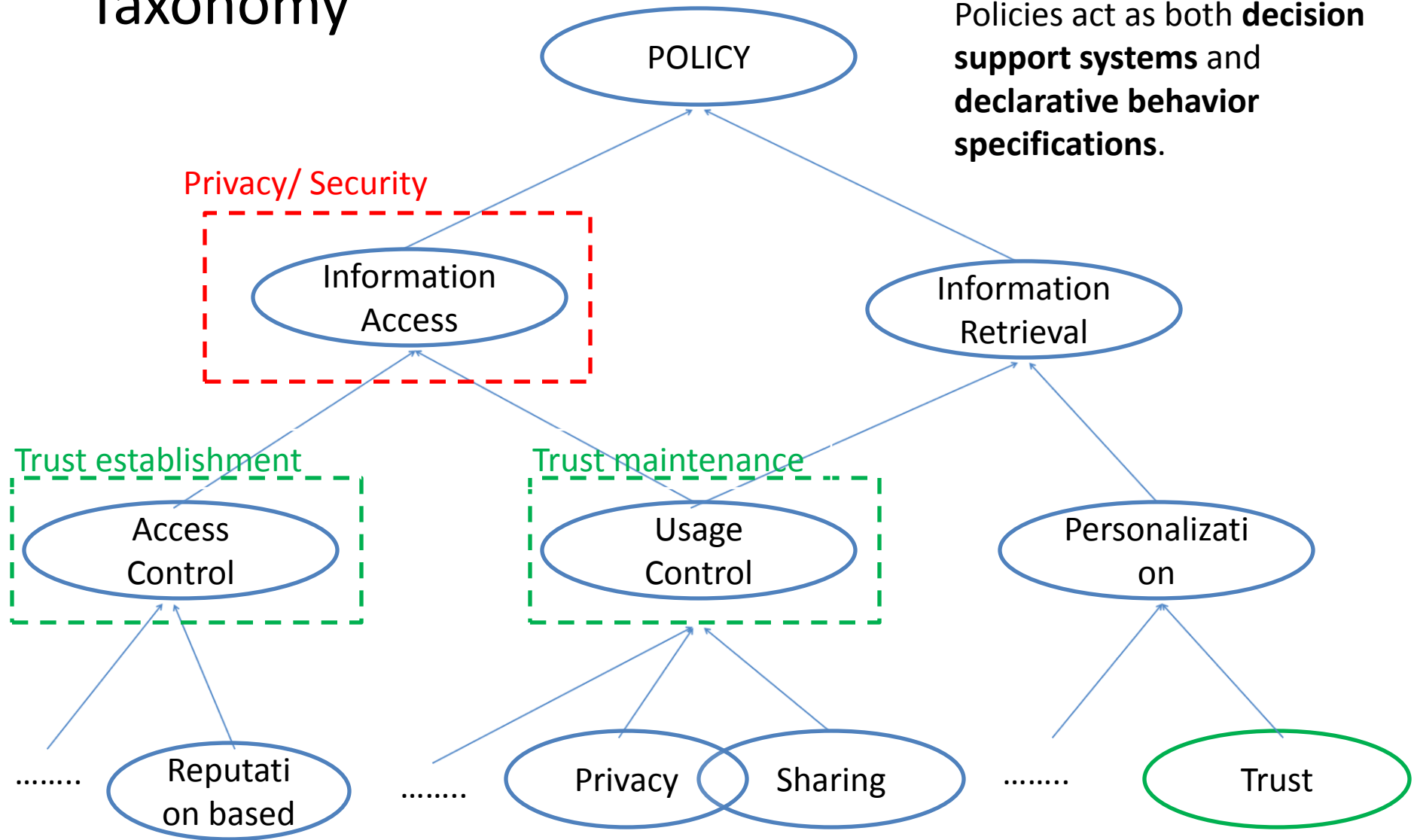
POLICY FOR USAGE CONTROL

Taxonomy



Taxonomy

Policies act as both **decision support systems** and **declarative behavior specifications**.



Policy Examples

Access Control

- Allow access to [associated-with TWC; studentAt RPI].

Usage Control

- Who should see what data, who should be able to copy that data, and what they should be able to do with it?
- *Sharing*: Must give credit (Eg. <photo> <cc:src> <Bob>)
- *Privacy*: What all private information I am ready to reveal.
- *Privacy*: What happens to my private data?

Personalization

- Filter high-quality information (metric: provenance info, reputation) [below]
- show all money amounts in dollars
- do not show * quizzes taken by my friends (application view → web view)
- When to display photos?

[**Quality-driven information filtering using the WIQA policy framework.** C. Bizer, R. Cyganiak. Web Semantics: Science, Services and Agents on the World Wide Web(2009)]

From Blogs

DataSphere [Beyond Applications - Introducing the Planetary Datasphere (Part 2). Orri Erling's Weblog. <http://www.openlinksw.com/weblog/oerling/?id=1537>]

- To facilitate open, distributed and scalable information access on the Web – we need strict usage policies.
- “While the Document Web enforces security when a thing is returned to the user, **Linked Data Web enforcement must occur whenever a query references something**, even if this is an intermediate result not directly shown to the user.”

DataPortability [Blog- E. Bizannes. <http://liako.biz/2008/03/my-presentation-at-kickstart-forum/>]

- “**Scope of Control**, however, seems to **stem from ownership**. That is, **you should only be able to control what you own**.”
- Ownership? : “**the ability to deny use of an asset by another entity**.”
- “**Ownership**’, however, is **tricky** when you are talking about bits and bytes that are **getting shared, indexed, replicated and mixed together**”

Ownership of Semantic Web Data

- **Ownership not always relevant** : popular ontology (owl, wine), facts (<RPI> <in> <Troy>)
- **But some places it is** : <me:Alice> <height> “5.5”. What if some one adds <me:Alice> <aids_status> “positive”.
- What if some one(Bob) declares <new:Alice> **owl:sameAs** <me:Alice>?
- What if some-one adds <new:Alex> <married-to> <me:Alice>? Does owner of <me:Alice> have right about this triple?
- **Triple (← graph) ownership** instead of resource ownership?
- By the way how many ways can the data be used or modified?

SW Data Usage + Policy?

- **“Privacy policy that needs the names removed from medical records before the summary information can be distributed to medical researchers who in turn can only publish the results in an open access journal, which must not allow commercial advertising next to the paper.”** [*A Policy Oriented Architecture for the Web: New Infrastructure and New Opportunities*. I. R. Henricksen. W3C Workshop on Languages for Privacy Policy Negotiation and Semantics-Driven Enforcement]
- Not just individuals- labs, companies warrant selective usage.
- A posteriori check.
- Audit. Logging.
- CC(limited form of restrictions), P3P (privacy on websites), DRM (special devices)

USAGE CONTROL, A POSTERIORI AUDIT

Papers on Usage Control and Audit

1. **Distributed usage control.** A. Pretschner, M. Hilty, D. Basin. Communications of the ACM ,Privacy and security in highly dynamic systems SPECIAL ISSUE (Sept 2006)
2. **Audit-based compliance control.** J. G. Cederquist, R. Corin, M. A. C. Dekker, S. Etalle, J. I. den Hartog and G. Lenzini. International Journal of Information Security (2007)
 - a) **An audit logic for accountability.** Cederquist J.G. et. Al. POLICY 2005.
 - b) **Logic for auditing accountability in decentralized systems.** Corin R. et. al. IFIP Workshop on Formal Aspects in Security and Trust (FAST) (2004)
3. **A Posteriori Compliance Control.** S. Etalle, W. H. Winsborough. Symposium on Access Control Models and Technologies (2007)
4. **Policy Evolution in Distributed Usage Control.** A. Pretschner. 4th International Workshop on Security and Trust Management (2008)
5. **Data-Purpose Algebra: Modeling Data Usage Policies.** C. Hanson, T. Berners-Lee, L. Kagal, G. J. Sussman, D. Weitzner. POLICY 2007)

1. Distributed Usage Control

- “Fundamentals of access control appear to be well understood, this is not the case for usage control” – a conceptual framework for specification and enforcement
- Assumptions: Sensitive data are stored at trustworthy places called **data providers**; To prevent unintentional violations.
- **Data consumers** request access to the data. What happens to the data once it has been released?
- Roles of consumers and producers are dynamic and interchangeable.
- Each data item has data owner who possesses the rights to the data.
- Usage requirements that can be **directly enforced** or whose **satisfaction** can at least be **observed** (audit) – **compensation**.

1. Concepts @ Distributed Usage Control

- **Actors:** Information systems or information processing device.
- **Action** that Actor can take:
 - **Operations on data:** storage, distribution, read access, computation of statistics
 - **Communication:** sending and receiving of messages not subject to usage control. E.g. request for data.
- **Provisions-** access control, **obligations-** requirements data consumer must adhere to. The constraints on operations on data
 - Time (stored <30)
 - Cardinality (copy <3)
 - Events (owner revokes)
 - Actions (notify each data usage)
 - Purpose (scientific purposes)
 - Technical / governance restrictions (encrypted storage)
 - Updates (correctness of personal data)

1. Policy Language @ Distributed Usage Control

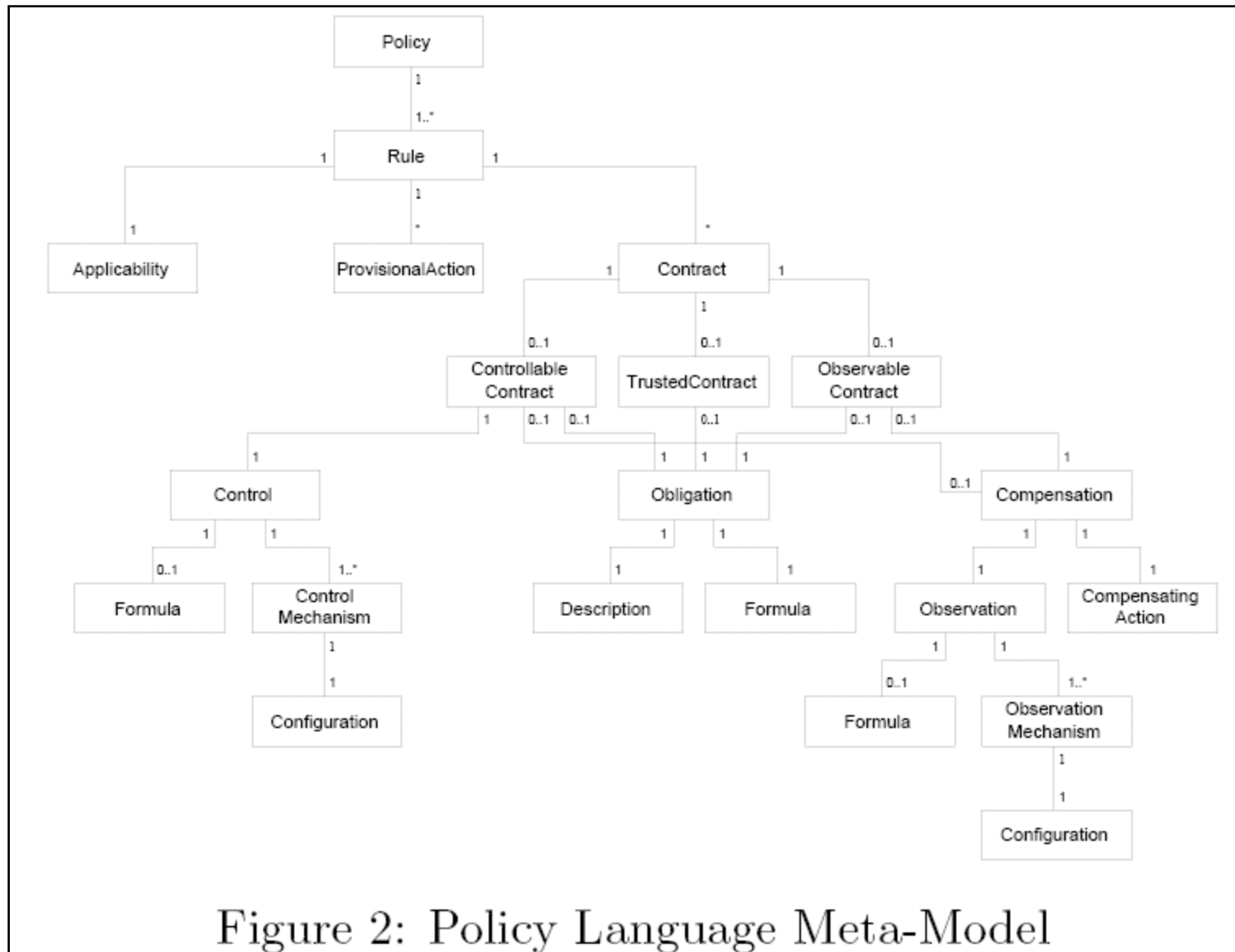


Figure 2: Policy Language Meta-Model

1. More clarity needed on usage control def. @ Distributed Usage Control

Authors reflect following missing:

1. Are usage control in context of privacy and IP management fundamentally different?
2. The propagation of rights and controlling the ways that data can be combined and distilled.

2. Audit-based compliance control

- **Framework that models ownership of data**
- Best **suited for a collaborative work environments**, where a small group of users create and manage document in decentralized way-flexible approach for controlling compliance to policies (> DAC Windows or Unix filesystems)
- **Users are administrators.** They can make exceptions to standard policy, provided they take the responsibility
- Policies are not enforced a priori, a posteriori checks.
- Audit user actions for compliance. Auditors observe critical actions
- User keeps a secure log of their actions and circumstances to provide favorable facts to auditors. (users take responsibility for their actions)
- Individual agents must find proofs to justify their actions. Auditors only checks the justification proofs.

2. Policy Language@ Audit-based compliance control

Policy language is similar to that of PCA

- **Atomic predicates** for **Permissions** for actions
- FOL formulas
- ξ obligations
- maySay – a is authorized to say ϕ to b. (adm. policy)
- $\phi \rightarrow \Psi$ - proof of ϕ is needed to obtain the permission Ψ
- The framework is semi-decidable

$$\begin{aligned} \phi &::= P(s_1, \dots, s_n) \\ &| \text{maySay}(a, b, \phi) \\ &| \text{owns}(a, d) \\ &| \top \mid \phi \wedge \phi \mid \forall x. \phi \mid \phi \rightarrow \phi \mid \xi \rightarrow \phi \\ \xi &::= !act \mid ?act \end{aligned}$$

Policy Grammar

2. Example Policy spec@ Audit-based compliance control

H1 $\forall a, d. (\text{isPatient}(a) \wedge \text{isPI}(a, d)) \rightarrow \text{owns}(a, d).$

H2 $\forall a, d. (\text{isPatient}(a) \wedge \text{isMD}(a, d)) \rightarrow \text{owns}(a, d).$

H3 $\forall a, b. (\text{isDoctorOf}(b, a) \wedge \text{isPI}(a, d)) \rightarrow \text{mayRead}(b, d).$

H4 $\forall a, b. (\text{isDoctorOf}(b, a) \wedge \text{isMD}(a, d)) \rightarrow (\text{mayRead}(b, d) \wedge \text{mayUpdate}(b, d)).$

H5 $\forall a, b, c. \text{isDoctorOf}(b, a) \rightarrow \text{mayGiveDrug}(b, a, c)$

H6 $\forall a, b, c, d. (\text{isDoctorOf}(b, a) \wedge \text{onStaffOf}(c, b)) \rightarrow b \text{ says } \text{mayGiveDrug}(c, a, d) \text{ to } c.$

H7 $\forall a, b, c, d. \text{isAdministration}(c) \rightarrow (!\text{giveDrug}(b, a, d) \rightarrow \text{mayBill}(c, a, d)).$

The Hospital's policy written in Audit Logic's Policy Language

[Audit-Based Access Control for Electronic Health Records. M.A.C. Dekkera, S. Etalle. Electronic Notes in Theoretical Computer Science (ENTCS) (2007)]

2. Proof @ Audit-based compliance control

Proof obligation: which policy an agent needs to satisfy in order to justify the execution of an action.

$$\begin{aligned} \text{pro}(\text{create}(a, d), b) &= \top \\ \text{pro}(\text{comm}(a, b, \phi), a) &= \text{maySay}(a, b, \phi) \\ \text{pro}(\text{comm}(a, b, \phi), c) &= \top \quad (a \neq c) \\ \text{pro}(\text{giveDrug}(a, b, c), a) &= \text{mayGiveDrug}(a, b, c) \end{aligned}$$

Proof Obligation

Conclusion derivation: describes what policy an agent can conclude from the evidence of an action that occurred

$$\begin{aligned} \text{concl}(\text{create}(a, d), a) &= \text{owns}(a, d) \\ \text{concl}(\text{create}(a, d), b) &= \top \quad (b \neq a) \\ \text{concl}(\text{comm}(a, b, \phi), b) &= \phi \\ \text{concl}(\text{comm}(a, b, \phi), c) &= \top \quad (c \neq b) \end{aligned}$$

Proof Conclusion

2. Audit Logic: Audit-based compliance control

Agent that owns a piece of data can derive any policy for the data

$$\frac{data(\phi) \subseteq \{d_1, \dots, d_n\}}{\Gamma_1, \text{owns}(a, d_1), \dots, \text{owns}(a, d_n); \Gamma_2; \Delta \vdash_a \phi} \text{owns-L}$$

Policy can only be refined.

$$\frac{\Gamma_1; v; v \vdash_a \psi}{\Gamma'_1, \text{maySay}(b, c, \Gamma_1); \Gamma_2; \Delta \vdash_a \text{maySay}(b, c, \psi)} \text{refine}$$

2. Accountability @ Audit-based compliance control

- **Logged action** is a triple

(Condition \subset Policies)

$\langle \text{Action, Conditions, Obligations} \rangle$

Logged Action

- **Log** is a list of logged actions

- Accountability from logs

Actions; Conditions; obligation \vdash *pro* (act, actor)

Accountability

- **prolog** implementation for finding proof, **Twelf** for proof checking

- Unanswered: How **Post Obligation** can be audited?

3. A Posteriori Compliance Control

- **A Posteriori Policy Enforcement** - APPLE
- “Enforcement mechanisms/ systems are incapable of **continuing to enforce policies on data objects as they move from one system to another.**”
- Detering unauthorized use- preventative policy enforcement:
“In the late '90's, a high-level IT manager at the NYSE told one of the authors that **access control on the trading floor was unnecessary because of auditing and the high opportunity cost of losing one's trader job.**”
- Significance of using a logical framework:
 - **Defines compliance** in a precise, unambiguous manner
 - Organizes the **content that must be included in the log**
 - **Defines precisely what queries an auditing authority can expect** answers to.
- Users may *receive, modify* and *redistribute* documents.
- Restrictions while movement. For eg. entailment check: **policy belonging to the document in possession entails the policies of its parent document.**

3. Formal System @ A Posteriori Compliance Control

- A: Individual or process is a process is a **principal** (*agent*).
- A.r : Principal can define a **role** (denotes a set of principles). A is the owner of A.r. (Only) A can issue four credentials:
 - $A.r \leftarrow D$
 - $A.r \leftarrow B.r1, A.r \leftarrow B1.r1 \cap B2.r2$. (**delegation**)
 - $A.r \leftarrow A.r1.r2$ (linking inclusion)
 - Eg. $D.trusted \leftarrow D.auditable \cap D.accountable$
 - Eg. $D.auditable \leftarrow D.taa.compliant$ (trusted-auditing-authority)
- **Policy atoms:** {owner(A), maymodify(A), mayrefine(A), maytell(A, B)}
(A or B may be role expressions)
- **Document:** $\langle id, ancestral-documents, policy-label \rangle$
(policy label is conjunction of policy atoms)

3. Formal System @ A Posteriori Compliance Control

- **Log**: seq of entries ::= Action | {credentials at a time} | role-membership
- To show if A is **authorized to have a document doc**, A must prove, $\log \vdash \textit{have}(\textit{doc})$.
- To show **legitimacy of action**, $\log_{\text{prev}} \vdash \textit{action}$.

APPLE \ Audit-Logic:

- Incorporation of trust
- Dynamic groups ($A.r \leftarrow D$)
- Derivative tracking (ancestral documents)

4. Policy Evolution in Distributed Usage Control

- Formal framework that encompasses both rights and duties (in uniform manner) for re-distribution in usage control.
- During re-distribution rights can be decreased and duties can be increased. (**sub-policies can be strengthened by roles that have the right to do so.**)
- **Event:** (play, {(object, o)}), **Refinement:** (play, {(object, o), (device d)}). Event: (play, {(obj, movA), (vq, 75)})
- **Rights:** permitonlyparam({0,...,75}, vq, play, {(obj, movA)}).
Permitonlyevname ({edit, sredit, vdoedit, play}, {(obj, movA)})
- **Duty:** $\text{always}(E_{\text{fst}}((\text{play}, \text{play}); \{(\text{obj}, (\text{movA}, \text{movA}))\}) \implies E_{\text{fst}}((\text{pay}, \text{pay}), \{(\text{amt}, (1, T_p))\}))$
- **Incr:** $\text{always}(E_{\text{fst}}((\text{play}, \text{play}); \{(\text{obj}, (\text{movA}, \text{movA}))\}) \implies E_{\text{fst}}((\text{pay}, \text{pay}), \{(\text{amt}, (2, T_p))\}))$

4. Event Ordering @ Policy Evolution in Distributed Usage Control

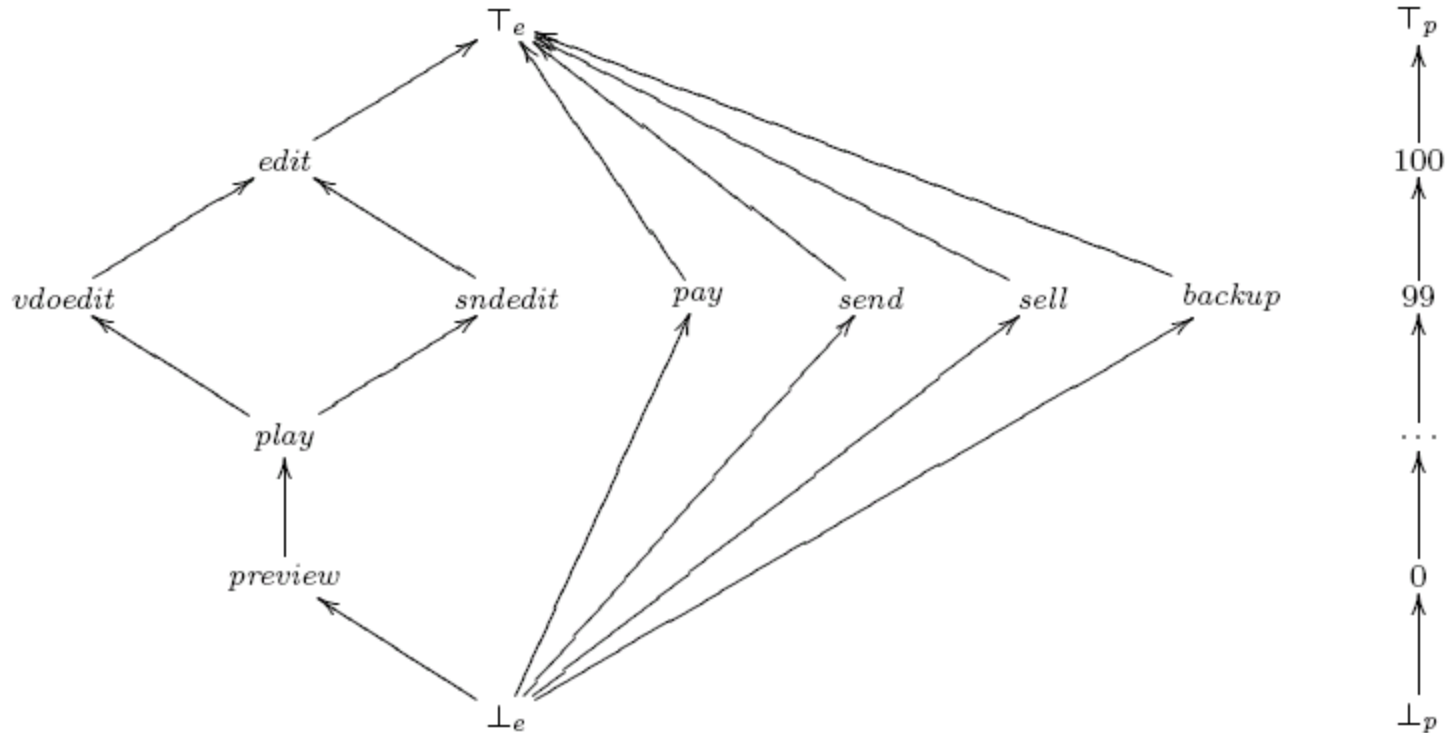


Fig. 4. Partially ordered event names (left) and parameter values (right)

- If events `edit` and `play` are allowed then so are `{voedit, sndedit}`

always(not($E_{all}(((\perp_e, play), \{(obj, (movA, movA)), (vq, (76, \top_p))\}))))$).

4. Usage Conditions @ Policy Evolution in Distributed Usage Control

Usage Conditions

$$\{(\text{dealer}, \underline{\text{permitonlyevname}}(\{(\perp_e, \text{sell}), (\perp_e, \text{send})\}, (\text{obj}, (\text{movA}, \text{movA}))))), \\ (\text{customer}, \underline{\text{permitonlyevname}}(\{(\perp_e, \text{play}), (\perp_e, \text{backup}), (\perp_e, \text{send})\}, \{(\text{obj}, (\text{movA}, \text{movA}))\})), \\ (\text{reporter}, \underline{\text{and}}(\underline{\text{permitonlyevname}}(\{(\perp_e, \text{play})\}, \{(\text{obj}, (\text{movA}, \text{movA}))\}), \\ \underline{\text{repmax}}(3, E_{fst}((\text{play}, \text{play}), \{(\text{obj}, (\text{movA}, \text{movA}))\})))), \\ (\text{academia}, \underline{\text{permitonlyevname}}(\{(\perp_e, \text{edit}), (\perp_e, \text{send})\}, \{(\text{obj}, (\text{movA}, \text{movA}))\})), \\ (\text{default}, \underline{\text{permitonlyevname}}(\{(\perp_e, \text{preview})\}, \{(\text{obj}, (\text{movA}, \text{movA}))\})).$$

Distribution Path

$$\{(\text{dealer}, \underline{\text{permitonlyparam}}(\{(\text{academia}, \text{academia}), (\text{reporter}, \text{reporter}), (\text{customer}, \text{customer})\}, \\ \text{recv}, (\text{send}, \text{send}), \{(\text{obj}, (\text{movA}, \text{movA}))\})), \\ (\text{academia}, \underline{\text{permitonlyparam}}(\{(\text{academia}, \text{academia})\}, \text{recv}, (\text{send}, \text{send}), \{(\text{obj}, (\text{movA}, \text{movA}))\})), \\ (\text{customer}, \underline{\text{permitonlyparam}}(\{(\text{customer}, \text{customer})\}, \text{recv}, (\text{send}, \text{send}), \{(\text{obj}, (\text{movA}, \text{movA}))\})).$$

Distribution Condition

$$\{(\text{dealer}, \underline{\text{always}}(E_{fst}((\text{sell}, \text{sell}), \{(\text{obj}, (\text{movA}, \text{movA}))\}) \Rightarrow E_{fst}((\text{pay}, \text{pay}), \{(\text{amt}, (10, \top_p)), \\ (\text{rcv}, (\text{FairMovies}, \text{FairMovies})))))).$$

Policy Modification

$$\{(\text{default}, \underline{\text{permitonlyparam}}(\{(\perp_p, \%QUALITY/2), vq, (\text{play}, \text{play}), \{(\text{obj}, (\text{movA}, \text{movA}))\}))))\}$$

Strengthening

$$\underline{\text{and}}(\underline{\text{permitonlyevname}}(\{(\perp_e, \text{edit}), (\text{send}, \text{send})\}, \{(\text{obj}, (\text{movA}, \text{movA}))\}), \\ \underline{\text{permitonlyparam}}(\{(\text{academia}, \text{academia})\}, \text{recv}, (\text{send}, \text{send}), \{(\text{obj}, (\text{movA}, \text{movA}))\})).$$

$$\underline{\text{and}}(\underline{\text{permitonlyevname}}(\{(\perp_e, \text{play})\}, \{(\text{obj}, (\text{movA}, \text{movA}))\}), \underline{\text{repmax}}(1, E_{fst}((\text{play}, \text{play}), \{(\text{obj}, (\text{movA},$$

4. Policy Entailment: Policy Evolution in Distributed Usage Control

- Policy entailment checking by a model checker, by translation to LTL (Linear Temporal Logic)
- This work/ Audit Logic:
 - Audit logic doesn't have the concept of interval (pay >\$1)
- Useful when you can enumerate how data can be used, and a bound path of distribution with nodes representing individuals or class of individuals.

Summary of techniques just covered

- Audit Logic
 - Models ownership of data
 - Introduced the general concepts of auditing, creation of logs, every actor is responsible to prove his right to perform certain actions
 - Post Obligation not clear
 - Apple
 - Similar to Audit Logic
- Recurring Theme: No clarity on how these techniques be applied on the web.
- While document can be received, modified and distributed, **concept of derivative** is introduced for the first time. (annotations of ancestral docs?)
- LTL based
 - For multimedia distribution
 - Parameterized representation of events
 - Concept of interval to capture notion of time, range of usage (preview to edit)
 - Policy sticks to the document, but **rights can be increased** in special case.

5. Data-Purpose Algebra: Modeling Data Usage Policies

- **Data restrictions imposed by the receiver**
 - This requirement is not considered by earlier works
- *Thesis*: **Restrictions on the uses** to which the data may be put are changed in ways that **can be formulated as algebraic expression**
- $\text{Item} = I(\text{content}, \text{agent}, \text{category}, \{\text{purpose}\})$, where category is the set of data items containing this item
- Discuss it for special case, I couldn't understand its generalized form.

6. Important work but DRM like

- **Laboratory for Information Security Technology**, George Mason Univ. – Prof. Ravi Sandhu
- *Usage Control Models, Architectures, and Technologies*- sponsored by NSF

Papers:

1. **The UCON_{ABC} Usage Control Model**. J. Park, R. Sandhu. ACM Transactions on Information and System Security (TISSEC), 2004
2. **A Logical Specification for Usage Control**. X. Zhang, J. Park, F. Parisi-Presicce and R. Sandhu. 9th ACM Symposium on Access Control Models and Technologies, 2004.
3. **Towards Usage Control Models: Beyond Traditional Access Control**. J. Park, R. Sandhu. 7th ACM Symposium on Access Control Models and Technologies, 2002
4. **Originator Control in Usage Control**. J. Park, R. Sandhu. POLICY 2002

Ideally

- **Negotiation** of usage policies and enforcement
- It should always be the case that $P_{\text{cons}} \subseteq P_{\text{prod}}$ or $P_{\text{req}} \subseteq P_{\text{owner}}$
- Requester and Owner should negotiate the usage policy, i.e. both owner and producer must agree on $P_{\text{agreement}} (\subseteq P_{\text{owner}} \cap P_{\text{req}})$.
- Note: this ensures that $P_{\text{req}} \supseteq P_{\text{owner}}$ then $P_{\text{agreement}} = P_{\text{req}} (\subseteq P_{\text{owner}})$. i.e. if requestor doesn't want to receive some data we pay heed.
- Best if we can always find $P_{\text{agreement}}$ s.t $P_{\text{agreement}} = P_{\text{owner}} \cap P_{\text{req}}$
- Upon agreement the owner should log (requester-id, data-id, $P_{\text{agreement}}$)
- Of-course, the requester is obligated to stick to $P_{\text{agreement}}$
- The **owner can anytime ask auditing authority to audit actions of the requester in future.**

PAW can make it possible

Fig. 1. Common Framework for POLICY

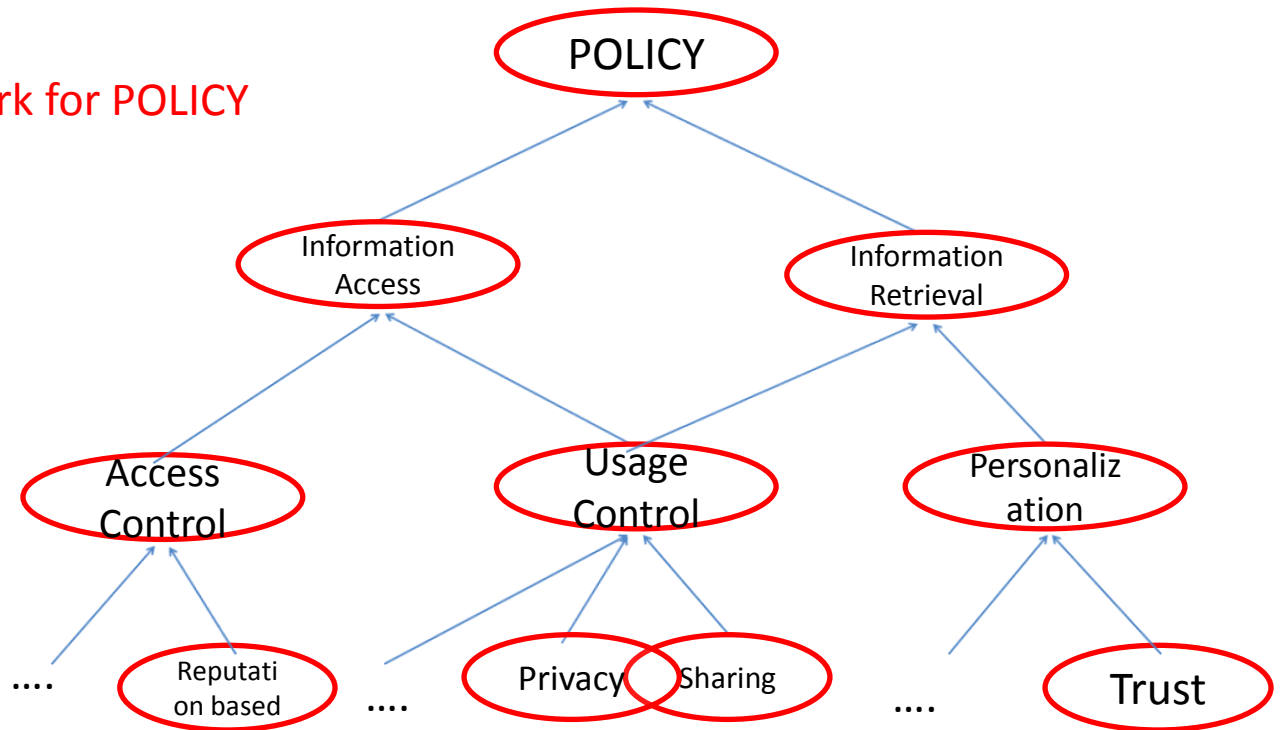


Fig. 2. RDF Quad

Property	Subject	Object	Policy
----------	---------	--------	--------

In future

1. Represent Rei, Protune semantics in AF logic of PCA.
2. Investigate definitions of ownership of SW data and usage control scenarios (starting with defining different usages).
3. Search and build usage control vocabulary (ontology). Further define an ontology-based usage policy language.

THANKS !

4. Investigate techniques of usage control discussed in the presentation in light of SW domain.
5. Description of “Policy aware SPARQL”- an extension to SPARQL that takes bunch of policies as input and returns query results that are compliant.
 - a. filter out disallowed-triples from query results.
 - b. consider `<input-graph \ disallowed-triples>` for querying.