Incorporating Software Engineering into Instrumentation Teams

GRD Seminar

November 5th 2020

Chris Moriarty

CENTER FOR ASTROPHYSICS
HARVARD & SMITHSONIAN

Motivation

- Science and Software have become inseparable
  - Software *Engineering* is separate though, for now ;)
  - Complex systems require more than “someone to finish the software later”
- Instrumentation software control systems are challenging
  - Developing code for constantly changing hardware is difficult
  - Safely operating expensive systems is paramount
  - Many systems just end up growing organically into spaghetti code
- Some software design, and best practices can go a long way
  - There are often barriers in the way of doing this though

Respect the Software People

"Software people are kind of strange, over in the corner, and kind of viewed as a service. But as we know nowadays, your system is very integrated. Your software capability really drives the capability of the overall system."

— Kathy Lueders, NASA associate administrator for human exploration and operations, discussing the importance of software during a call with reporters Tuesday about reviews of last December's Starliner test flight, which suffered software problems.

Historical Barriers

The Nature of New Projects & Missions

- New projects are often started with only science folks on the team
  - Proposals generally don’t emphasize software design or funding
- Initial development often done without software engineers
  - Crucially misses design and requirements
- Success defined by publishing single results, rather than reliable, repeatable results.
- Complex and unstable systems
  - “Ok, now maybe we should hire some software people”
- **Not enough funding to hire additional software support**

Staffing Grad students & Post Docs

- By no fault of their own are driven by “Science First”
  - Incentive for honing software skills is often low
  - Software skills don’t publish papers -> doesn’t progress science career
  - Often treated as a hobby for one to pursue on their own time
- Turnover
  - Term limited staff will inevitably leave
  - Science leaves with them, which is the norm in the science world
  - **Systems knowledge of complex code leaves the project too**
    - The team will have to ramp someone new up
    - Bugs and problems will go unfixed, potentially blocking other efforts

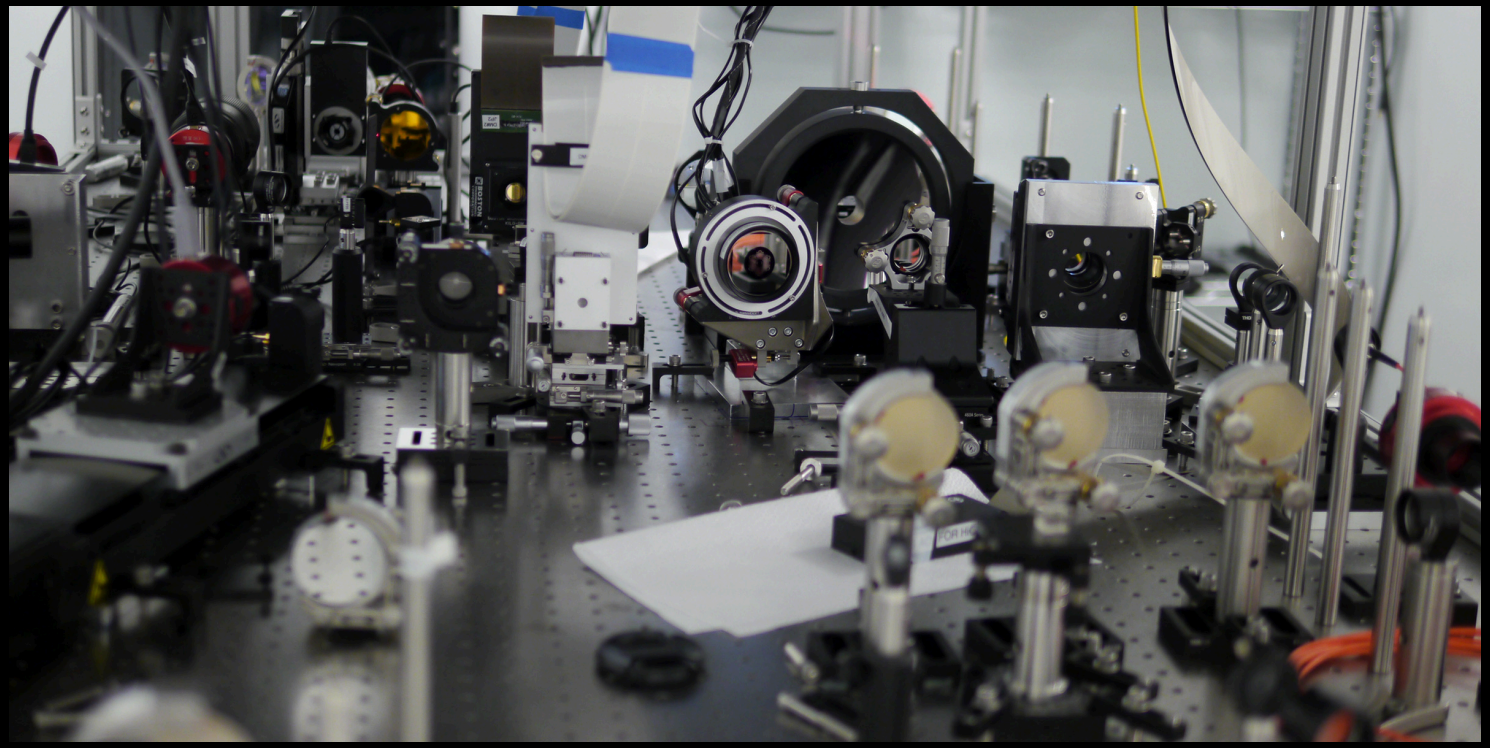
Institutional hurdles

- Institutes and Observatories often separate engineering and science
  - Teams tend to be either all science people, or all engineering people
  - Standards set in one team aren't adopted elsewhere
    - Uncoordinated silo development
  - Instrumentation teams particularly suffer from this model
    - Which almost always require a diverse collaborative team
- Software often viewed as a service
  - Since teams are separated and uncoordinated, close integration is more difficult
- Even pure engineering teams often end up with science managers
- Its not always obvious how to share resources between divisions
- **Changing an organization's culture is hard and takes time**

First steps to get passed barriers

- Incorporate realistic software budget into proposals
- Consult software engineering collaborators
  - Conduct software design study early and iteratively develop with hardware
    - Treat both as a single system
- Create team standards, documentation, and onboarding process
  - Less of a hit when students and post docs leave project
- Build a software friendly culture
  - Support and advertise software and management training when available
  - Go to software conferences too

High-contrast imager for Complex Aperture Telescopes (HiCAT)



In the beginning…

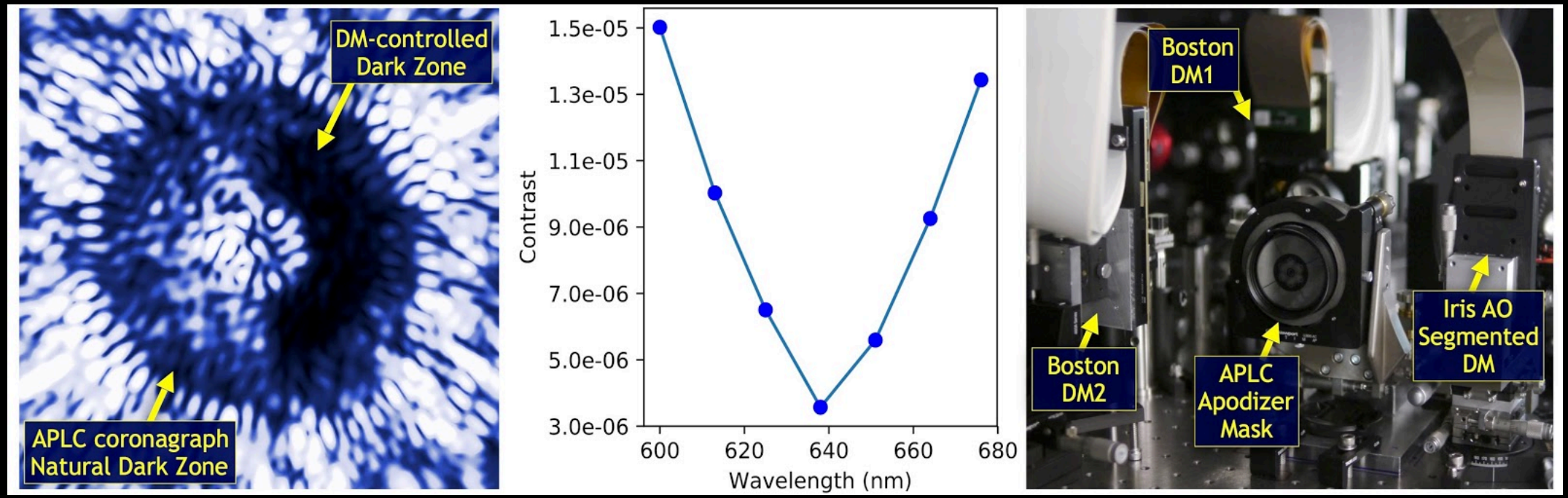
- Stalled project, 4 years of development, no results, low funding
- Strategically proposed for director's discretionary funds
  - Happened to knock on Remi's door with a crazy idea
  - Shaped the proposal to be a collaboration between divisions
- Became the first STScI software engineer to work on a DD project
  - Tried to change my title and lower my salary
  - Was told there may not be a position for me when the DD ends
  - Decided to be the guinea pig
- Software system was a complete MESS!
  - Matlab, LabVIEW, Mathematica, C++, oh my!

Initial Core HiCAT DD Team

- PI – Expert in high contrast imaging
- Senior Software Engineer with no experience in optics
- Part time Senior Hardware Engineer with lots of experience
- Part time IT administrator
- Part time science staff with optics and data processing experience

Road to a dark zone

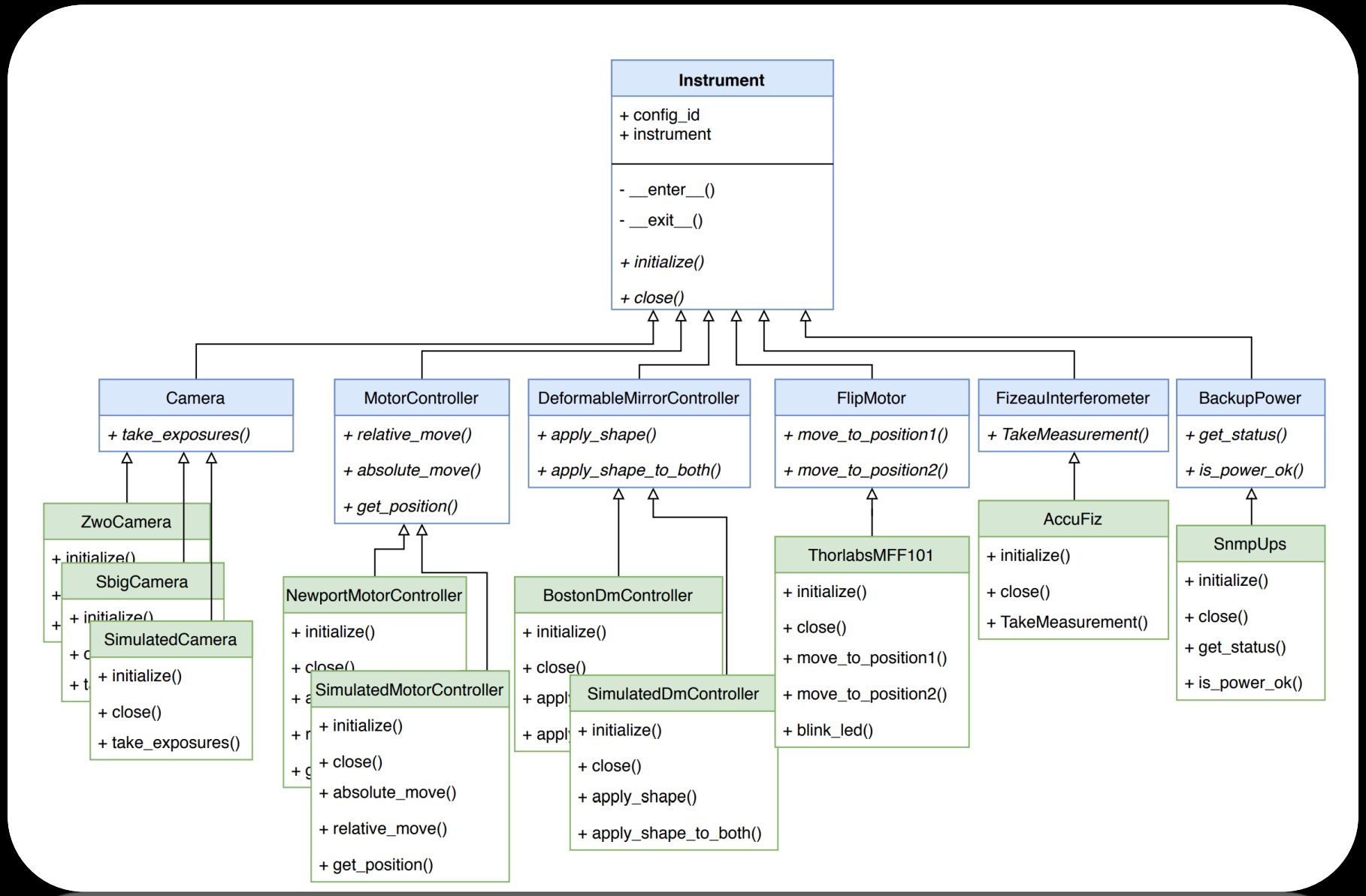
- Found and moved all code into version control
- Re-wrote entire software control system in Object Oriented Python
- Trained entire lab in git, conda, and Jira
- Took a class on fourier transforms at JHU
- Developed a realtime data pipeline
- Incorporated a simple darkzone algorithm (speckle nulling)
- Began project management using proposal and paper deadlines
- Automated control and safety checks
- Created a queueing system for overnight and weekend experiments



“The software infrastructure he has designed is totally unique and I believe unheard of in our field” -Rémi Soummer

HiCAT Debrief

- Diverse team covering IT, Hardware, Software and Science Theory
- Software budget in proposal
- Software best practices
  - Ditched spaghetti code - > Object oriented design
  - Version control
- Cross training science theory, optics, project mgmt and software
- Onboarding documentation
- Expresso machine ☺


Instrument
+ config_id
+ instrument
- __enter__()
- __exit__()
+ initialize()
+ close()
Camera
+ take_exposures()
MotorController
+ relative_move()
+ absolute_move()
+ get_position()
DeformableMirrorController
+ apply_shape()
+ apply_shape_to_both()
FlipMotor
+ move_to_position1()
+ move_to_position2()
FizeauInterferometer
+ TakeMeasurement()
BackupPower
+ get_status()
+ is_power_ok()
ZwoCamera
SbigCamera
SimulatedCamera
+ initialize()
+ close()
+ take_exposures()
NewportMotorController
+ initialize()
SimulatedMotorController
+ initialize()
+ close()
+ absolute_move()
+ relative_move()
+ get_position()
BostonDmController
+ initialize()
+ close()
SimulatedDmController
+ initialize()
+ close()
+ apply_shape()
+ apply_shape_to_both()
ThorlabsMFF101
+ initialize()
+ close()
+ move_to_position1()
+ move_to_position2()
+ blink_led()
AccuFiz
+ initialize()
+ close()
+ TakeMeasurement()
SnmpUps
+ initialize()
+ close()
+ get_status()
+ is_power_ok()