# Workflow Management Using FSM in Django

Calvin Hendryx-Parker, CTO, Six Feet Up

IndyPy Python Web Conf 2019

six feet up
python & cloud expert consulting

# Who is this guy?

# So What are Workflows?

# Finite State Machine

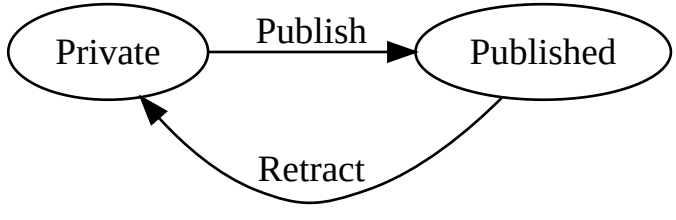
*A finite-state machine (FSM) or finite-state automaton (plural: automata), or simply a state machine, is a behavioral model used to design computer programs.*

*It is composed of a finite number of states associated to transitions.*

*A transition is a set of actions that starts from one state and ends in another (or the same) state.*

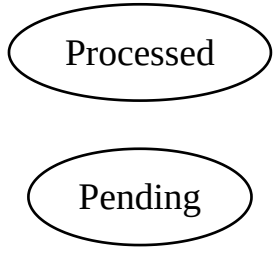
# A Simple Example

```
digraph {
    Private -> Published [label=Publish];
    Published -> Private [label=Retract];
}
```



# Truly a Boolean?

## Not when it comes to state...



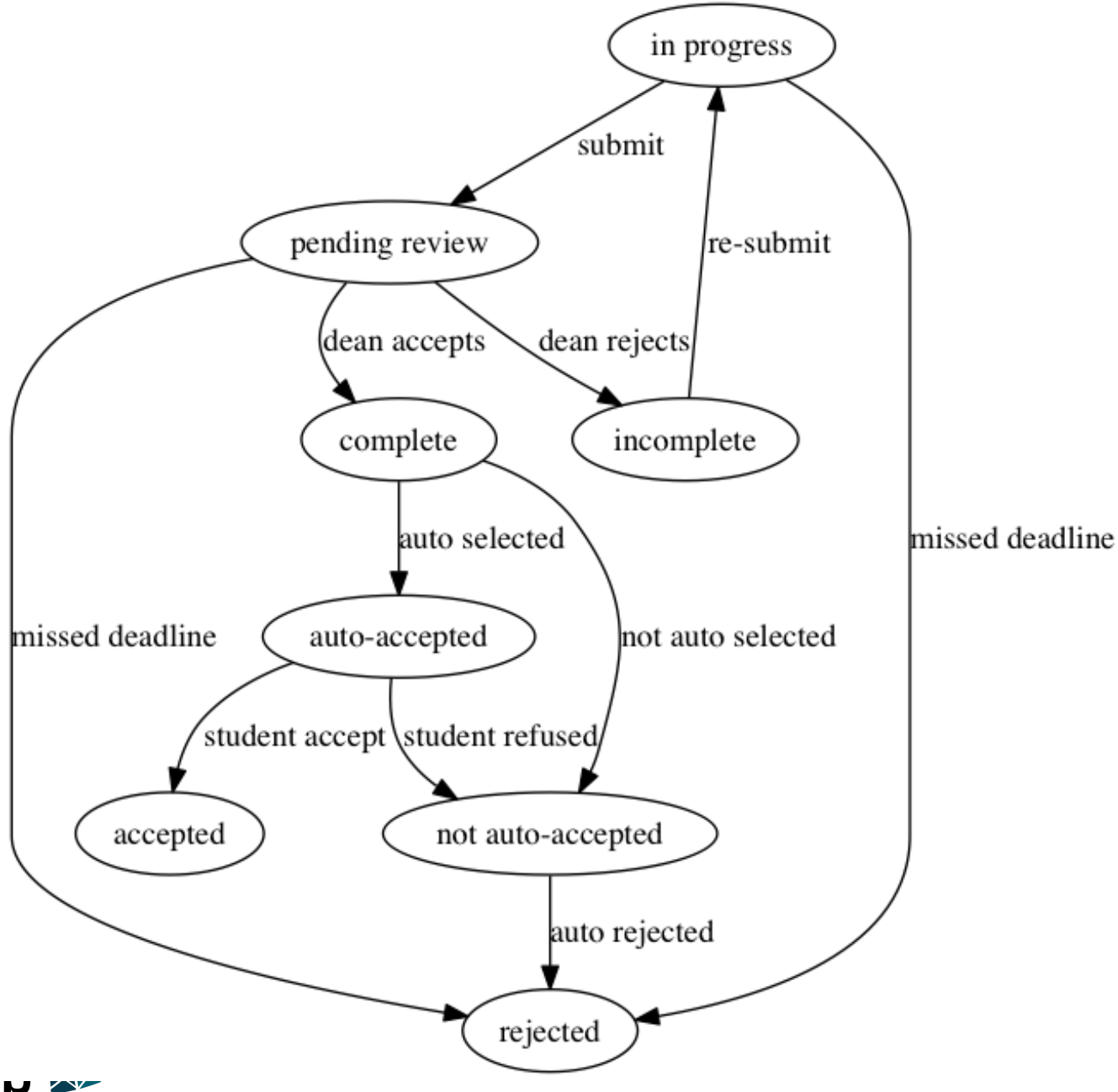

six feet up
python & cloud expert consulting

# Extended to collaboration

# OK, Real Life Please

## Ever been a student at a university?

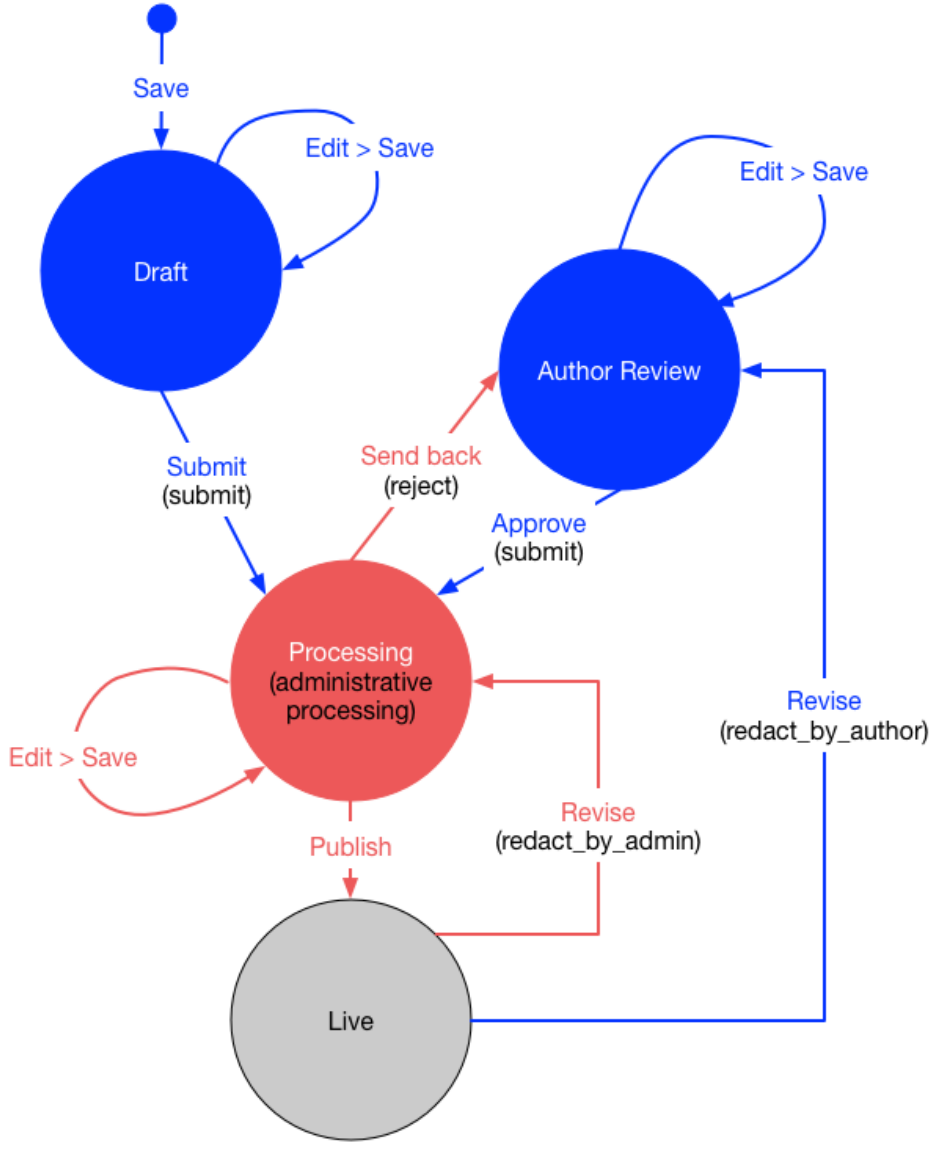
six feet up
python & cloud expert consulting




six feet up
python & cloud expert consulting

# What about researching cancer?

# Case Report Publication Workflow




six feet up
python & cloud expert consulting

# Some workflows are automatic

*A transition is started by a trigger, and a trigger can be an event or a condition.*

# So, Django....

## The Web framework for perfectionists with deadlines

six feet up
python & cloud expert consulting

# Who

## Can do what

# When

# Make some Fat Models

# Enter `django-fsm`

# Get started

https://github.com/viewflow/django-fsm

# Simple Approach

## Complicated Problem

six feet up

python & cloud expert consulting

# Basics

- Custom Model Field – `FSMField`
- Python Decorator – `@transition`
- Model Methods via `contribute_to_class()`

# FSMField

```
class CaseReport(CRDBBase, SharedObjectMixin):
    title = models.CharField(max_length=500,
                             null=True,
                             blank=True,
                             verbose_name='Case Title')
    ...
    # Workflow control fields
    author_approved = models.BooleanField(default=False,
  blank=True)
    admin_approved = models.BooleanField(default=False,
  blank=True)
    workflow_state = FSMField(max_length=50,
                              choices=WorkflowState.CHOICES,

  default=WorkflowState.INITIAL_STATE,
                              help_text="Workflow state")
```

six feet up
python & cloud expert consulting

# States Definition

```
class WorkflowState(object):
    DRAFT = 'draft'
    ADMIN_REVIEW = 'processing'
    AUTHOR_REVIEW = 'author review'
    LIVE = 'live'
    RETRACTED = 'retracted'
    ...
    CHOICES = (
        (DRAFT, DRAFT.title()),
        (ADMIN_REVIEW, ADMIN_REVIEW.title()),
        (AUTHOR_REVIEW, AUTHOR_REVIEW.title()),
        (LIVE, LIVE.title())
    )
    ...
    INITIAL_STATE = DRAFT
    ...
```

# How do you get to these states?

```
@transition(field=workflow_state,
            source=[WorkflowState.DRAFT],
            permission=can_edit,
            target=WorkflowState.DRAFT
            )
def edit(self, by=None):
    pass
```

six feet up
python & cloud expert consulting

```
@transition(field=workflow_state,
            source=[WorkflowState.AUTHOR_REVIEW, ],
            permission=can_submit,
            target=WorkflowState.ADMIN_REVIEW)
def approve(self, by=None):
    """ send to admins with approval """
    self.author_approved = True
    self.admin_approved = False
    try:
        emails.approved(self)
    except Exception as mail_err:
        print(mail_err)

    return "Thank you for approving your Case Report. We will
  contact you" \
           " when it goes live."
```

six feet up
python & cloud expert consulting

# Define Permissions

```
def can_edit(self, user=None):
    if not user:
        user = CurrentUserMiddleware.get_user()
    if self.workflow_state in (WorkflowState.DRAFT,
                               WorkflowState.RETRACTED,
                               WorkflowState.AUTHOR_REVIEW)
  and \
        user.email == self.primary_author.email:
        return True
    if self.workflow_state in (WorkflowState.ADMIN_REVIEW,)
  and \
        user.is_staff:
        return True
    return False
```

six feet up
python & cloud expert consulting

```python
    def can_submit(self, user=None):
        # ensure author
        if not user:
            user = CurrentUserMiddleware.get_user()
        return user.email == self.primary_author.email
```

# Some Convenience

**`get_all_FIELD_transitions`**

Enumerates all declared transitions

**`get_available_FIELD_transitions`**

Returns all transitions data available in current state

**`get_available_user_FIELD_transitions`**

Enumerates all transitions data available in current state for provided user

# Signals

- `django_fsm.signals.pre_transition`
- `django_fsm.signals.post_transition`

```
def casereport_workflow_transitions(sender, **kwargs):
    # this is the hook that should handle all side effects of
  state
    # change transitions like sending emails, clearing
  queues, etc.
    cr = kwargs['instance']
    transition_name = kwargs['name']
    source_state = kwargs['source']
    end_state = kwargs['target']
    if end_state != source_state:
        print("handling %s transition for %s" %
  (transition_name, cr))
        print("...TODO...\n\n")



fsm_signals.post_transition.connect(
    casereport_workflow_transitions,
    sender=CaseReport
)
```

# Who Dunnit!

```
    def can_edit(self, user=None):
        if not user:
            user = CurrentUserMiddleware.get_user()
    ...
```

six feet up
python & cloud expert consulting

```
    @transition(field=workflow_state,
                source=[WorkflowState.ADMIN_REVIEW, ],
                permission=can_reject,
                target=WorkflowState.AUTHOR_REVIEW,
                )
    def send_back(self, by=None):
        """ send the CR back to the author
        """
        self.admin_approved = False
        try:
            emails.send_back(self)
        except ConnectionRefusedError:
            pass
        user = CurrentUserMiddleware.get_user()
        author =
User.objects.get(email__exact=self.primary_author.email)
        action.send(user, verb='sent back',
action_object=self, target=author)
        return "The case report has been sent back to its
author."
```

# Optimistic Locking

```
from django_fsm import FSMField, ConcurrentTransitionMixin

class BlogPost(ConcurrentTransitionMixin, models.Model):
    state = FSMField(default='new')
```

six feet up
python & cloud expert consulting

# Want an audit log?

```
@fsm_log_by
@transition(field=workflow_state,
            source=[WorkflowState.DRAFT],
            permission=can_edit,
            target=WorkflowState.DRAFT
            )
def edit(self, by=None):
    pass
```

https://github.com/gizmag/django-fsm-log

six feet up
python & cloud expert consulting

# Add FSM to the Django Admin!

https://github.com/gadventures/django-fsm-admin

six feet up
python & cloud expert consulting

# Keep from making a mess...

## Take an FSM first strategy

Add this to your config for `django-fsm-admin`

```
FSM_ADMIN_FORCE_PERMIT = True
```

six feet up
python & cloud expert consulting

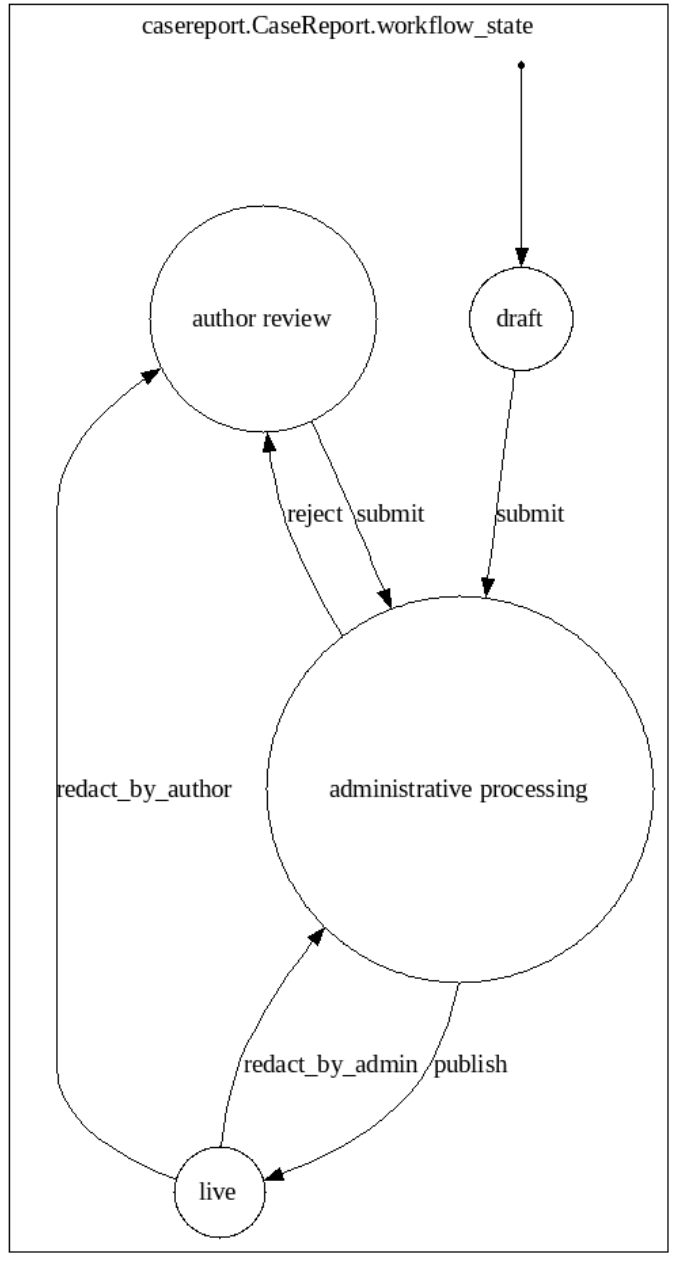
# Django Management Command!

```
$ ./manage.py graph_transitions casereport.CaseReport  > t.dot
$ dot -O -Tpng t.dot; open t.dot.png
```

six feet up

python & cloud expert consulting

```
ModuleNotFoundError: No module named
'graphviz'
```




six feet up
python & cloud expert consulting

# Questions?

calvin@sixfeetup.com

@calvinhp