The Science of Security

Introduction from the Perspective of Secure Collaboration

Munindar P. Singh
singh@ncsu.edu

Department of Computer Science
North Carolina State University

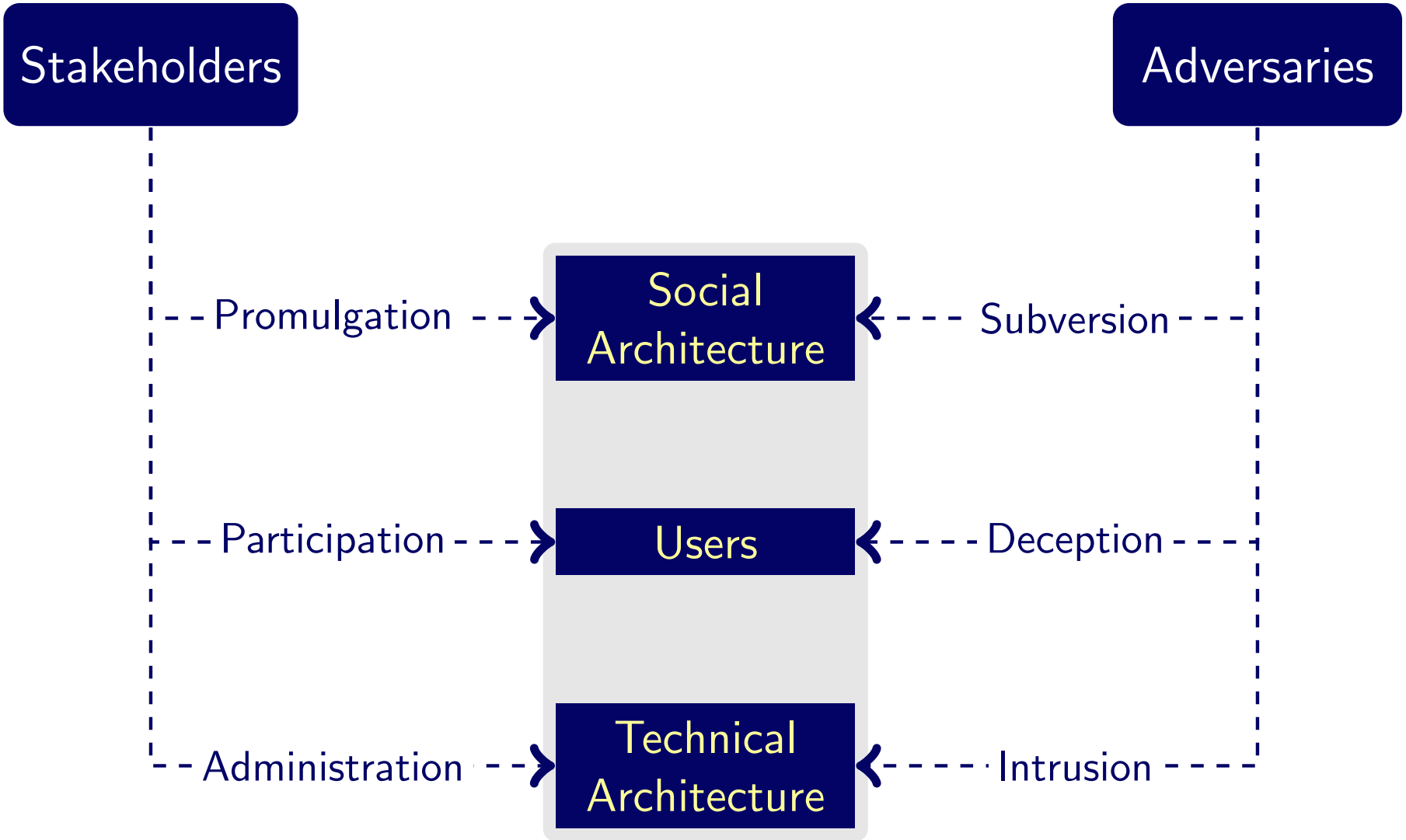
Science of Security

Developing a scientific foundation for security

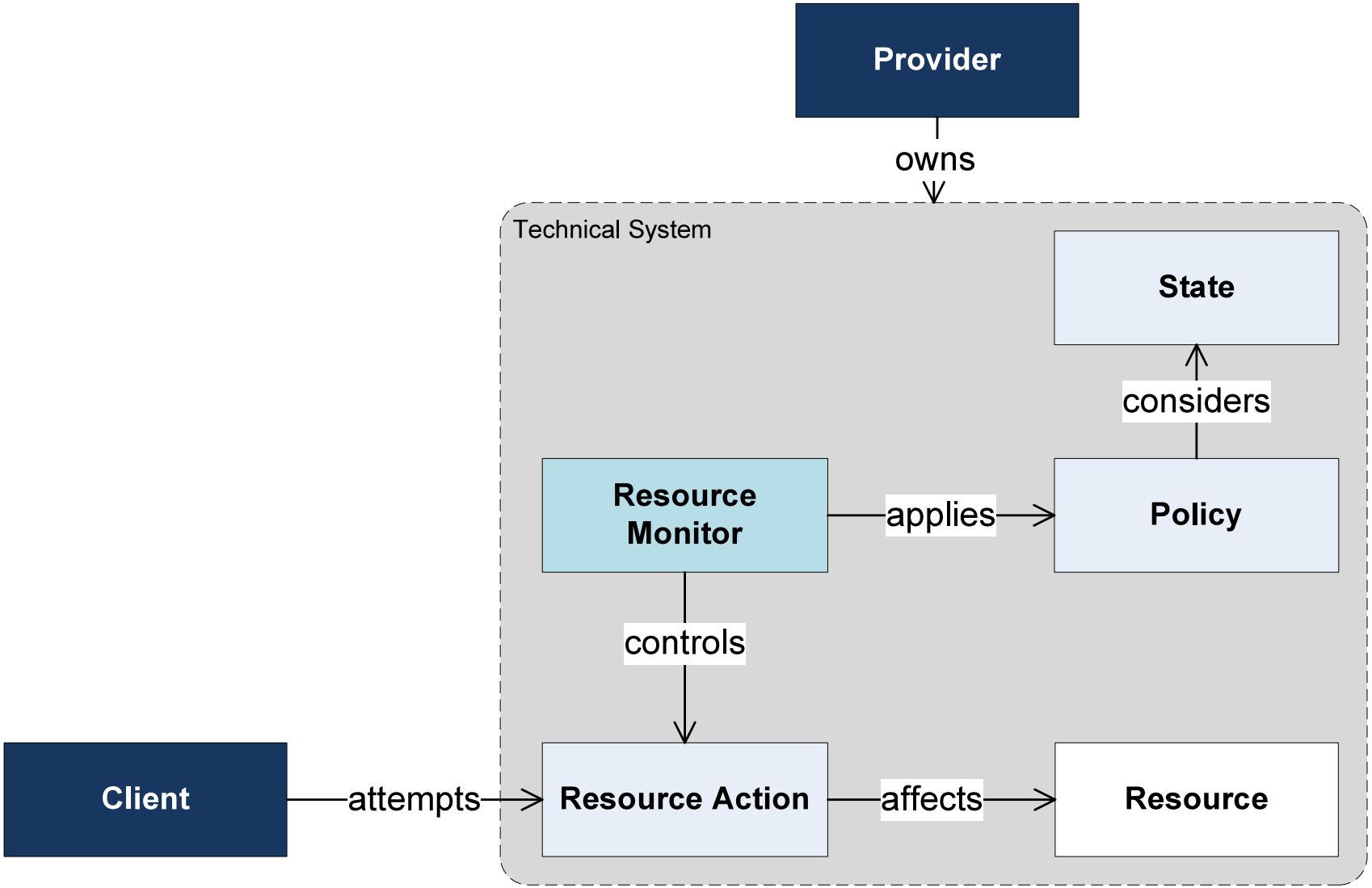
- Existing security approaches are ad hoc
  - Poorly characterized
  - Not predictive
  - Find a bug, patch it, find another bug...
  - "Secure" systems suffer from all manner of disaster
- Science
  - Theoretical principles
    - Mathematical, quantifiable, ...
  - Empirically grounded
  - Reproducible and repeatable
  - Verifiable, falsifiable, strongly inferable hypotheses
- SoS: Let's think of new principles of security
  - Pertaining to the participants, artifacts, and their interactions
    - Adopt ideas from physical and social sciences
    - Design experiments

Participants and Artifacts in Security

Greatest challenges arise in the upper two; most past effort is on technical architecture

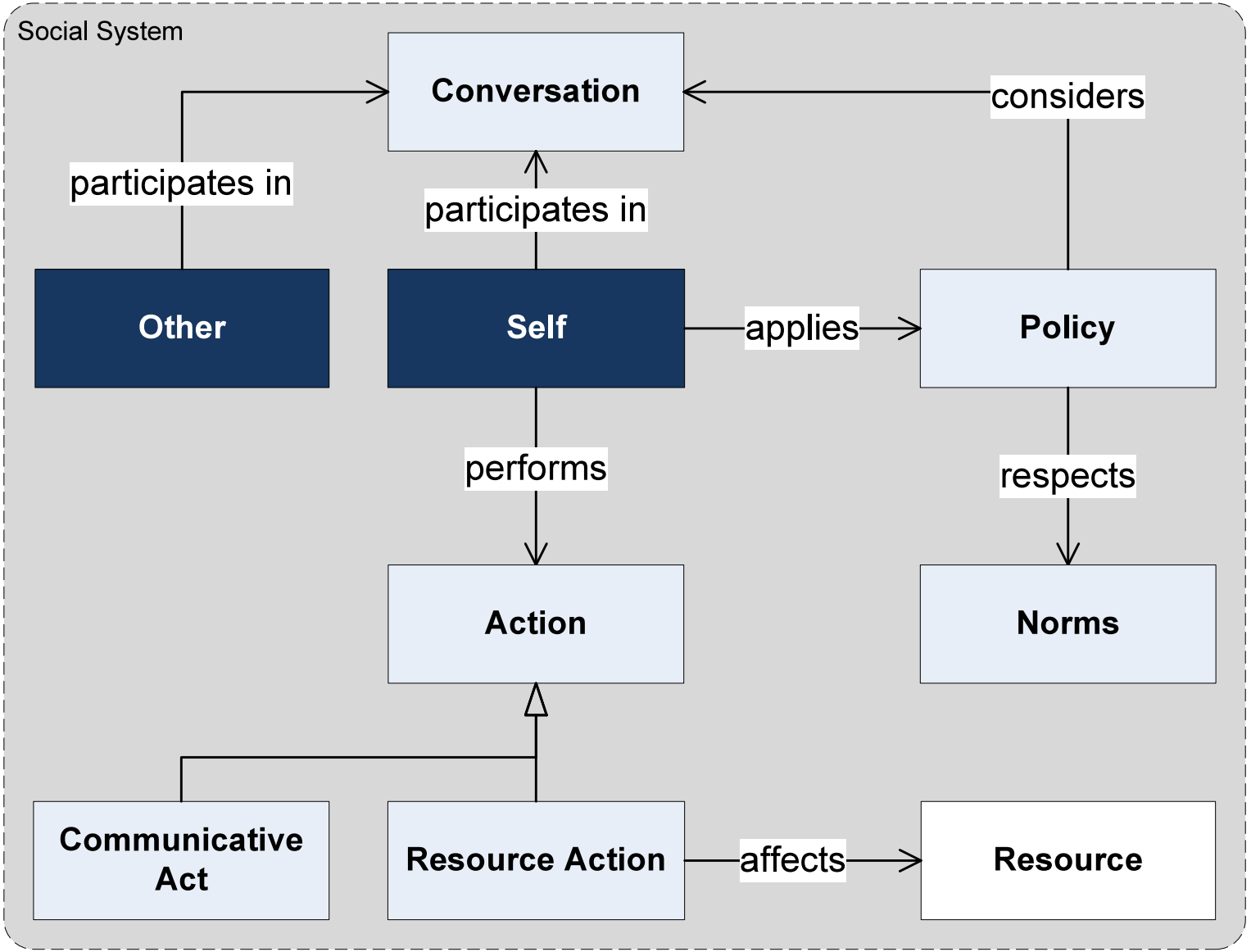


Traditional View: Systems as Artifacts



Proposed View: Systems as Societies

Conversations with autonomous parties; control over resources

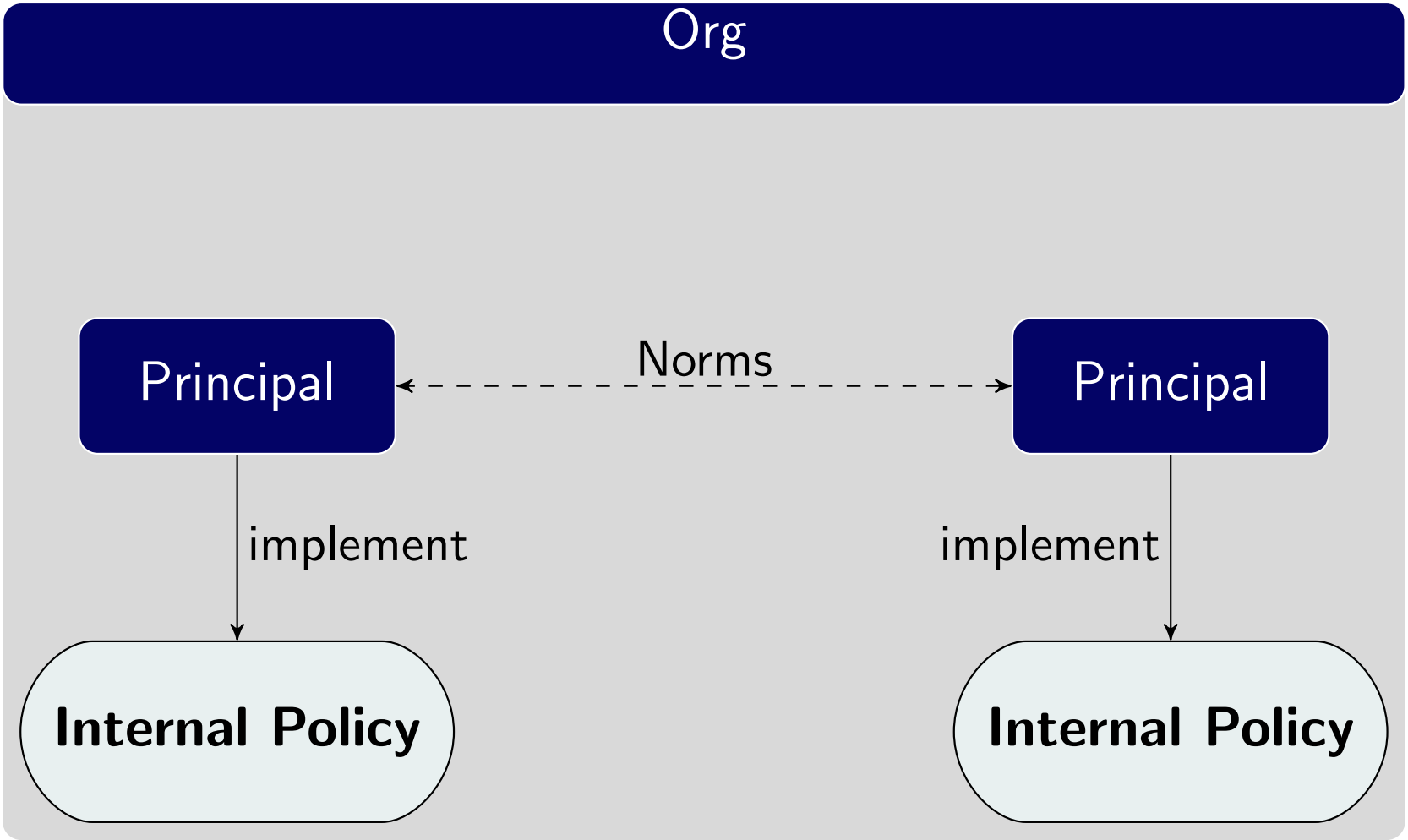


Sociotechnical Systems

Combine IT with real-life societal considerations

- System characteristics
  - Longevity and identity
  - Autonomy
  - Essentially a society
  - Characterized via norms, not operationally
- Member characteristics
  - Longevity and identity
  - Autonomy
  - Heterogeneity
  - Ability to deal with norms, e.g., via goals realized in policies
- Realization
  - Top down: Members fit into existing system
    - Adopt suitable goals given system norms
  - Bottom up: Members design new system
    - Negotiate suitable norms given individual goals

Simple Normative Framework for Sociotechnical Systems



Regulation versus Regimentation

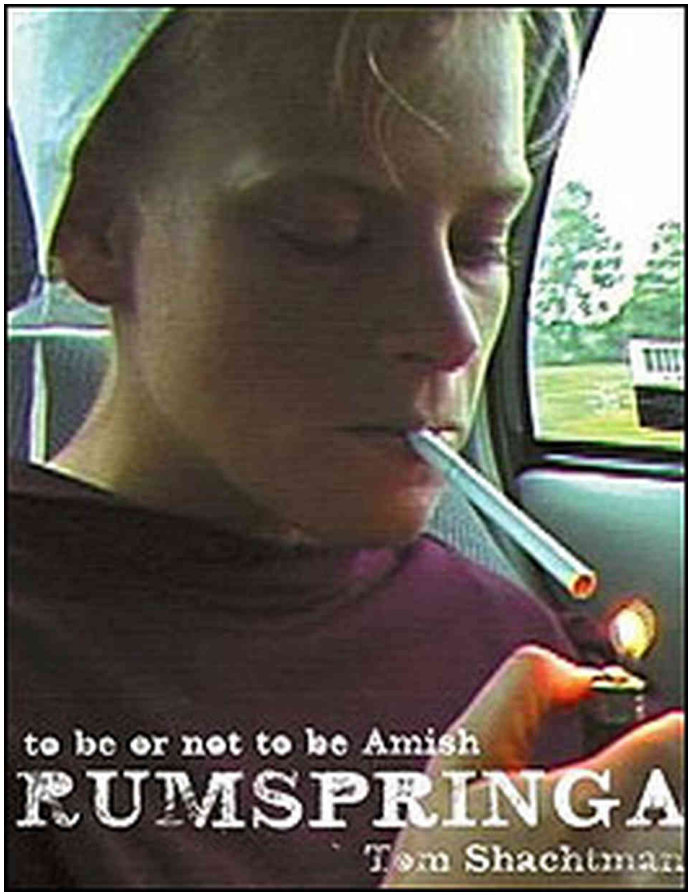
Amish Rumspringa

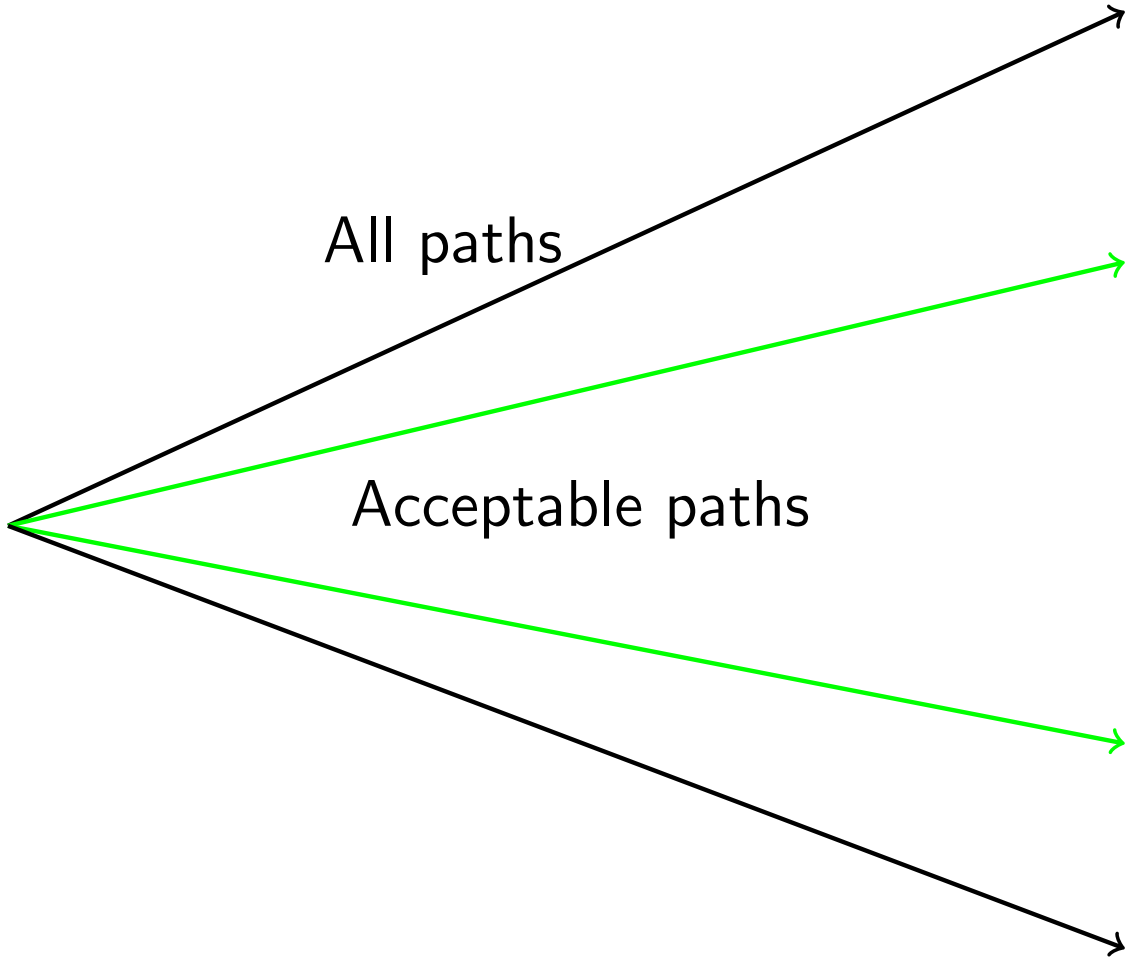
- Regimentation: preventing bad behavior
  - Fits a closed system
  - Reflects a pessimistic stance
  - Presumes a regimenting infrastructure
- Regulation: discouraging and correcting—though *allowing*—bad behavior
  - Fits an open system
  - Reflects an optimistic stance
  - Presumes a regulating social system

Regulation versus Regimentation: Amish Rumspringa

Autonomy: Technical architecture allows bad behavior; social architecture discourages it

Crucial for innovation





http://media.npr.org/books/images/2006/rumspringa200-d4edb2697bb547c7c12c73e2a7058289ce374ac9-s6-c30.jpg

Violating a Norm

Benefit of regulation over regimentation: Sometimes you just gotta break the rules

Violating a Norm

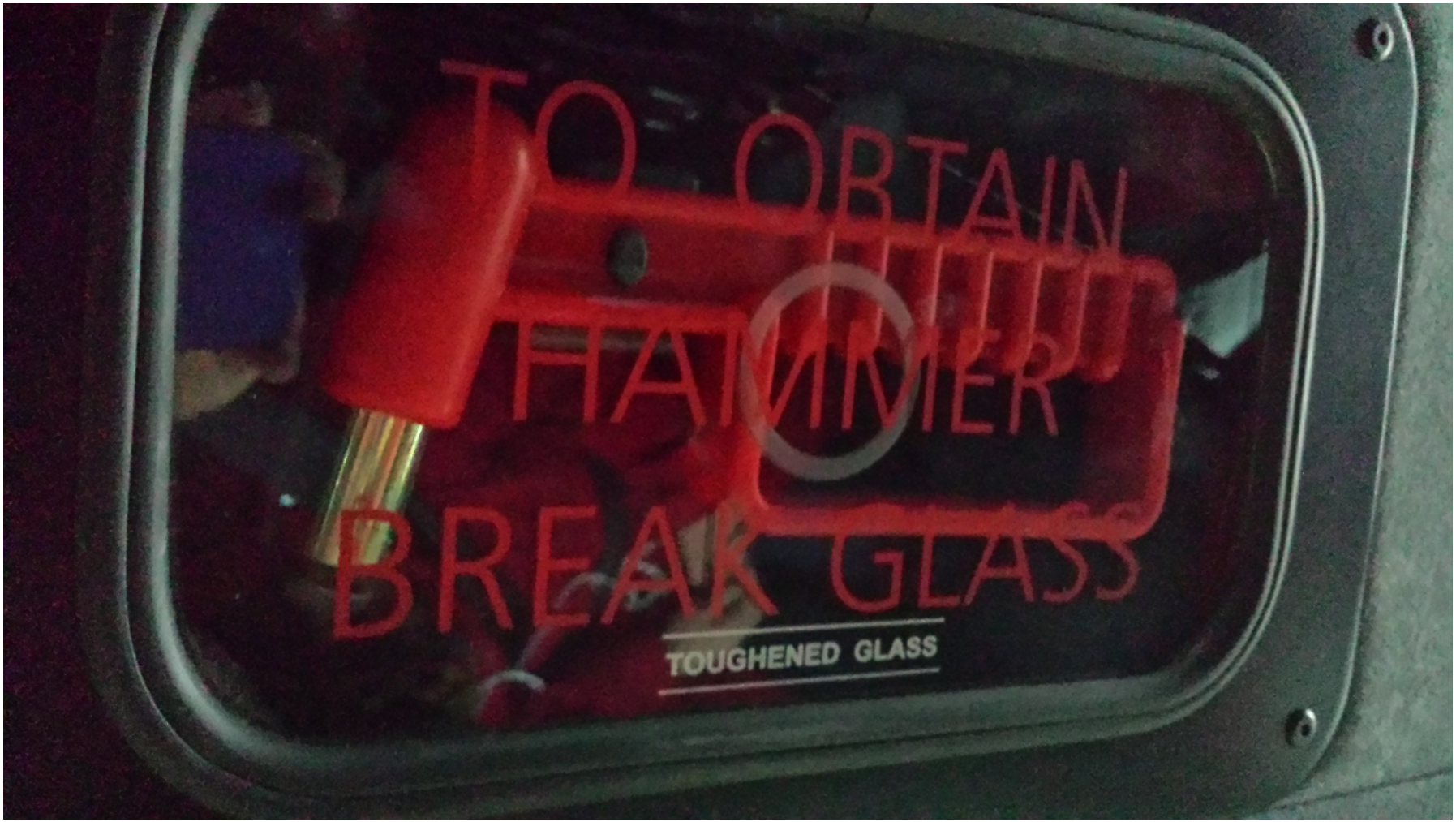
Maybe violations can be overdone? (This is the Farnham Road Hospital after all)

Authorization to Violate a Norm

Norms about norms

Poorly Designed Norms?

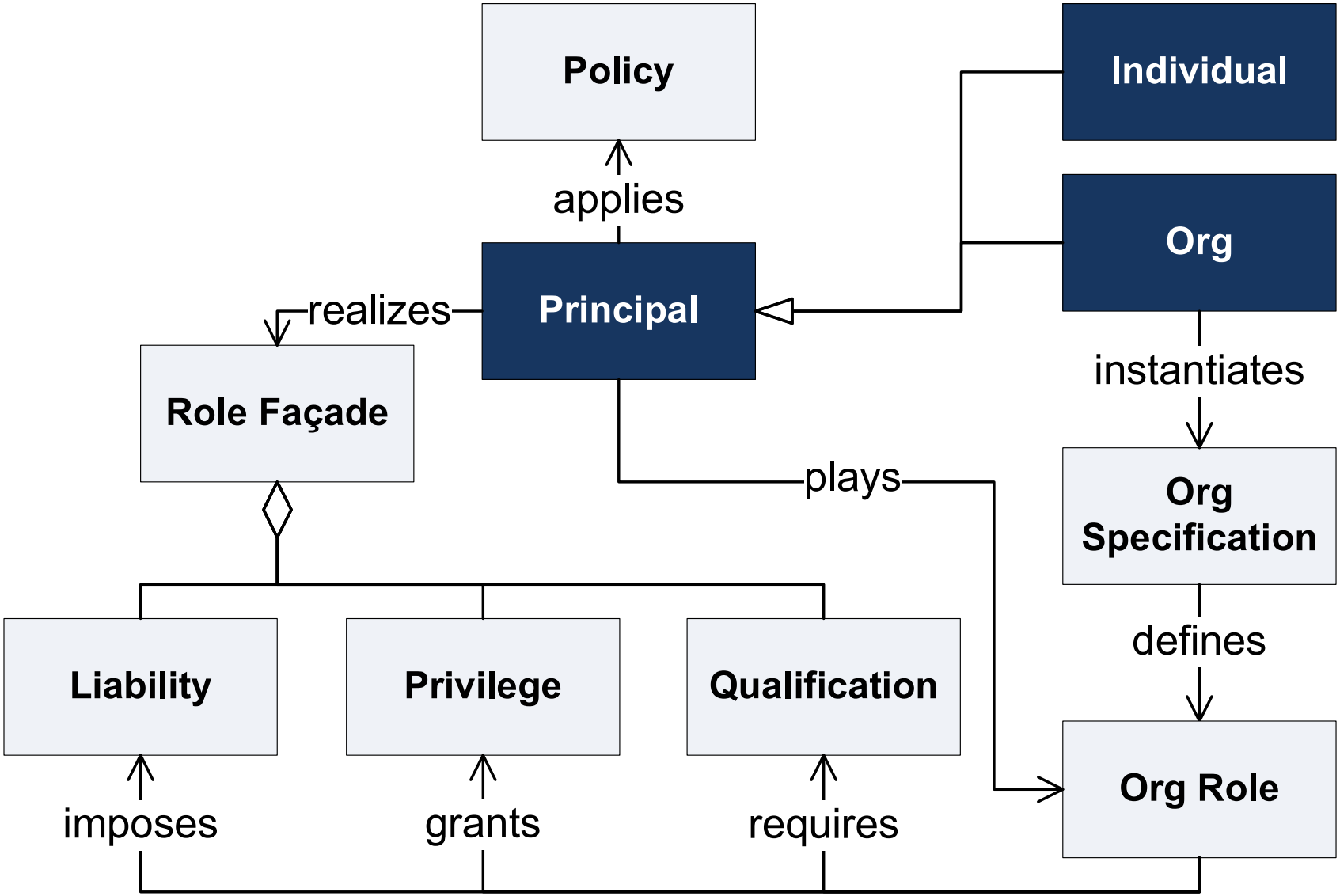
From the same Heathrow bus



Conception of Norms, Orgs, and Policies

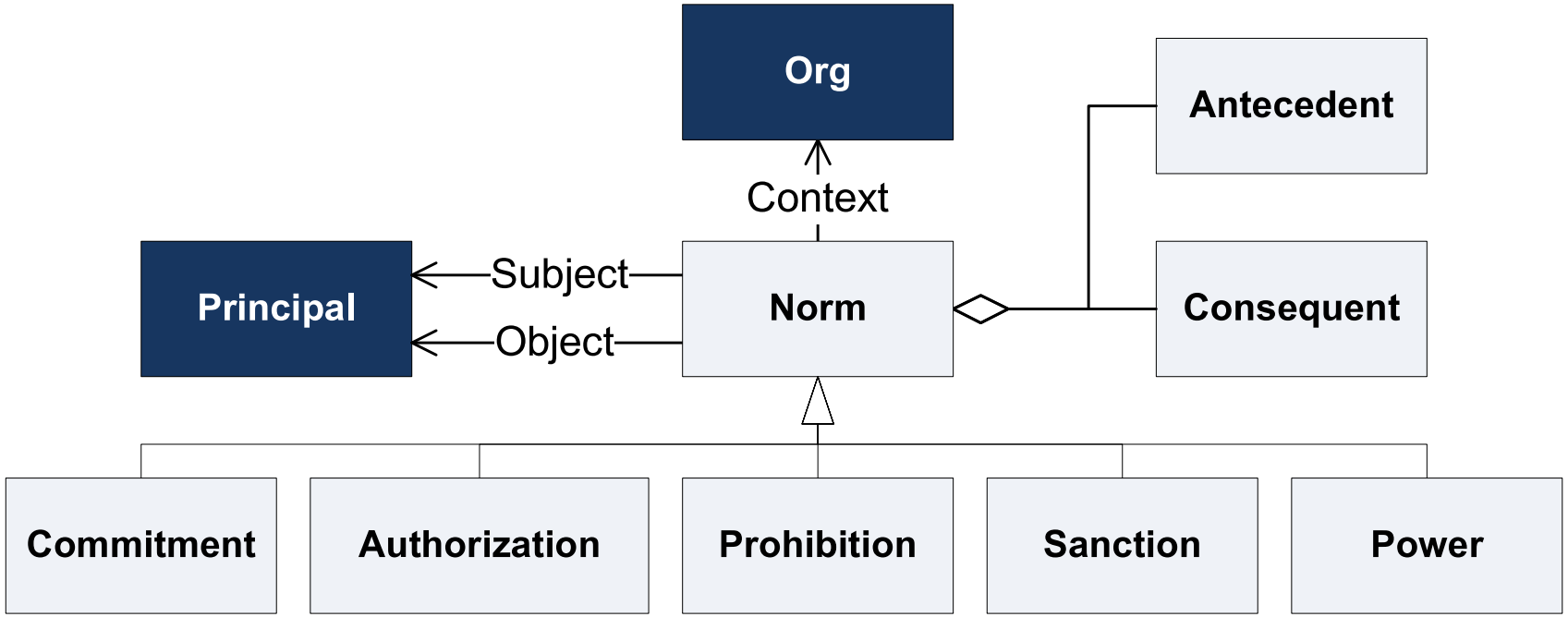
- Key concepts
  - Orgs host norms and members
  - Norms as standards of correctness
  - Internal policies of agents address norms
  - Decision making and behavior of agents address policies
- Societal structure relates to other important concepts
  - Trust
    - Engendered by norms
    - Assigned based on policies
  - Economic concepts
    - Incentives correspond to policies
    - Mechanisms correspond to norms

Governance Overview



Types of Norms

Unified logical form: Norm(subject, object, context, antecedent, consequent)

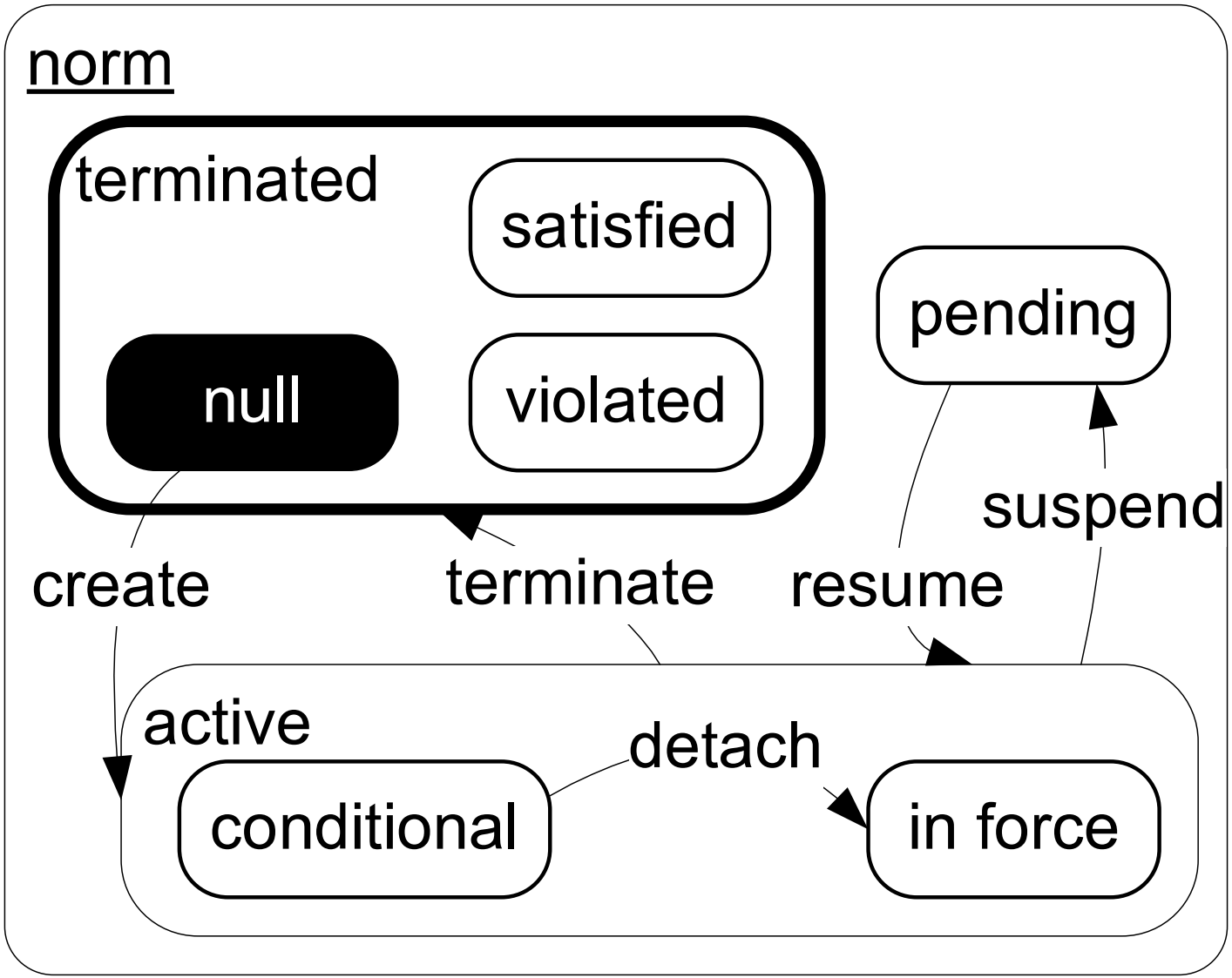


- Directed: capture accountability
- Declarative
- Composable
- Manipulable

Norms as Façades

Norm	**Subject's Façade**	**Object's Façade**
Commitment	Liability	Privilege
Authorization	Privilege	Liability
Power	Privilege	Liability
Prohibition	Liability	Privilege
Sanction	Liability	Privilege

Norm Life Cycle: 1



Norm Life Cycle: 2

Substate of a terminated norm

If terminated in		Then				
ant	**con**	**Com**	**Aut**	**Pro**	**San**	**Pow**
false	false	null	null	null	null	null
false	true	sat	vio	null	null	null
true	false	vio	null	sat	null	vio
true	true	sat	sat	vio	sat	sat

Architecture

Differentiating from traditional software architecture

- Autonomy is key
  - Partly recognized in ULSSIS: Ultra-Large-Scale Software-Intensive Systems
- Abstraction: norms describe what, not how
- Opacity: internal policies are hidden
- Dynamism, both
  - Membership
  - Participation is not regimented
- Fractal structure of Orgs
  - Turtles all the way

Conceptions of Accountability

- Traceability: when certain actions can be traced to the accountable party
  - Unnecessary: Alice is a bully and openly commits infractions
  - Insufficient: Alice gets Bob to submit a form for her
- Deterrence: when certain actions yield a negative utility for the accountable party
  - Deterrence simply suggests a more complex norm "N or else penalty" but doing so, voids any accountability
  - When deterrence is nonzero, it serves as sanctioning—after the fact
  - Even when deterrence is zero, the accountability remains
- Proposed normative formulation
  - A party is accountable to another party when the second party has standing to expect certain behavior from the first party
- Autonomy and accountability are two faces of the same coin
  - For any principal: No accountability without autonomy
  - For any society: No autonomy without accountability

Security Properties and Threats

To use as demonstration cases

- Properties
  - Least privilege
  - Separation of duties
  - Two-person rule (e.g., for nuclear missile launch)
- Threats
  - Denial of service
  - Information inference
  - Insider attacks

Challenge: Specification

- Framework
  - Operational model (aka "system spec")
    - Computable
    - Mathematical and abstract
    - Provides the underpinnings for correctness
  - Correctness (aka "property spec")
    - To be verified
    - Expressed on top of the operational model
- Specification modalities
  - Policies
  - Incentives
  - Sanctioning
  - Normative relationships

Challenge: Architectural Patterns and Properties

Parametric families of systems

- Examples of architectural patterns
  - Make at least one party accountable for each requirement
  - Make exactly one party accountable for each requirement
  - Ensure each Org controls its infrastructure
  - Ensure each Org provides identity for its members
- Examples of properties
  - The information inference vulnerability is avoided
  - Certain actions cannot be performed unless two agents agree

Challenge: Robustness

Guarantees of system states reached

- Under combinations of threats, e.g.,
  - Faults
  - Attacks
  - Specific agent policies
  - Collusion
- From the perspective of
  - Specific agents or roles
  - Org
  - External party, where relevant (?)
- In the context of
  - Particular infrastructure
  - Orgs

Challenge: Toward a Type Theory

Foundation for design of normative systems

- Explore well-known concepts in the present setting
  - Refinement of norms by norms
  - Realization of norms by role specifications
  - Conformance of roles to roles
  - Alignment of agents
  - Interoperability of roles
- Example fundamental theorem
  - Substituting a role by a conformant role preserves interoperability

Challenge: Requirements Engineering

- Designing an Org
  - Capturing requirements
  - Validating norms with requirements
- Multiparty design
  - Argumentation
  - Capturing design rationale
  - Evolution
  - Incorporating evidence

Highlights

- To understand security presumes
  - Autonomy and accountability
  - Standards of acceptable behavior
- A system as a society
  - Regulation, not regimentation
  - Orgs help delineate the social context
- A normative architecture
  - Dynamism
  - Support for incentives
  - Doesn't regiment interactions: members can violate norms
- Raising the abstraction level opens up additional possibilities
  - Mapping personal norms (psychology)
  - Organizational culture (social psychology)

Thanks!

Amit Chopra and Science of Security Lablet colleagues

http://www.csc.ncsu.edu/faculty/mpsingh/