Relational Access Control with Bivalent Permissions in a Social Web/ Collaboration Architecture

Todd Davies and Mike D. Mintz
Symbolic Systems Program
Stanford University
http://deme.stanford.edu

PIECE

deme

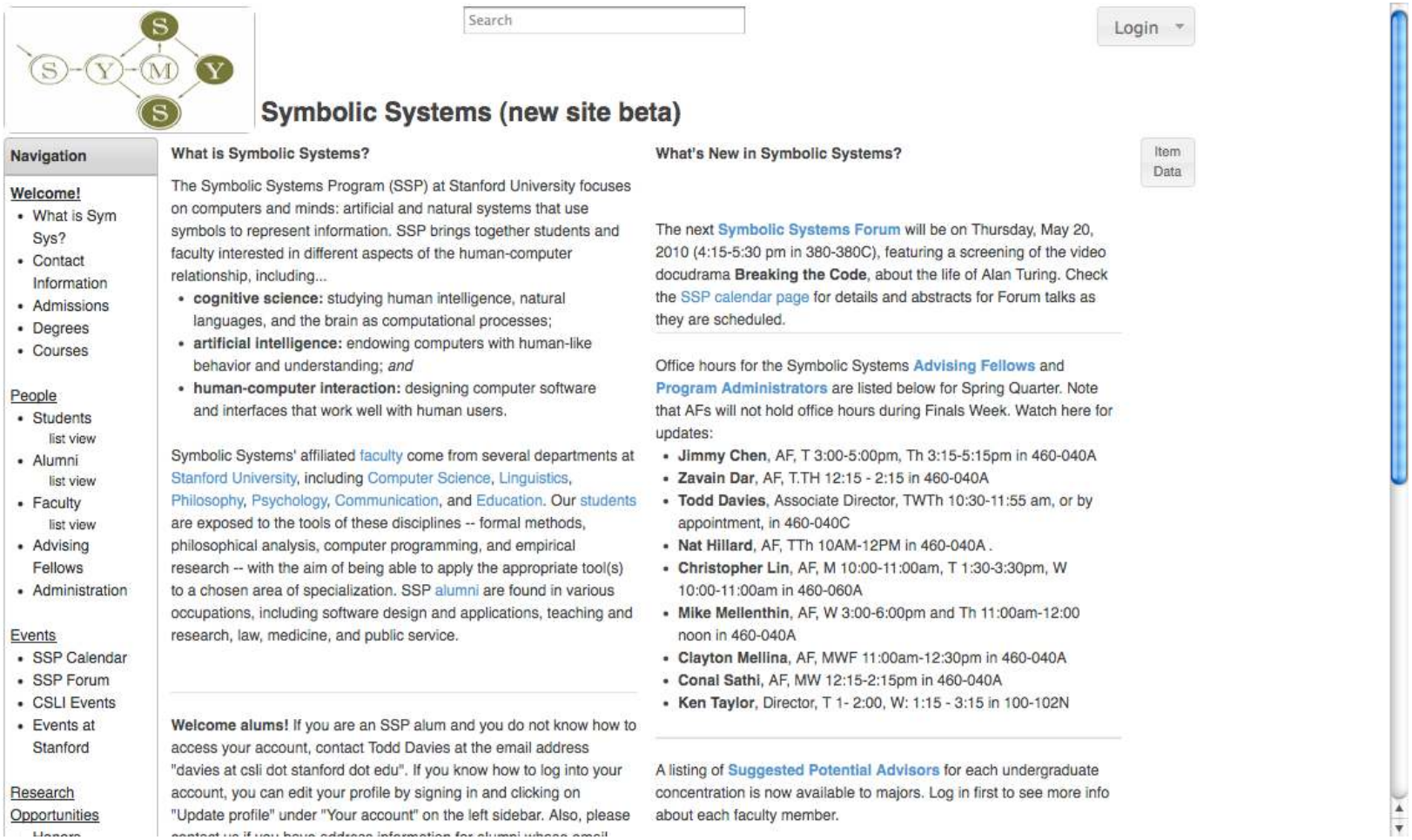
This paper is about

access control.

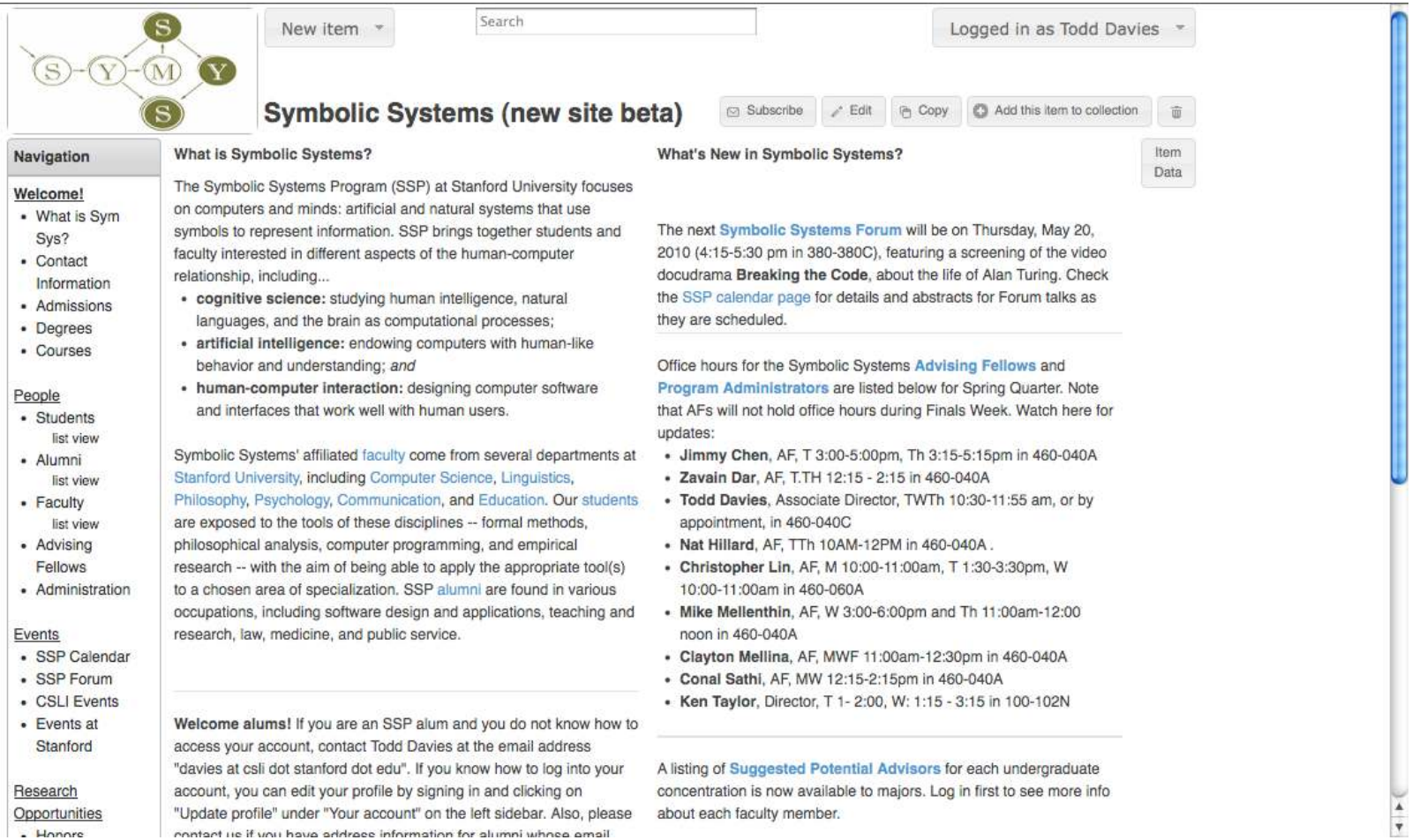
But we are not

specialists in access control research.

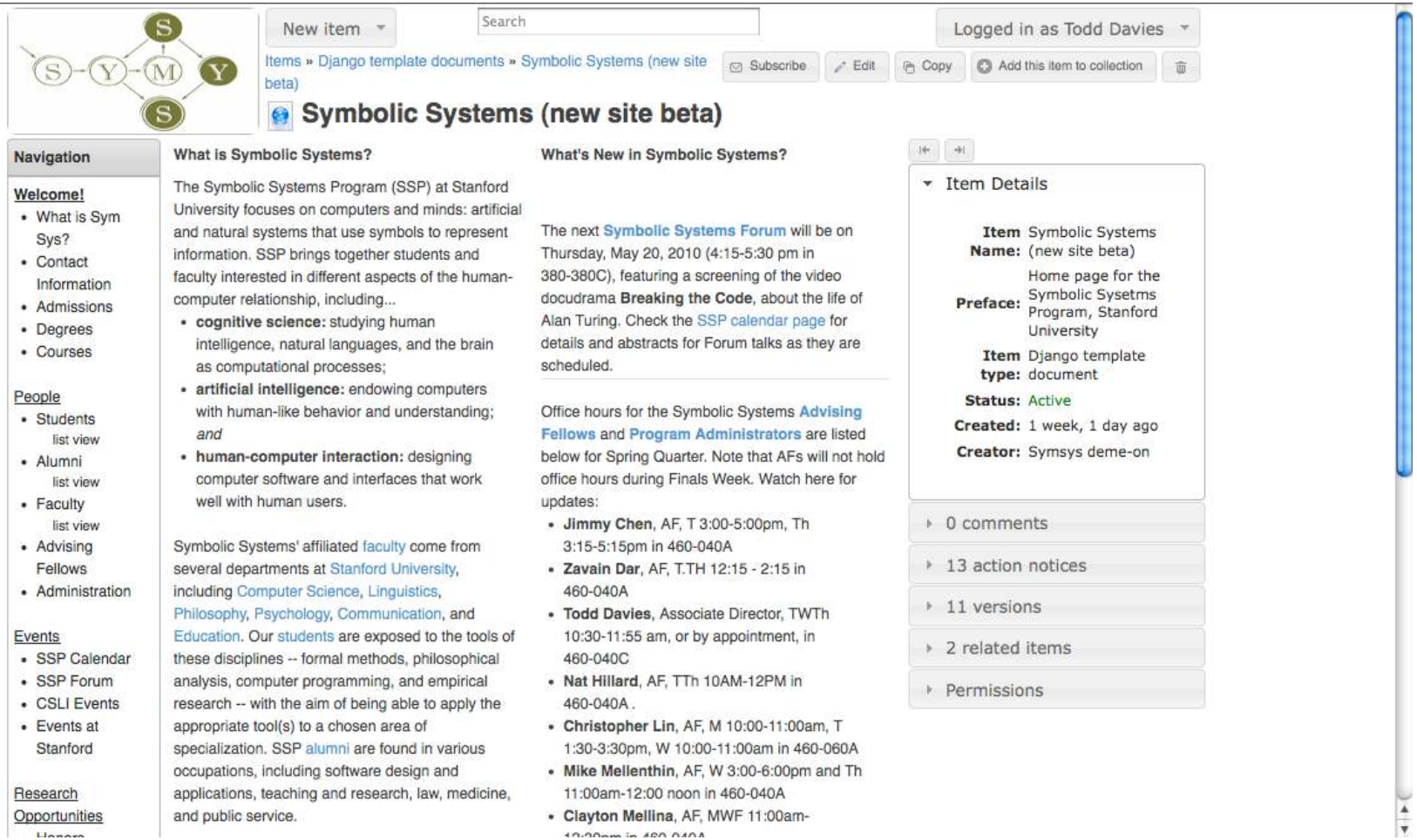
Deme with Anonymous user



Deme with logged in user



Deme with item data



Deme aims to mirror the structure of real world groups.

Deme…

aims to merge

collaborative production, document-centered discussion, and group decision making

with

content management, social networking, data sharing and portability, and user control

Deme's technical orientation

End-user OOP/extensible content management

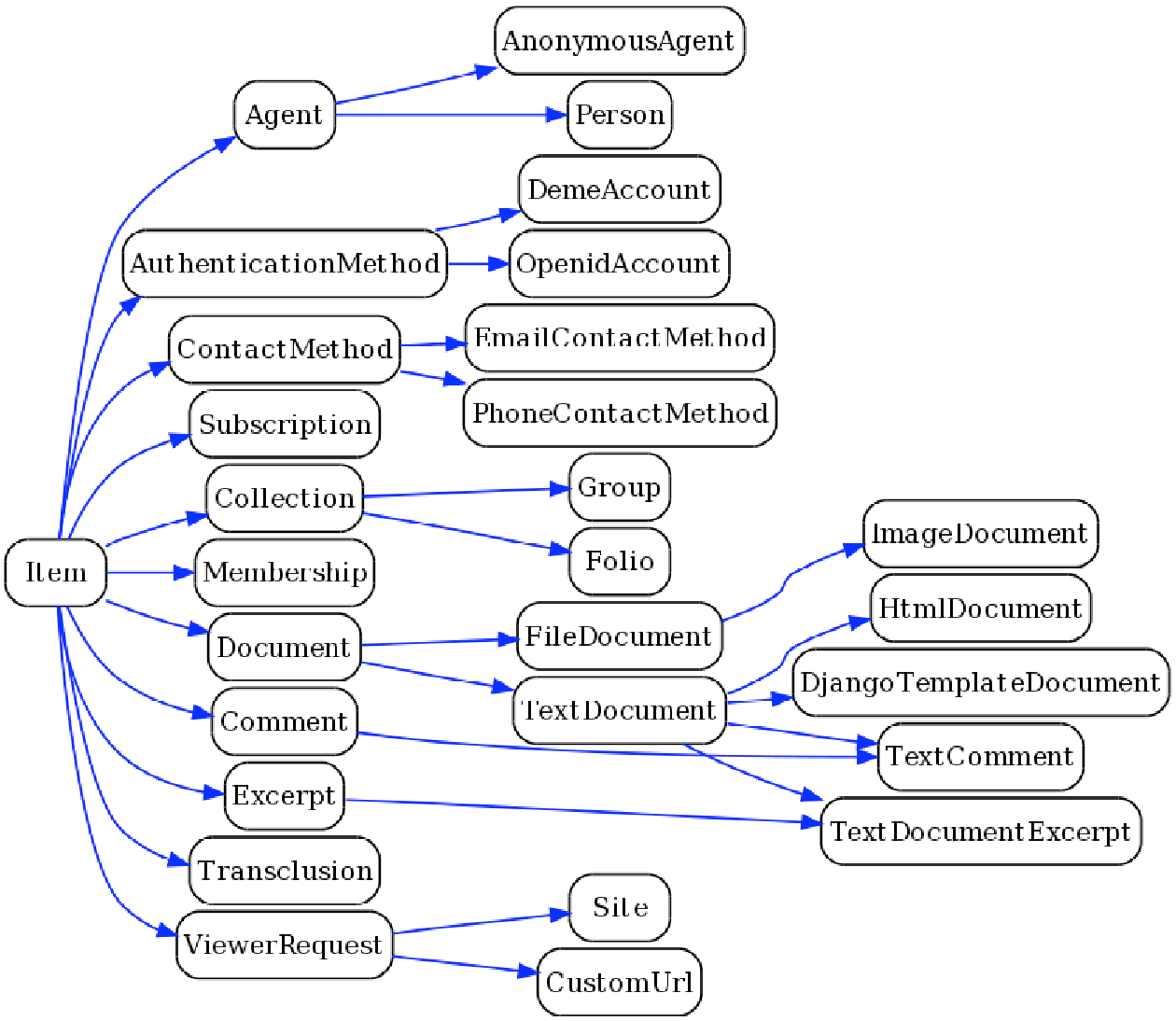
Content type inheritance

The Django web app framework

- Object-relational mapping
- Model-view code separation

Standard relational practice (no complex data structures in db cells)

Deme architecture (see IWWOST '09 paper)



Access control has evolved...

Old, discretionary access control (DAC) model:

- Files with single owners, users
- Permissions stored with user as capabilities; or with file as an access control list (ACL)

Role-based access control (RBAC) adds:

- Permissions for roles
- Support for finer grain control (e.g. fields of a database record)

An emerging paradigm for the social Web:

Relational access control (RAC)

- access control rules (ACRs) stored separately from both subject and object
- allows very flexible specification of rules as a relation between subject, object, ability, and sign (positive and negative permissions)
- subjects can be groups of users; objects can be collections of objects
- rules can be subjects of further rules
- developed in depth in theoretical work on XML access control (especially by Dongwon Lee et al.)

ACRs may be stored...

as a set of rules in a language for specifying ACRs;

or

as first-class relation objects in the same database as the objects/subjects of permissions

(relation object access control - ROAC)

ROAC versus ACMs

In an access control matrix (ACM), rows are subjects and columns are objects, and the permission is defined at each cell

In a ROAC database, each permission is its own row; columns are the fields of the permission, which is a relation object

Some advantages of ROAC

- Integrates permissions within database, so that code designed to interact with objects can access permissions/ACRs as well
- Allows permissions to be searched and discussed more easily
- Allows dynamic referencing through pointing
- Allows end users to modify permissions within the normal UI

BROAC - *Bivalent* relation object access control

Traditional permissions are positive only - no distinction between absence of permission and prohibition

Bivalent permissions may be positive or negative

Bivalent permissions are useful for representing conflicts in permissions, e.g. a personnel staff member who would otherwise have access to their own interview file

Some characteristics of social Web/collaboration environments

- Objects (photos, webpages, comments, etc.) can be tagged/labeled into multiple overlapping categories, with competing indications of permission
- Users can be members of multiple overlapping groups
- Groups can have positive, negative, or unspecified permissions

Deme permissions

Principles:

1. A permission is a relation between a subject, an object, an ability, and a sign
2. Closed world assumption - if no relevant permission exists between a subject and an object, subject does not have that ability
3. Precendence:
   - More specific has precedence over less specific
   - Subject specificity has precedence over object specificity
   - Negative has precedence over positive

Practicalities: in *Deme*, you...

can specify a permission through membership in a collection (RecursiveMembership)

cannot specify competing permissions differing only in sign

cannot specify precedence between groups or collections

Precedence by permission types in *Deme*

		Object		
		Item	*Collection*	*All Items*
	Agent	One To One (**1**)	One To Some (**2**)	One To All (**3**)
Subject	*Group*	Some To One (**4**)	Some To Some (**5**)	Some To All (**6**)
	All Agents	All To One (**7**)	All To Some (**8**)	All To All (**9**)

Conflict Resolution in Deme - examples

Example 1. The executive director of a nongovernmental organization, who is hired and supervised by the NGO's board of directors, has access to most board documents as a member of the board's Group, but does not have access to those documents related to the board's deliberations over the executive director himself. The board's Group permission for reading its Folio is positive for the Collection of executive director hiring and review documents. The executive director's Agent permission for reading this Collection is negative. The latter (negative) permission has precedence. *2(-) defeats 5(+).*

Conflict Resolution in Deme - examples

Example 2. Each student has access to their own transcript, but not to those of other students. The Group of students has a negative permission for reading a student's transcript. But a student's Agent permission is positive for reading their own transcript. The latter (positive) permission has precedence. *1(+) defeats 4(-).*

Conflict Resolution in Deme - examples

Example 3. A student is a programmer for an academic program, and also a member of the staff Group as well as the Group of students. The staff Group has a positive permission for reading student intern applications. The students Group has a negative permission for reading intern applications. The latter (negative) permission has precedence, reflecting a policy that students cannot view transcripts of other students, regardless of their staff status. *5(-) defeats 5(+).*

For more info...

http://deme.stanford.edu

Sites powered by Deme:

- http://symsys.stanford.edu
- http://odbook.stanford.edu
- http://mindroll.org