Jacqueline

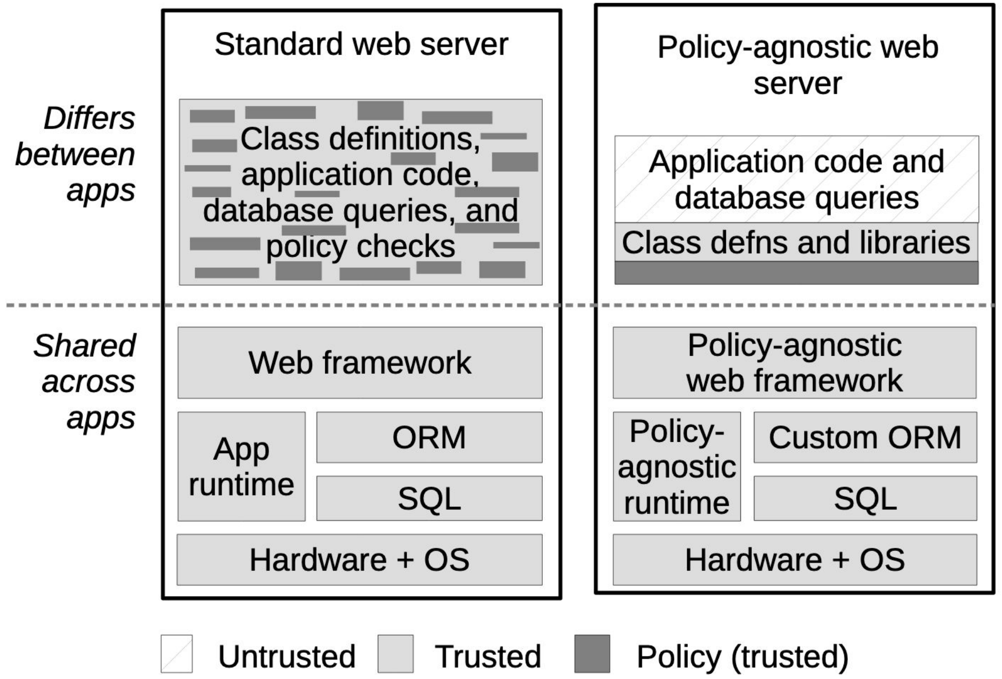
Archer Wheeler

Another Information Flow Paper?

Things to think about

- How is this different?
- Do we need another information flow control system?
- What problems does this solve?
- Does it succeed?

The Goal: Policy “Agnostic” Coding


Standard web server
Policy-agnostic web server
Differs between apps
Class definitions, application code, database queries, and policy checks
Application code and database queries
Class defns and libraries
Shared across apps
Web framework
Policy-agnostic web framework
App runtime
ORM
SQL
Policy-agnostic runtime
Custom ORM
SQL
Hardware + OS
Hardware + OS
Untrusted
Trusted
Policy (trusted)

What is an ORM?

SQL:

```
SELECT id, snn
FROM taxes
WHERE id = 47
```

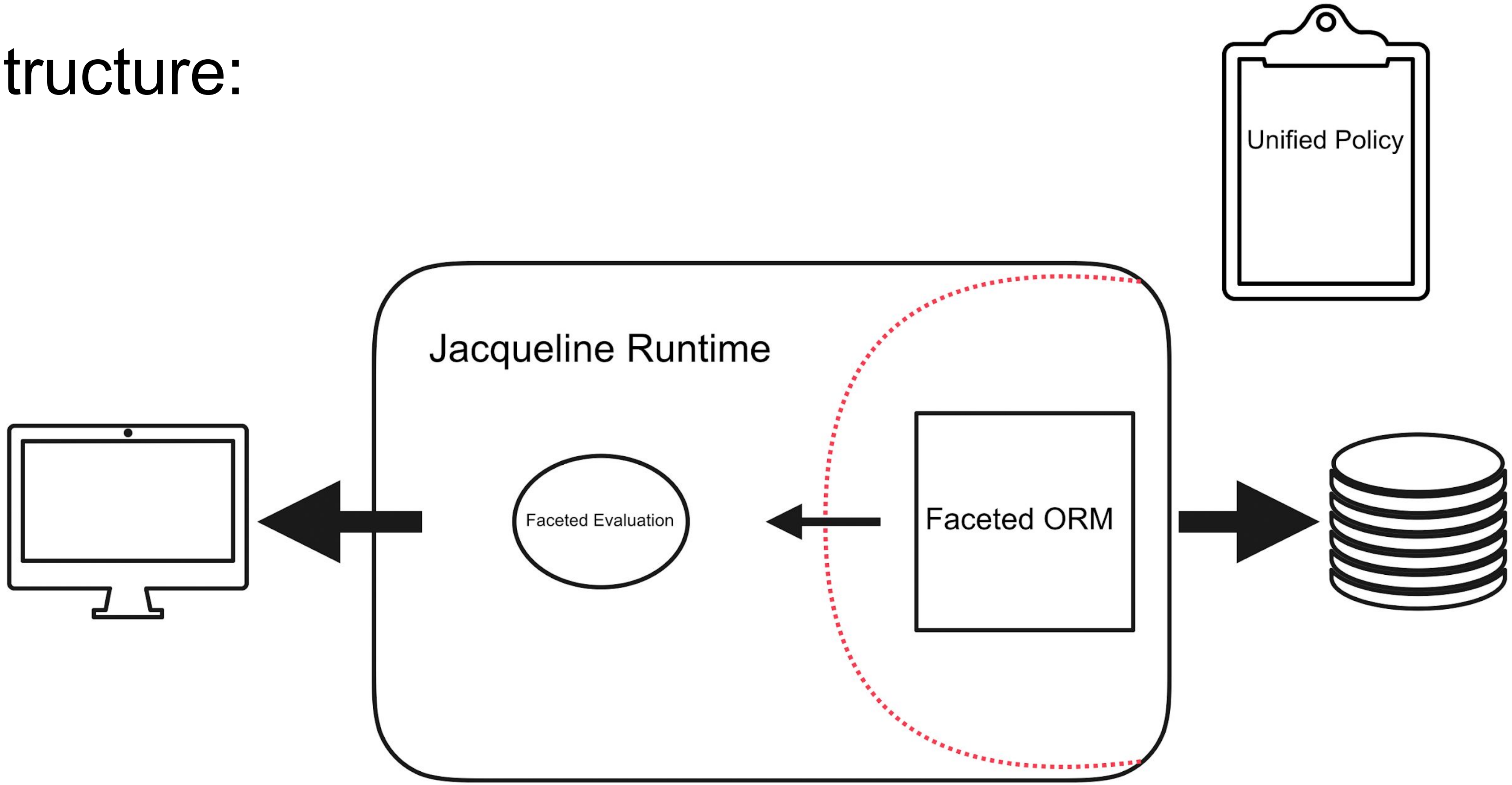
ORM pseudocode:

```
db.taxes.getById(47).name
```

What is an ORM?

- What are the advantages of using an ORM?
- What are the disadvantages?
- Why does Jacqueline use an ORM?

Structure:


Unified Policy
Jacqueline Runtime
Faceted Evaluation
Faceted ORM

Faceted Evaluation

- Data evaluates to public or private depending on the user’s runtime permissions

Paymaxx Example

- Alice’s taxes:
    - create (user = 47, name = “Alice’s taxes”, snn = “xxx-xx-xxxx”)
- Attach policy to taxes
- taxes.getById(47) evaluates to:

  < k ? (user = 47, name = “Alice’s taxes”, snn = “xxx-xx-xxxx”) : (user = 47, name = “private”, snn = “private”) >

Faceted Evaluation

- Partially hidden info. Name could be public, but ssn not:

```
for employee in my_employees:
        email(“please fill out your taxes”)
```

Faceted Evaluation

- How complicated can faceted “partial evaluations” get?
- Does this scale?

Circular Dependencies

- What if permissions to view the data depends on the data itself?
- Example:
  - Info = (name = “”, snn = “” , emergency_contacts = [Bob, Carol])

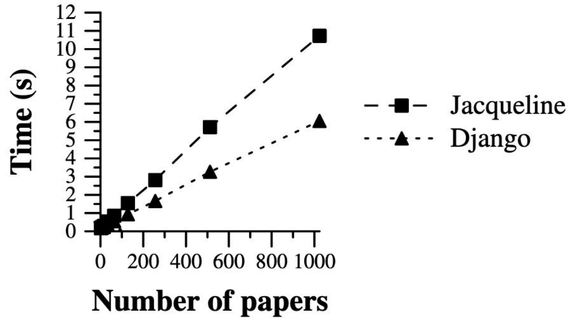
- Permission to view is dependant on the content

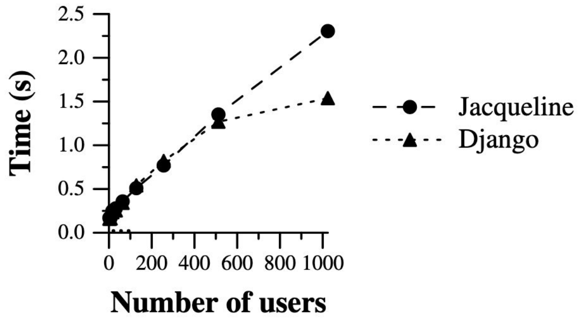
Circular Dependencies

- How does Jacqueline solve this?
- Resin & Dstar handle this problem?
- Is this a good solution?

Runtime

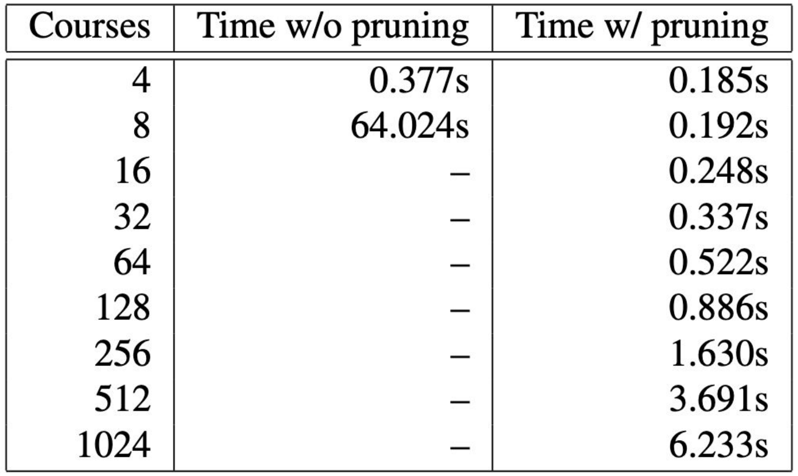
- Is Jacqueline fast enough?





Runtime

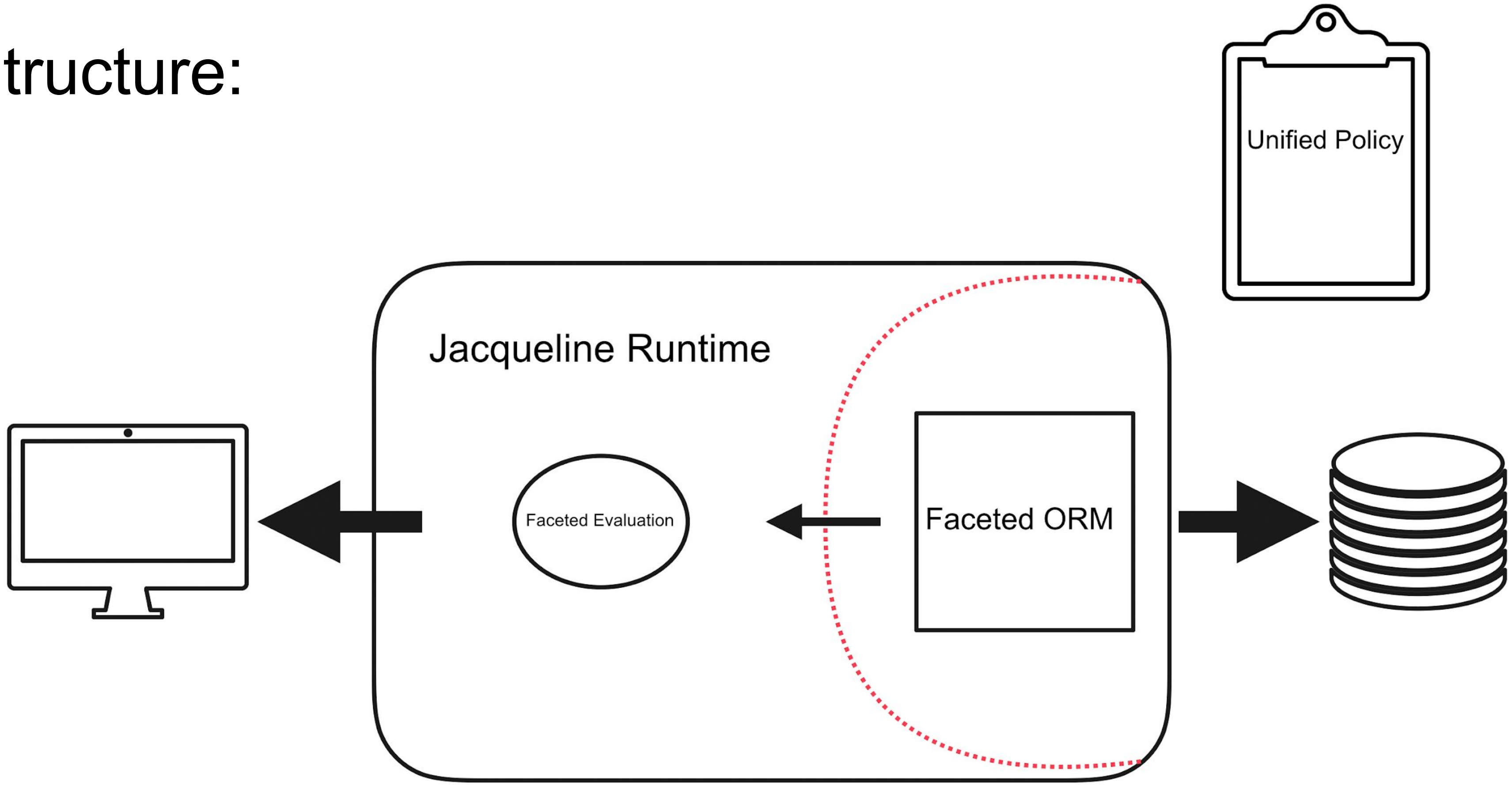
- Early pruning “optimization”



Courses	Time w/o pruning	Time w/ pruning
4	0.377s	0.185s
8	64.024s	0.192s
16	–	0.248s
32	–	0.337s
64	–	0.522s
128	–	0.886s
256	–	1.630s
512	–	3.691s
1024	–	6.233s

Table 5. Showing all courses, with and without Early Pruning.

Structure:


Unified Policy
Jacqueline Runtime
Faceted Evaluation
Faceted ORM

Structure:

- How is Jacqueline’s structure different from previous information flow papers?
- What are the tradeoffs?
- Are Jacqueline’s limitations reasonable?

Policy Agnostic Programing

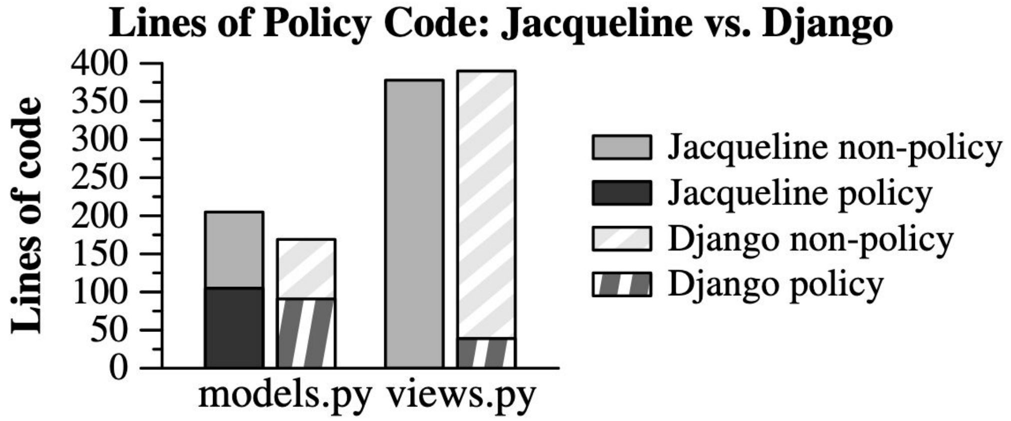


Figure 6. Distribution of policy code with Jacqueline and Django conference management systems.

Policy Agnostic Programing?

views.py

```
@login_required
@request_wrapper
@jeeves
def papers_view(request):
    user = UserProfile.objects.get(username=request.user.username)
    JeevesLib.set_viewer(user)

@login_required
@request_wrapper
@jeeves
def submit_view(request):
    user = UserProfile.objects.get(username=request.user.username)
```

Dstar vs Resin vs Jacqueline

- What problem does each solve?
- Which would you want to use?
- What is best for user privacy?
- How do they compare to a “cloud scale” solution like Zanzabar?