

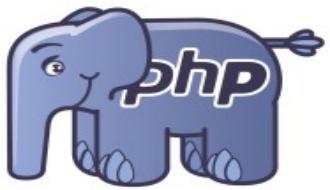


HEY, BOSS!
EVENT SOURCING
CAN FIX THAT!

EMILY STAMEY



WOMEN WHO
CODE
RALEIGH/DURHAM



I LOVE LEGACY!



LEARNING OBJECTIVES

- Basics of Event Sourcing using an example of a library (the kind where you check out books)
- Three projects I have worked on and the ways we used Event Sourcing on them

BEFORE WE BEGIN

Learning Event-Sourcing (or DDD) is tough! i

Be kind to learners (including yourself)

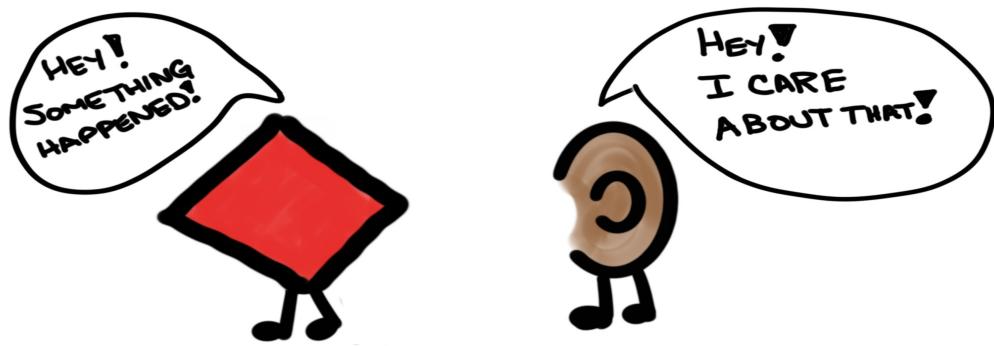
Event-Sourcing isn't for your whole application

EVENT SOURCING

The fundamental idea of Event Sourcing is that of ensuring **every change to the state of an application is captured in an event object**, and that these event objects are themselves stored in the sequence they were applied for the same lifetime as the application state itself.

Martin Fowler

EVENTS AND LISTENERS



AN EVENT

What happened?

BookWasCheckedOut

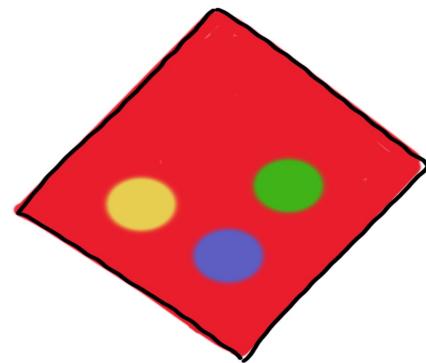
What do I need to remember about
it?

(book, patron, date)



EVENT ATTRIBUTES

- Save only what you need to preserve
- The rest can be looked up
(book id, patron id, date)



EVENT CLASS

```
<?php

namespace Library\Events;

use Library\Support\Event;

class BookWasCheckedOut
{

    /**
     * @var DateTime
     */
    protected $checkoutDate;
```

RULES FOR EVENTS

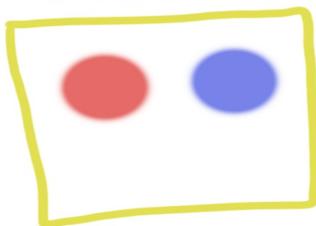
- Usually named as past-tense verbs
- **RARELY** changed
- **Never** deleted
- Has attributes that are values
 - not model, object, collection, or aggregate root

NEVER DELETE Events

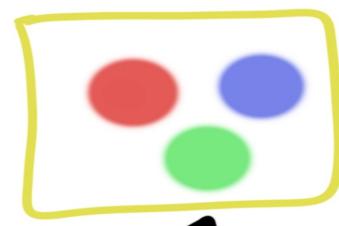


DON'T STORE OBJECTS

OBJECT 1

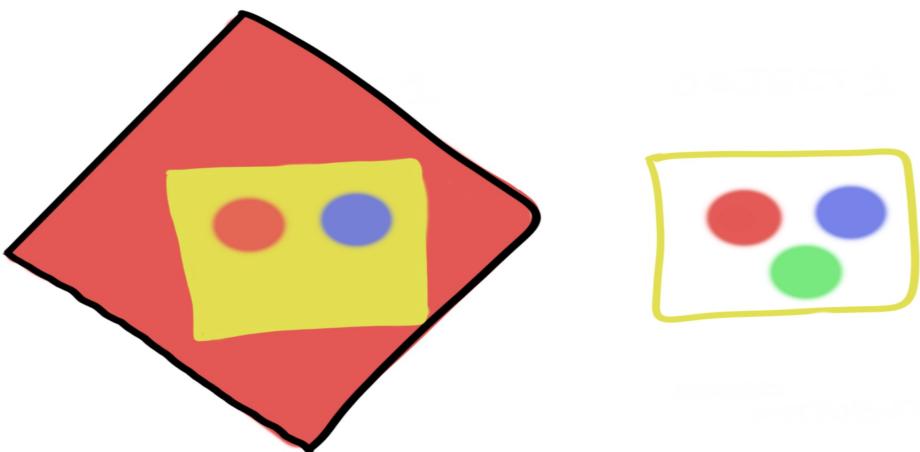


OBJECT 1



ADDED
ATTRIBUTE

IF WE STORED OBJECTS IN AN EVENT...



YOU WERE SO PREOCCUPIED WITH WHETHER OR NOT YOU COULD

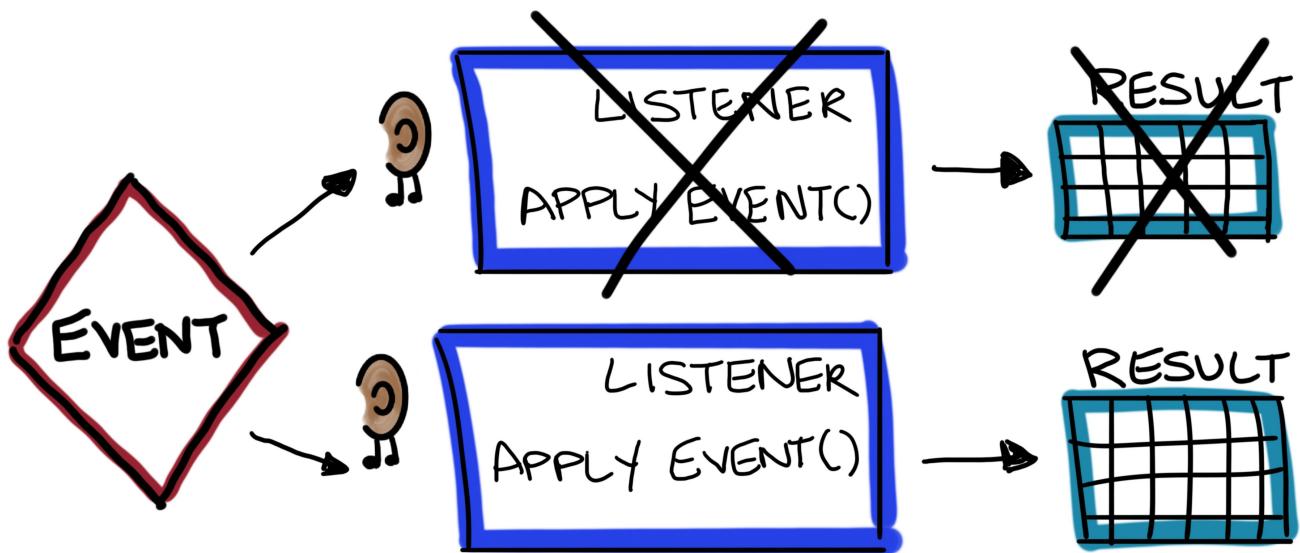


YOU DIDN'T STOP TO THINK IF YOU SHOULD

EVENTS RARELY CHANGE

- The part of the code that will change is most likely the **result that follows that event**.
- The **structure of the resulting data** is more likely to change than the thing that happened







REASONS TO USE EVENTS

- State transitions are important
- We need an audit log, proof of the state we are currently in
- The history of what happened is more important than the current state
- Events are replayable if behavior in your application changes



DOMAIN MESSAGE ID (UUID)

TYPE

PAYLOAD →



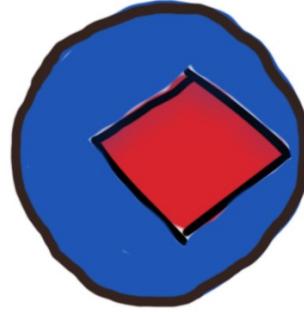
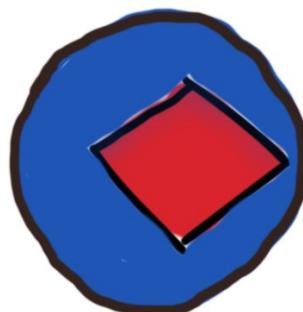
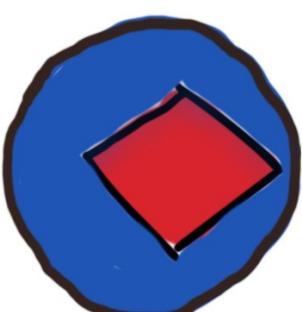
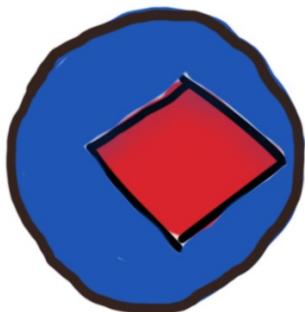
← THE
EVENT

TIMESTAMP

VERSION

EVENT STORE

- Domain-specific database
- Based on a Publish-Subscribe message pattern



PROJECTOR

```
<?php

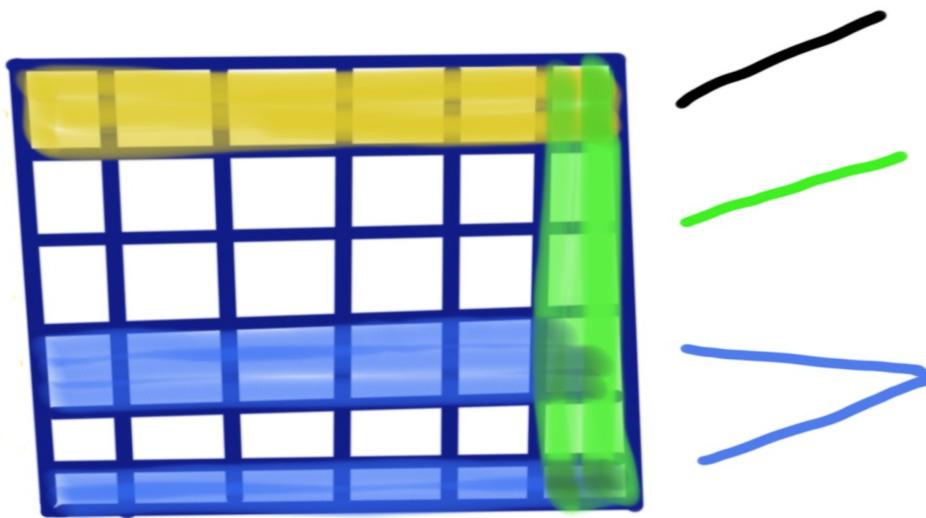
namespace Library\ReadModel;

use Library\Events\BookWasCheckedIn;
use Library\Events\BookWasCheckedOut;
use Library\Events\BookAddedToBookshelf;
use App\Support\ReadModel\Replayable;
use App\Support\ReadModel\SimpleProjector;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Connection;

class BookshelfProjector extends SimpleProjector implements Replayable
{
```

A set of event handlers that work together to build and maintain a table to be accessed by the read model.

READ MODEL



READ MODEL

```
<?php

namespace Library\ReadModel;

use Carbon\Carbon;
use Illuminate\Database\Eloquent\Model;

/**
 * @codeCoverageIgnore
 */
class Bookshelf extends Model
{
    protected $table = 'proj_bookshelf';
    public $incrementing = false;
```

CREATING THE EVENTS

- An event is created only after validation
 - Directly in a controller 'checkout' method
 - Using a Check Out Book Command and Handler

CQRS

COMMAND AND QUERY RESPONSE SEGREGATION

An application architecture pattern commonly used with event sourcing

CQRS involves splitting an application into two parts internally.

CQRS

- **Command** is any method that mutates state
- **Query** is any method that returns a value
- Should only be used on specific portions of a system, not the system as a whole

COMMAND HANDLER

1. Validate the command on its own merits.
2. Validate the command on the current state of the aggregate.
3. If validation is successful, create an event(s)
4. Attempt to persist the new events. If there's a concurrency conflict during this step, retry or exit.

COMMAND HANDLER EXAMPLE

```
public function update(Request $request)
{
    // $request has book id, patron id

    try {

        $command = new CheckOutBook($request->bookId, $request->patronId);

        $this->bookLendingService->handleCheckOutBook($command);

    } catch (InvalidUserException $e) {
        return response()->json("Not authorized to check out a book.", Response::HTTP_FORBIDDEN);
    } catch (BookUnavailableException $e) {
        return response()->json("Book was not available to be checked out", 400);
    }
}
```

OPTIMIZE

- Separate the load from reads and writes allowing you to scale each independently.
 - All **commands** go into a WriteService
 - All **queries** go into a ReadService

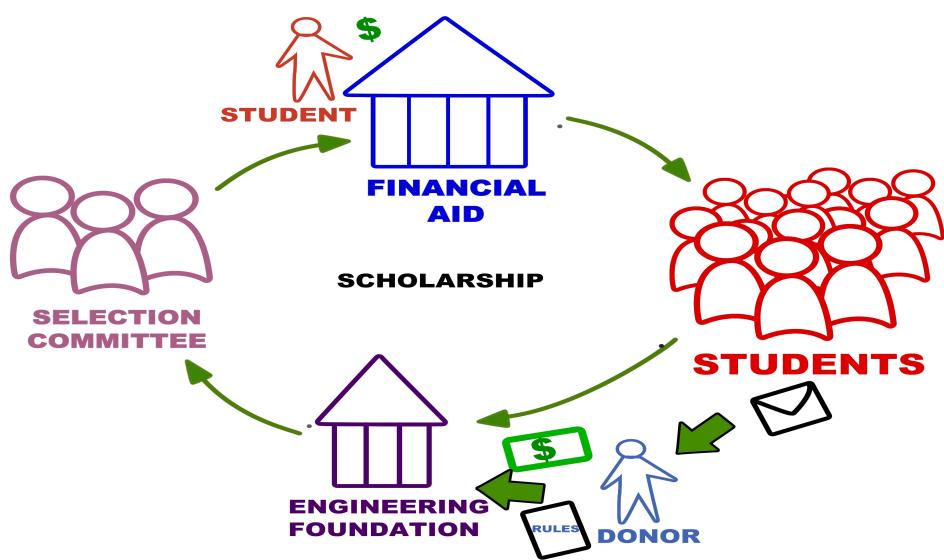
YOU CAN CHOOSE WHICH IS BEST

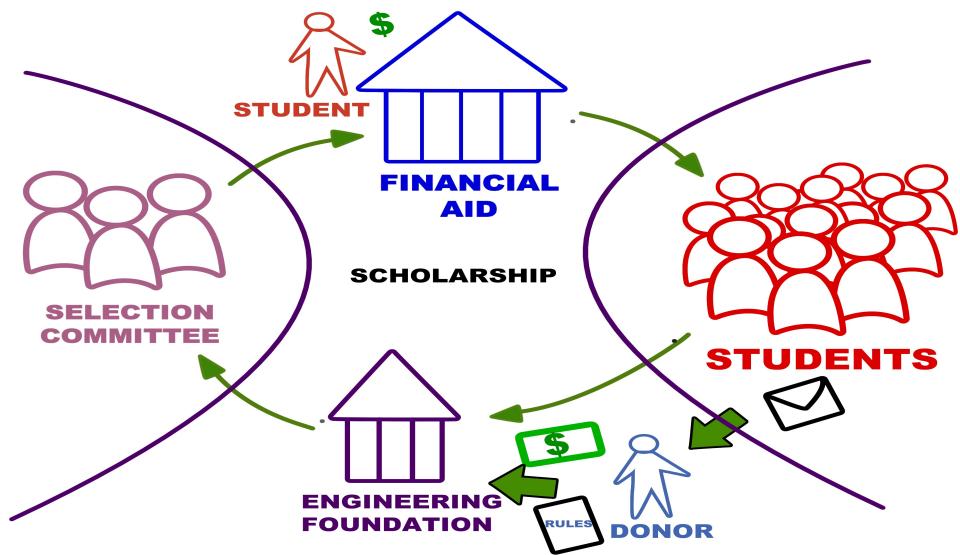
- CRUD
- Event-Sourcing
- Event-Sourcing with CQRS
- Event-Sourcing, CQRS, DDD

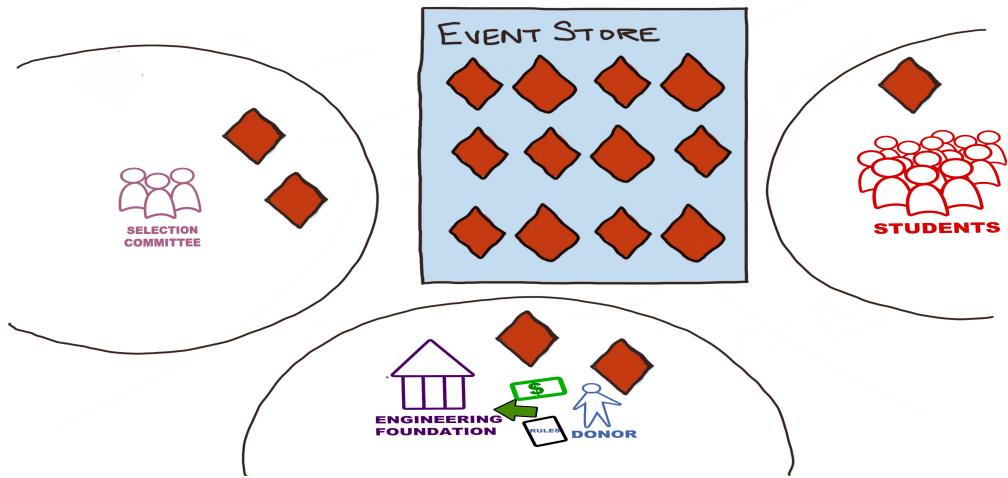


MY PROJECTS

- Scholarships
- Course Registration
- Threat Reports







FLEXIBLE TO CHANGES

- Selection Committee would never take away an award
 - until they did
- New academic year, new event store

SCHOLARSHIPS WRAP-UP

- Modernized the code in pieces
- View events from multiple contexts
- Flexible to changes in the application

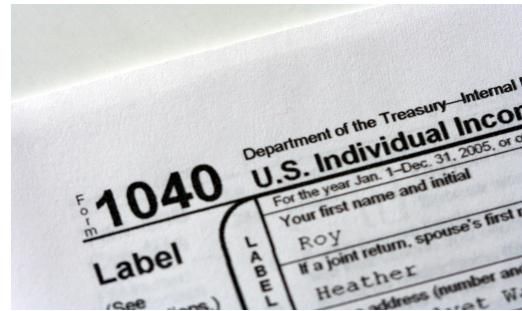
STUDENT ENROLLMENT PROCESS

- For Distance Education Students in the College of Engineering
- Rewrote Application
- ES to follow the process
- Status drop-down versus events

BEGAN WITH PAPER FORMS

A lot of our systems were built to
replace paper processes

They often closely map to this
physical form.



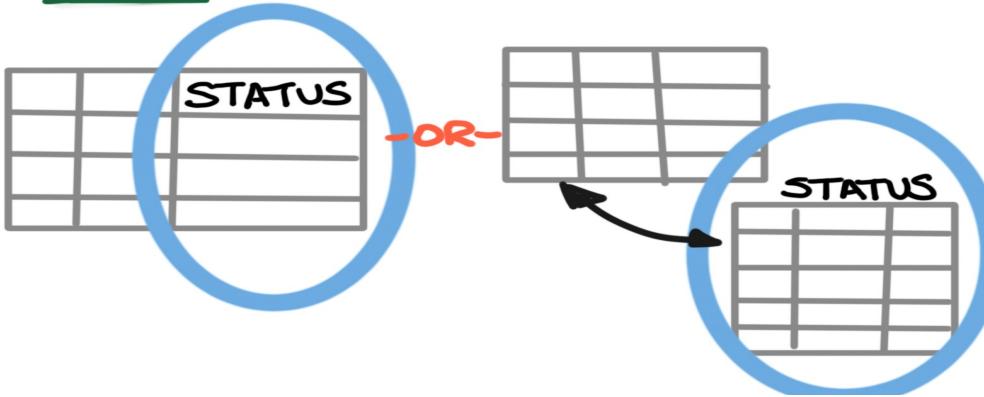
© PHOTO BY AIDAN MORGAN

PAPER FORMS HANDLING STATE

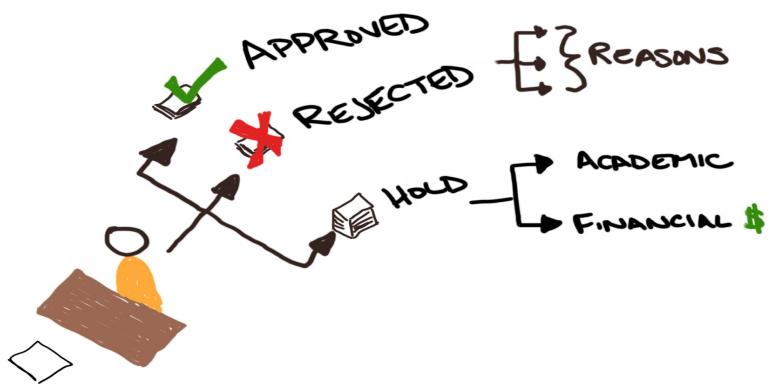


Status labels are like a rubber stamp
Status doesn't always communicate
why or what happened

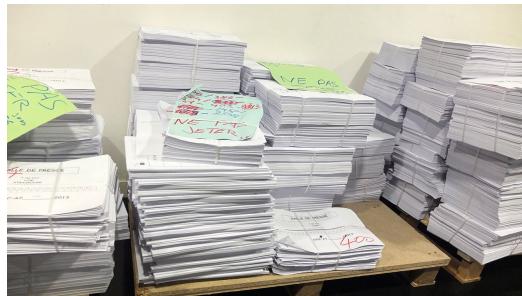
COMPENSATING MEASURES



HOW WORKFLOWS BECOME COMPLEX



PAPER FORMS HANDLING STATE



Piles indicate status of the form

A SIMPLE STATUS DROP-DOWN

RECEIVED

A "SIMPLE" STATUS DROP-DOWN

RECEIVED
CANCELLED
COURSE CANCELLED
COURSE NOT APPROVED
CPC DENIED
CPC NOT APPROVED
CPC PENDING TRANSCRIPT
CPC PROCESSING
DENIED
DROPPED AFTER CENSUS
DROPPED BEFORE CLASSES BEGUN
DROPPED BETWEEN BEGINNING OF CLASS AND CENSUS DATE
DROPS/WITHDRAWALS AT SITES
ECE ON CAMPUS STUDENTS
ENROLLMENT CANCELLED
EOL APPROVED
EOL MISC
NEW STUDENT REGISTRATION
PENDING ON CAMPUS REQUEST
ON CAMPUS STUDENT NOT APPROVED
PENDING CASHIER HOLD
PENDING EOL APPROVAL
PENDING INSTRUCTOR APPROVAL
PENDING INTERNATIONAL STUDENT
PENDING NDS OPEN ENROLLMENT
PENDING OIS VISA STUDENT
PENDING PERMANENT RESIDENT
PENDING TERM ADVISEMENT HOLD
PENDING TRANSCRIPT
PENDING TUITION PREPAYMENT
PREAPPROVED RETURNING
PROCESSING NDS
PROCESSING SITE
PROCESSING Z
PROJECT MESSAGE - MAE 586
PROJECT MESSAGE - NE 693
REGISTERED
REGISTERED ASHEVILLE
REGISTERED HAVELOCK
REGISTERED WILMINGTON
SITE APPROVAL: ASHEVILLE
SITE APPROVAL: HAVELOCK
SITE APPROVAL: WILMINGTON

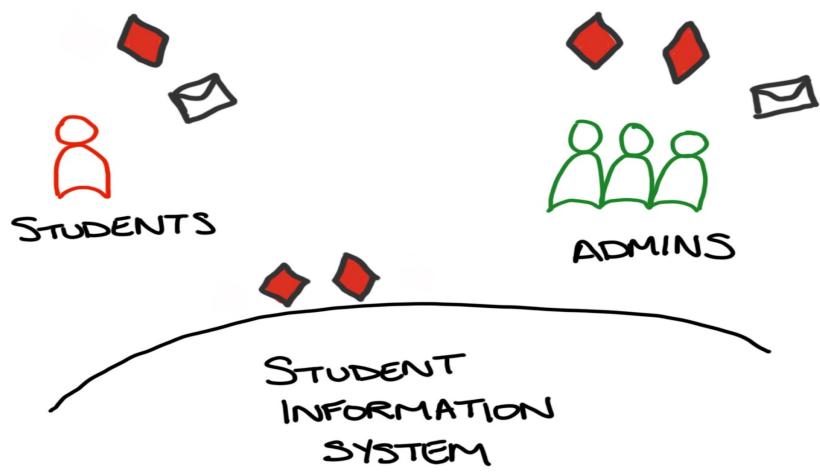
SOMETHING HAPPENED

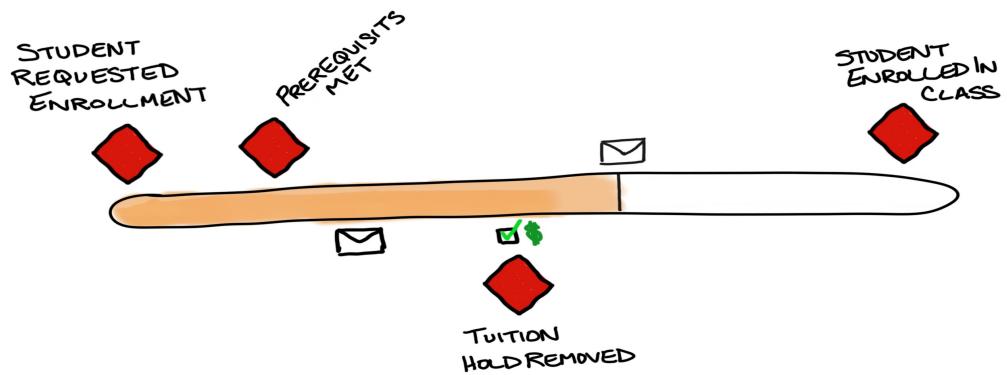
Status is a reflection of something that happened

There is ONE of each status + reasons/details

Events can record what happened







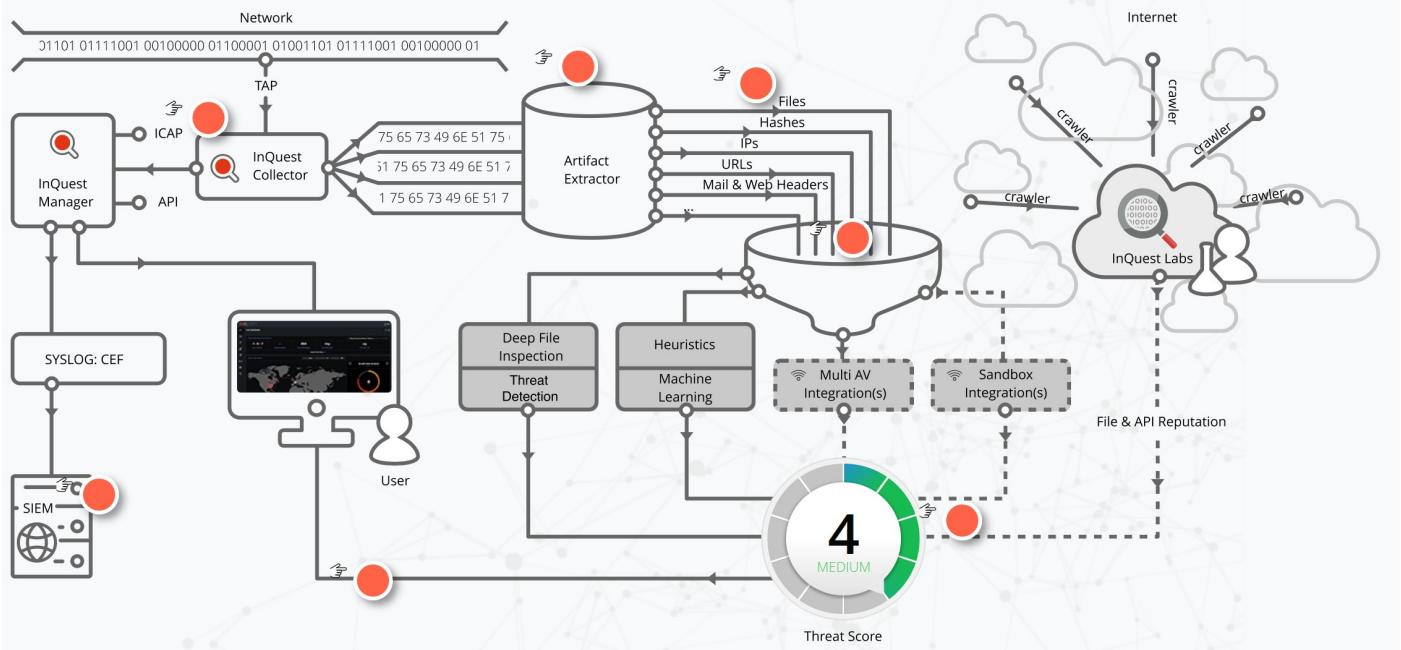
STUDENT ENROLLMENT WRAP-UP

- Streamlined Process
- Connected the system to Student Information System
- Facilitated communication between students and admins

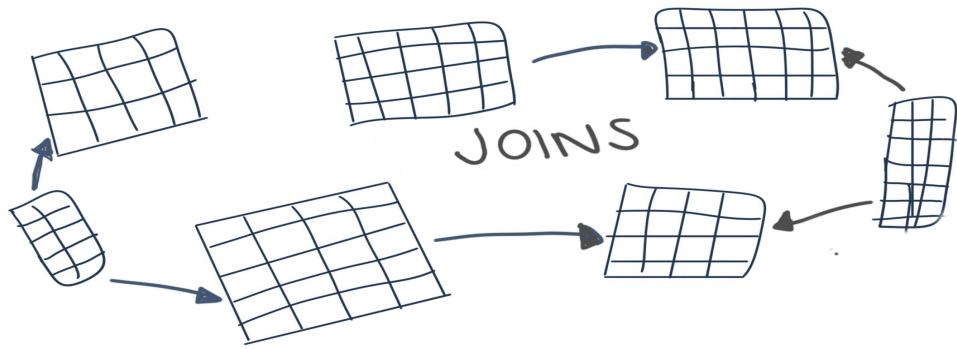


How It Works

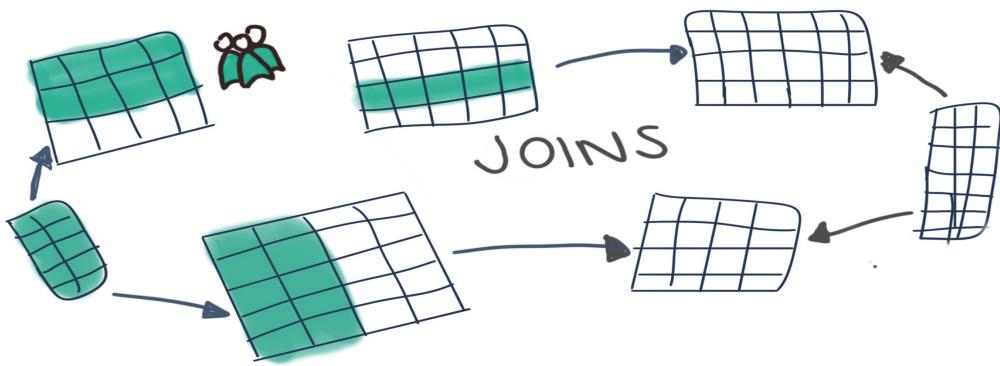
SOC oriented tool designed by SOC analysts for SOC analysts. Primary deployment today is within the US federal government.



ANALYSIS



DATA FILTERED FOR USER ACCESS



INQUEST WRAP-UP

- Events created by the Engine; used in Customer-facing Site
- Events capture Session Threat History
- Optimize the results for Read, reducing time to retrieve large amounts of data
- Results are still filtered by a user's access
- Full audit log
- These events will lead to more improvements

PLANNED IMPROVEMENTS

- Fewer complex DB queries
- Could separate us from older schema; or reduce dependence on it
- Separate the logic of what an event means based on context and purpose
- Flexible to change, our interpretation of events can change, and we can rebuild projections without losing the full history

THANK YOU!



EMILY STAMEY

@ELSTAMEY

[HTTP://ELSTAMEY.COM](http://ELSTAMEY.COM)