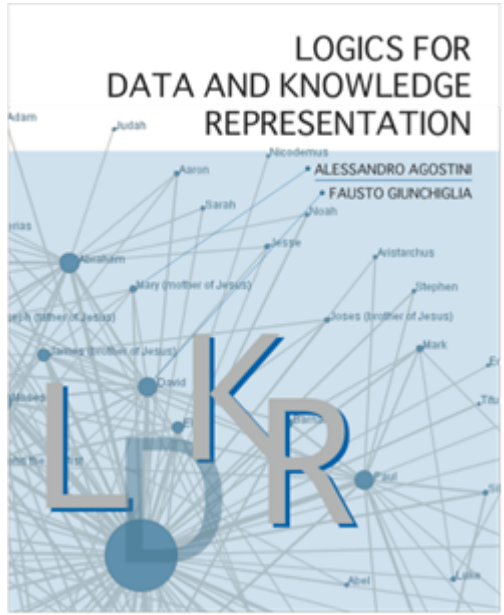


Logics for Data and Knowledge Representation

Application of DLs: RelBAC

Outline

- New Challenges for Access Control
- Model and Logic
- Automated Reasoning
  - Reasoning tasks
  - SoD

New Challenges

- Objects
  - Various scales: eBusiness, eScience
  - Various types: Blogs, Wiki, Flickr, Youtube
- Subjects
  - Social network explosion: MySpace, Facebook
- Permissions
  - Context: Pervasive Computing

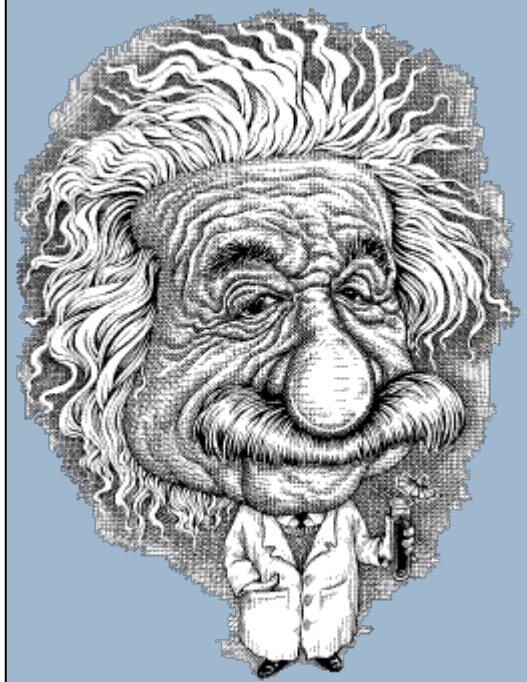
Dynamic Permissions

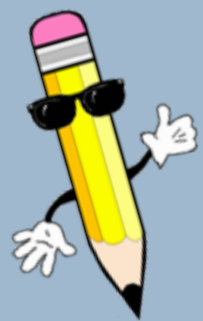
- Time
  - Access time, duration, frequency, etc.
- Location
  - Physical address
- System
  - System condition such as load, connection number, priority, etc.

State of the Art

Right	Pencil	Pen
Einstein	Use	-Use

- Request
- Access
- Use



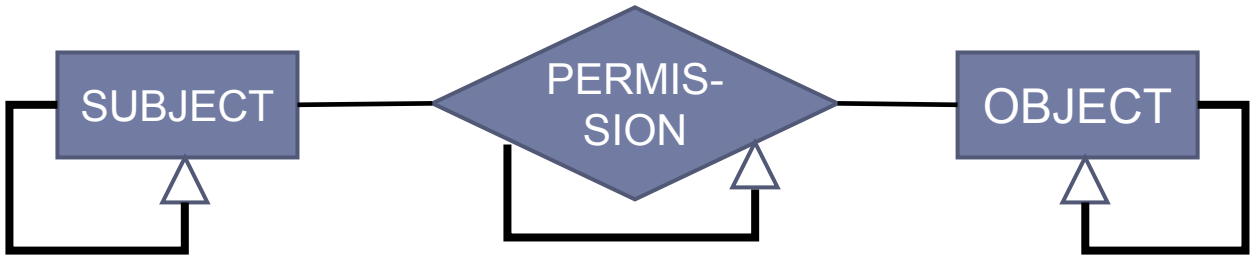


- AC Models
  - AM
  - ACL
    - MAC, DAC
  - RBAC
  - TBAC
- Formalisms
  - Non-logical
  - Logical

Motivations

- Natural
  - Friendly to ordinary user
  - Automated tools for management
- Flexible
  - Coverage of various domains
  - Extensible for new requests
- Formal
  - Compact syntax and semantics
  - Security Analysis

RelBAC Model



- SUBJECT: Anna, Bob, Client 001, Friends, …
- OBJECT: File, Email, Picture, Music, Video, Tags, …
- PERMISSION: Read, Upload, Correct, Remove, …

Logic Language

- ALCQIb
  - ALC = AL with full concept negation
  - Q = Qualified number restrictions
  - I = inverse properties
  - b = safe boolean role expressions

ER Model	DL Formalization
SUBJECT	Concept
OJBECT	Concept
PERMISSION	Role
PARTIAL ORDER	Subsumption
RULE	Subsumption *

* a RelBAC rule may take the form of equality, but seldom used.

The partial order

$A_1 \geq A_2$	iff	$A_1 \sqsubseteq A_2$
$U_1 \geq U_2$	iff	$U_1 \sqsubseteq U_2$
$O_1 \geq O_2$	iff	$O_1 \sqsubseteq O_2$
$P_1 \geq P_2$	iff	$P_1 \sqsubseteq P_2$

SUBJECT HIERARCHY: Coder $\sqsubseteq$ KnowDive

OBJECT HIERARCHY: Video $\sqsubseteq$ Entertainment

PERMISSION HIERARCHY: Write $\sqsubseteq$ Read

Access Control Rules

- Three kinds of axioms

$C \equiv D$	$C \sqsubseteq D$	$C \sqsupseteq D$

- General Access Control Rules

$U \sqsubseteq \exists P.O$	(1)	$U \sqsubseteq \geq n\ P.O$	(5)
$O \sqsubseteq \exists P^{-1}.U$	(2)	$O \sqsubseteq \geq n\ P^{-1}.U$	(6)
$U \sqsubseteq \forall P.O$	(3)	$U \sqsubseteq \leq n\ P.O$	(7)
$O \sqsubseteq \forall P^{-1}.U$	(4)	$O \sqsubseteq \leq n\ P^{-1}.U$	(8)

- User-centric vs. Object-centric rules

Access Control Rules: example

Policy	RelBAC Representation
All friends can download some music	$\text{Friend} \sqsubseteq \exists \text{Download}.\text{Music}$
Music can be downloaded by some friend	$\text{Music} \sqsubseteq \exists \text{Download}^{-1}.\text{Friend}$
All friends can download only music	$\text{Friend} \sqsubseteq \exists \text{Download}.\text{Music}$
Music can be downloaded by only friend	$\text{Music} \sqsubseteq \exists \text{Download}^{-1}.\text{Friend}$
KnowDive members should program at least one project code	$\text{KnowDive} \sqsubseteq \geq 1\ \text{Program}.\text{Code}$
Each project code should be programmed by at most 2 KnowDive members	$\text{Code} \sqsubseteq \leq 2\ \text{Program}^{-1}.\text{KnowDive}$
Each manager should manage exactly 3 project codes	$\text{Manager} \sqsubseteq \leq 3\ \text{Manage}.\text{Code} \sqcap \geq 3\ \text{Manage}.\text{Code}$

TAC (Total Access Control) Rule

- All to all mapping

$$\{P(u_1,o_1),\ldots,P(u_m,o_1),\ldots,P(u_m,o_n)\}$$

$$\exists O.P \equiv \forall \neg P.\ \neg O$$

$$\begin{aligned} (\exists O.P)^I &= \{u \in User^I \mid \forall o\ O(o) \rightarrow P(u,o)\} \\ &= \{u \in User^I \mid \forall o\ \neg P(u,o) \rightarrow \neg O(o)\} \\ &= (\forall \neg P.\ \neg O)^I \end{aligned}$$

"Close friends can read all the entertainment files."
$Close \sqsubseteq \exists Entertain.Read$

Correspondences to Motivations

- Natural
  - permission ⇨ binary relation
  - partial order ⇨ subsumption axiom
  - rule ⇨ formula(e)
- Flexible
  - hierarchy ⇨ partial order
  - attribute ⇨ binary relation
- Formal
  - domain specific description logics

Reasoning Services

- TBox

  ‘A business friend can update some entries.’

- ABox

  ‘Bob is a business friend.’

- ABox + TBox

  ‘Bob is a business friend so that he can update some entries.’

- Design vs. Run time Reasoning

Reasoning Tasks: Design

- Hierarchy

  IPod $\sqsubseteq$ DigitalDevice

- Membership

  DigitalDevice(ipod-2g0903)

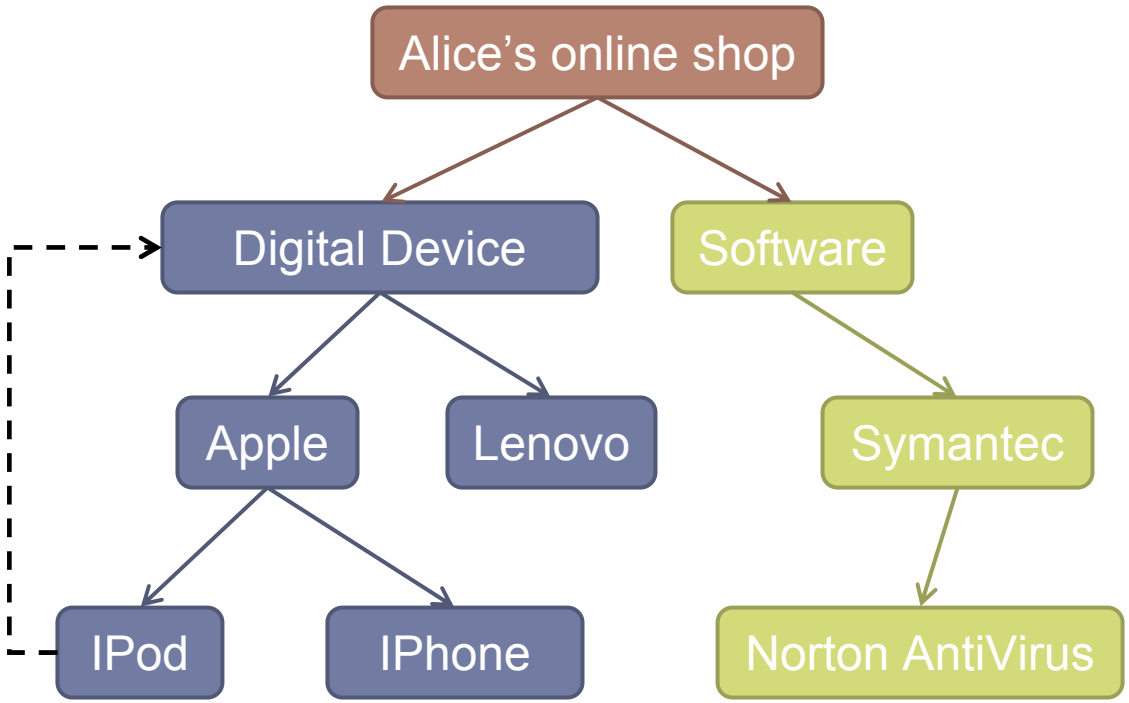
- Separation of duties

  ‘customer and sales manager are to be separated.’

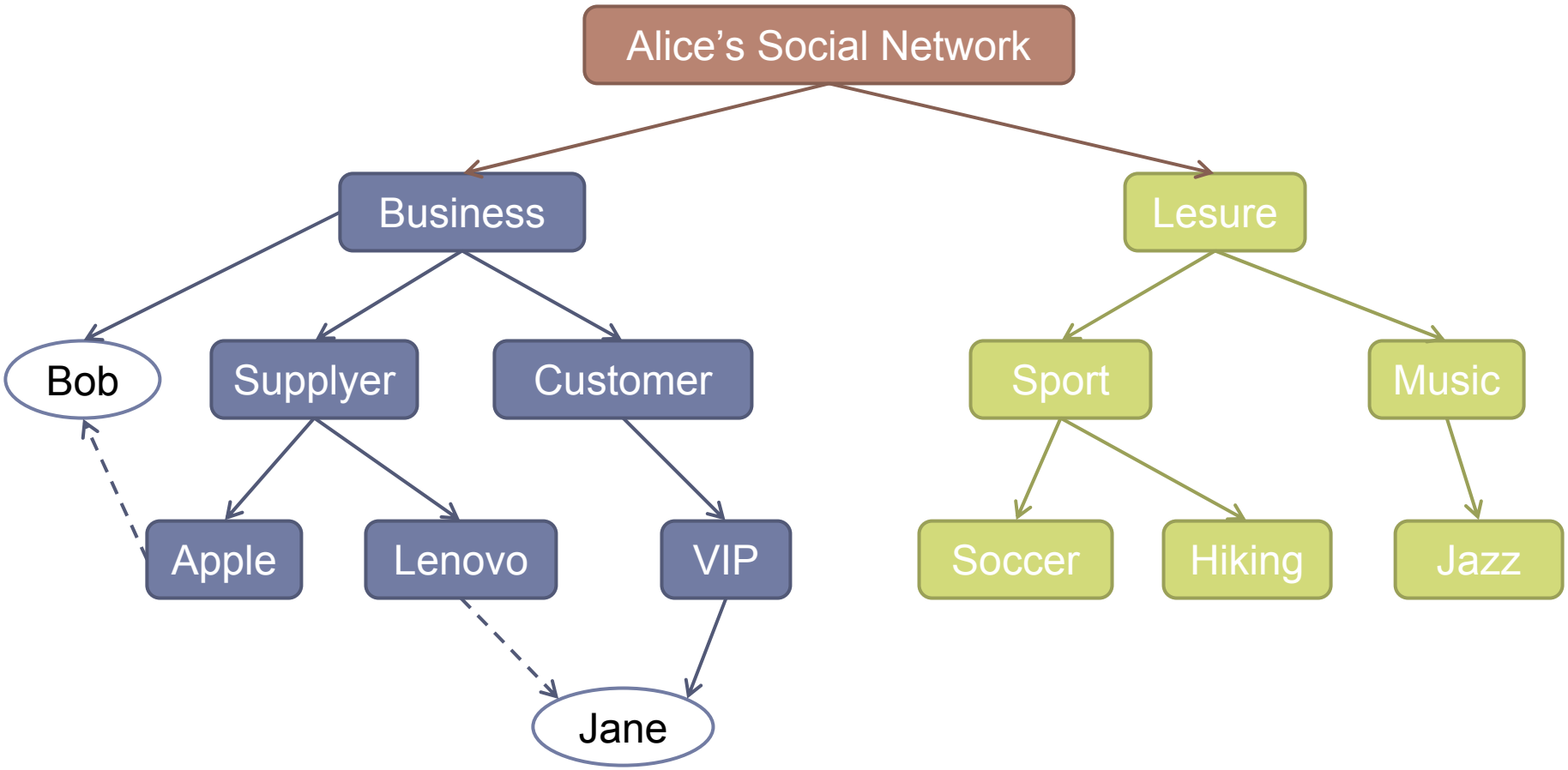
- High-level Concern

  ‘the 3 users to commit an order should include 1 customer, 1 sales agent and 1 sales manager.’

Design Time Reasoning: Hierarchy



Design Time Reasoning: Membership



Separation of Duties (from RBAC)

- 'For a task consisting of *n* steps, no one can complete all the steps to complete the task.'

$$\sqcap_{i=1}^{n} \Box Pi.Oi \sqsubseteq \bot$$

- '…no one can complete more than one of the steps.'

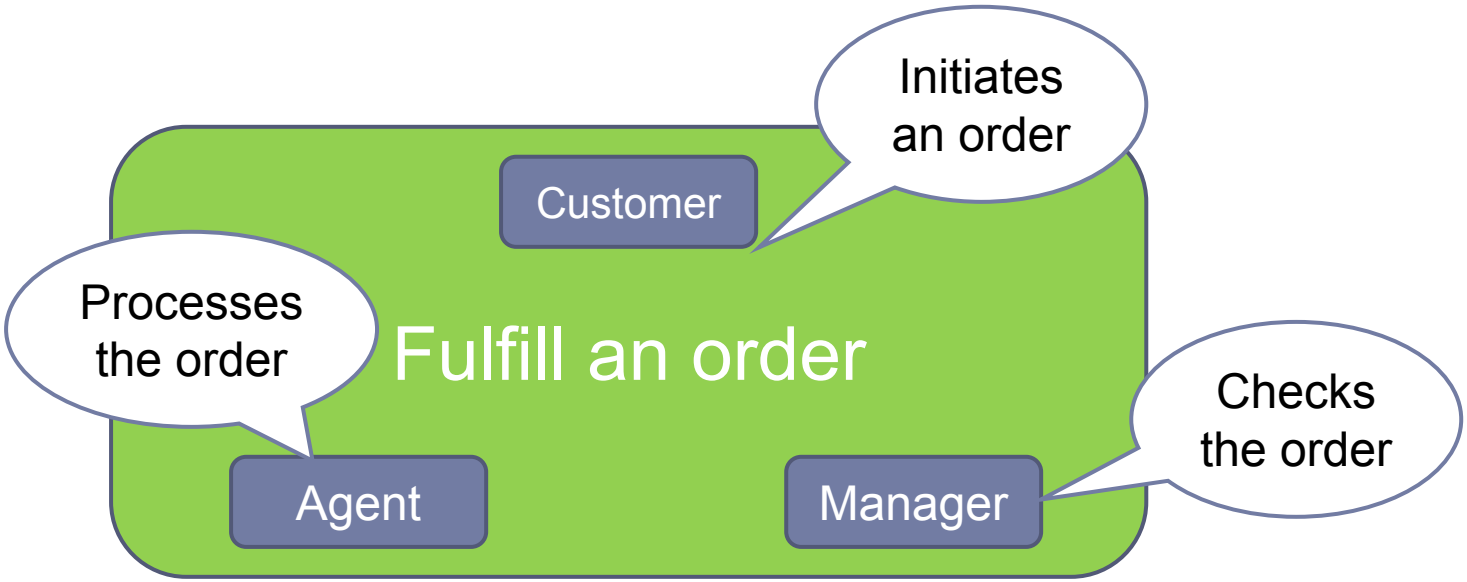
$$\Box Pi.Oi \sqcap \Box Pi.Oj \sqsubseteq \bot \quad 1 \leq i < j \leq n$$

'To cash out a check, a check has to be signed by a customer and cashed out by a clear (in a bank).'

$$\exists Sign.Check \sqcap \exists Cashout.Check \sqsubseteq \bot$$

Separation of Duties: High-level Concern

- Composition of the k users



- Order $\sqsubseteq \geq 1\ \text{Initiate}^{-1}.\text{Customer} \sqcup \geq 1\ \text{Process}^{-1}.\text{Agent} \sqcup \geq 1\ \text{Check}^{-1}.\text{Manager}$