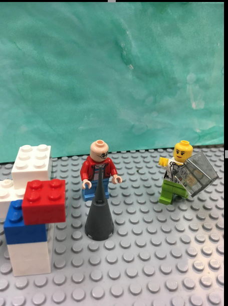
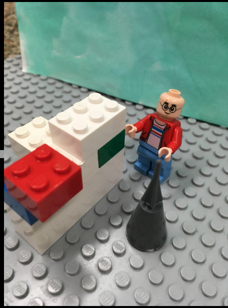
Enabling Successful Onboarding of Scientific Tools Via Development Best Practices

Corinne Bassin
ECS in support of NOAA Fisheries
NOAA Toolbox Coordinator
corinne.bassin@noaa.gov





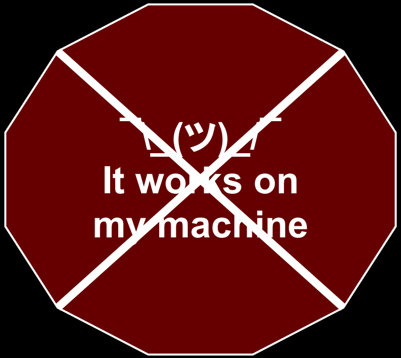
Building Software in Science

Building scientific applications is hard:

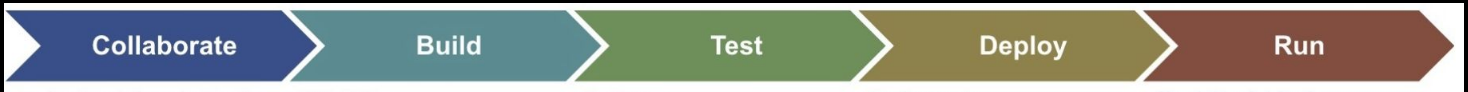
- Complexity involved due to the scientific basis of tools
- Scientific R & D occurs throughout software development cycle
- Developers and users range:
  - Software Developers
  - Scientific Software Developers
  - Software Developing Scientists
  - Scientists
  - Policy Administrators
- Continued development with a variety of collaborators

Onboarding Tools

- Ensuring tools and processes are reproducible
- Ensuring methods and algorithms are testable
- Long term thinking gives way to cloud based applications and infrastructure

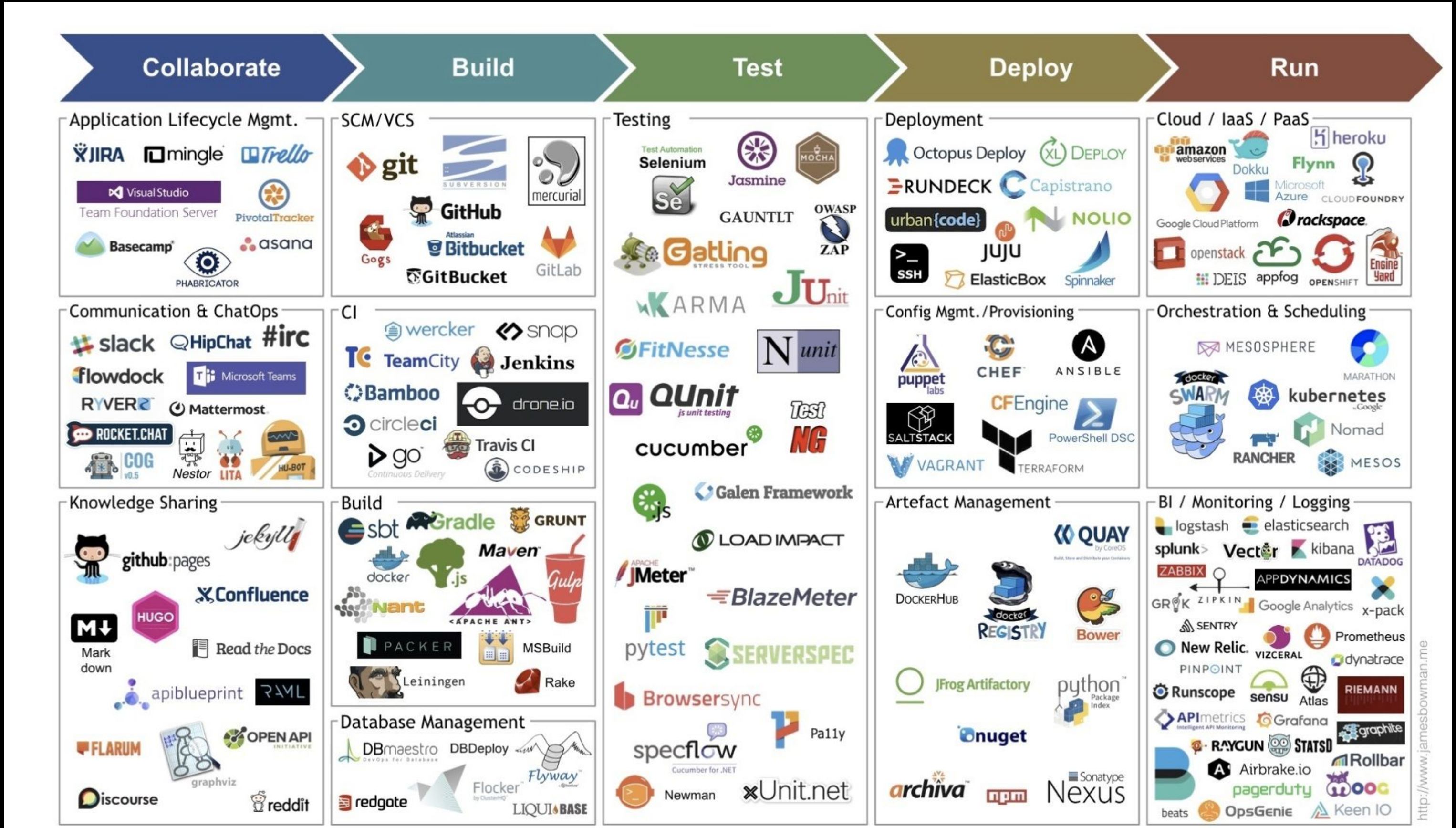


Many of these needs can be accomplished by using software development best practices during development.



What Does Everyone Else Do?

The DevOps tool chain is complicated



Standard Software Development Strategies

- Ensuring best practices build towards a goal
- Determining essential practices
- Use highly supported methods from larger community
- Creating a base level of sound practices to ensure:
    - Collaboration and oversight
    - Reproducibility
    - Scientific integrity
    - Ease of use

Collaboration and Oversight

Working as if there are multiple collaborators:

- Using version control
- Code review
- Status indicators
- Documentation - automated documentation from in-line code

Reproducibility

Identifying what tools and code were used for a given result

- Release structure
- Citation of code
- Unit/Functional testing
- Documentation and examples

Scientific Integrity

- References to internal algorithms
- Modules to run algorithm and compare between models
- Sound science/statistical approaches
- Citation of work
- Publishing work

Ease Of Use

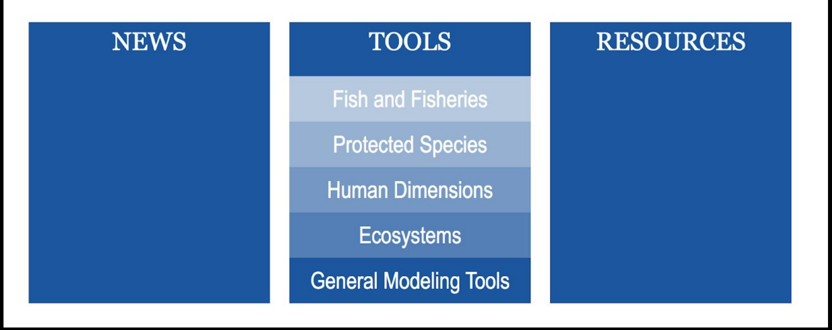
- Standardized input & output (.json format)
- Complementary interface standardization (GUI, API, workflows)
- Cloud-based virtual instances
- Infrastructure for multi-model inference

Think ahead about what onboarding will look like for your tool and organization.

Take the time to invest early in architecture and documentation strategies.

NOAA Fisheries Integrated Toolbox

https://noaa-fisheries-integrated-toolbox.github.io/



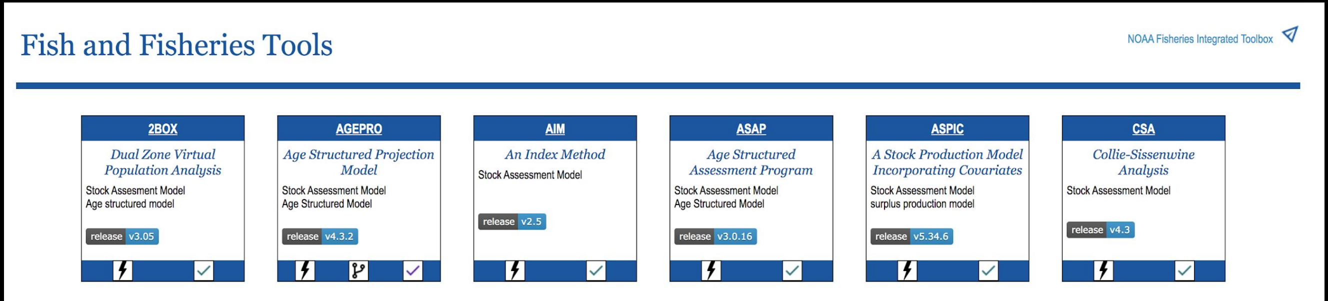
NEWS

TOOLS

- Fish and Fisheries
- Protected Species
- Human Dimensions
- Ecosystems
- General Modeling Tools

RESOURCES

https://nmfs-fish-tools.github.io/



Fish and Fisheries Tools

NOAA Fisheries Integrated Toolbox

2BOX
Dual Zone Virtual Population Analysis
Stock Assesment Model
Age structured model
release v3.05

AGEPRO
Age Structured Projection Model
Stock Assessment Model
Age Structured Model
release v4.3.2

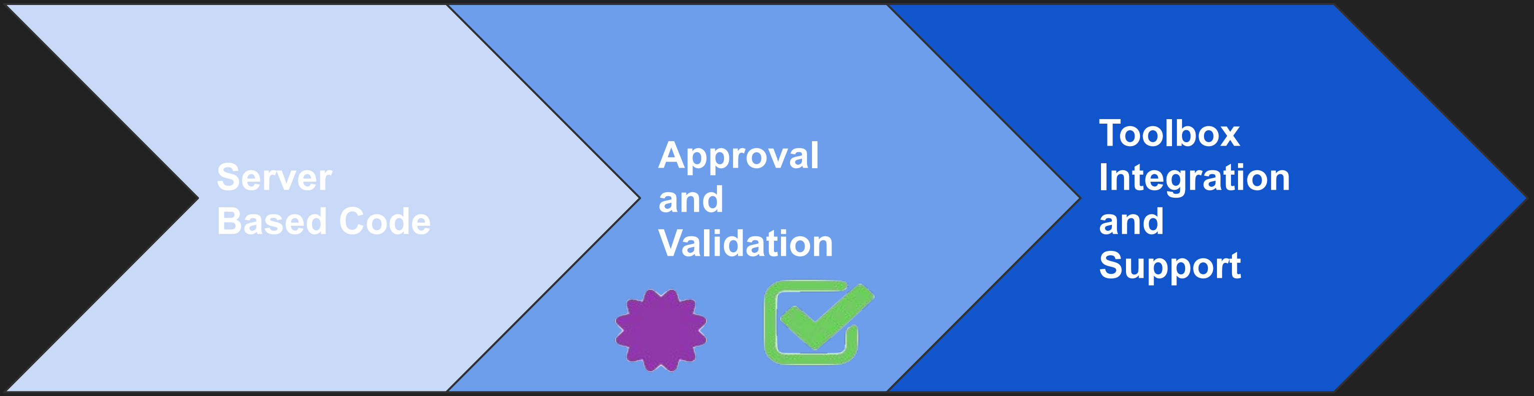
AIM
An Index Method
Stock Assessment Model
release v2.5

ASAP
Age Structured Assessment Program
Stock Assessment Model
Age Structured Model
release v3.0.16

ASPIC
A Stock Production Model Incorporating Covariates
Stock Assessment Model
surplus production model
release v5.34.6

CSA
Collie-Sissenwine Analysis
Stock Assessment Model
release v4.3

Overview of Architecture and goals



Server Based Code

Approval and Validation

Toolbox Integration and Support

Request Onboarding
NOAA Github Policy walk through for NOAA developers
Version Control Implementation (includes Readme and License)
Metadata form about tool
Coding Standards best practices cheat sheet
Code Repository Stats (active/inactive, latest version)
DOI for referencing code (https://www.zenodo.org or similar)

Link to Tool posted on Organizational web page
Basic support level determined

CII Badge (https://bestpractices.coreinfrastructure.org/en)
Documentation/User manual
Tutorials
Unit and Functional Testing
Integrated testing (Travis CI)
Journal of Open Source Software
Standard Inputs/Outputs

Support Levels

Internal active

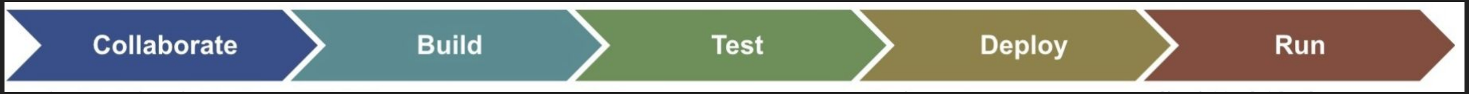
External active

Internal inactive

External inactive

Short -> long term vision

- Code in one place, under version control, with standard repo structure
- Supported code meets minimum approval standards
- Support development via resources/training
- Standardized input & output (.json format)
- Complementary interface standardization (GUI, API, workflows)
- Cloud-based virtual instances
- Infrastructure for multi-model inference



Collaborate → Build → Test → Deploy → Run

- Technology organizations continue to streamline these processes and develop tools to enable best practices.
- Supporting scientists and developers with resources on best practises will make onboarding simpler and faster.
- Creating robust systems and architectures that take into account the likelihood for changes in development strategies and use case scenarios will enable long term use of scientific tools.