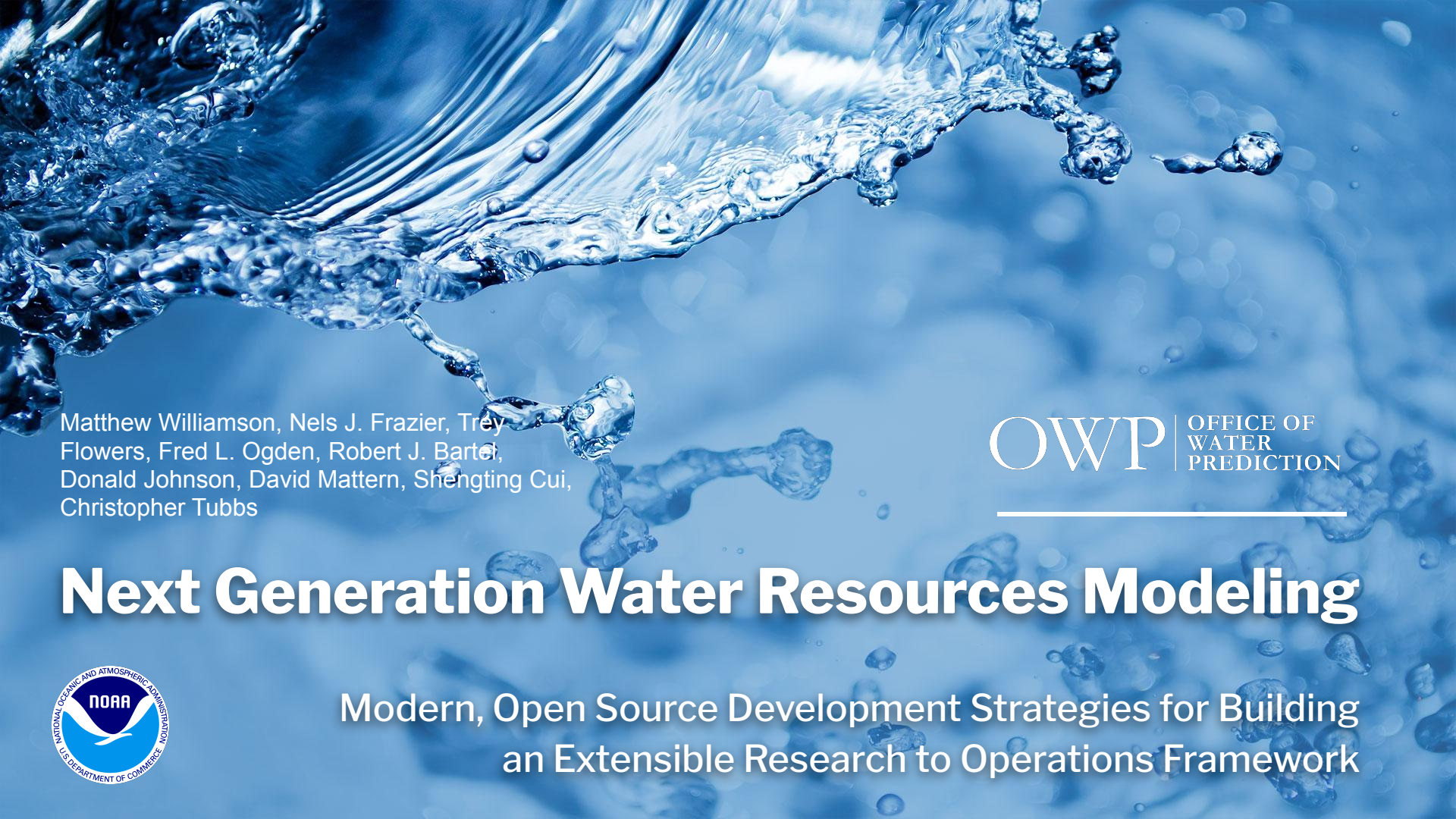


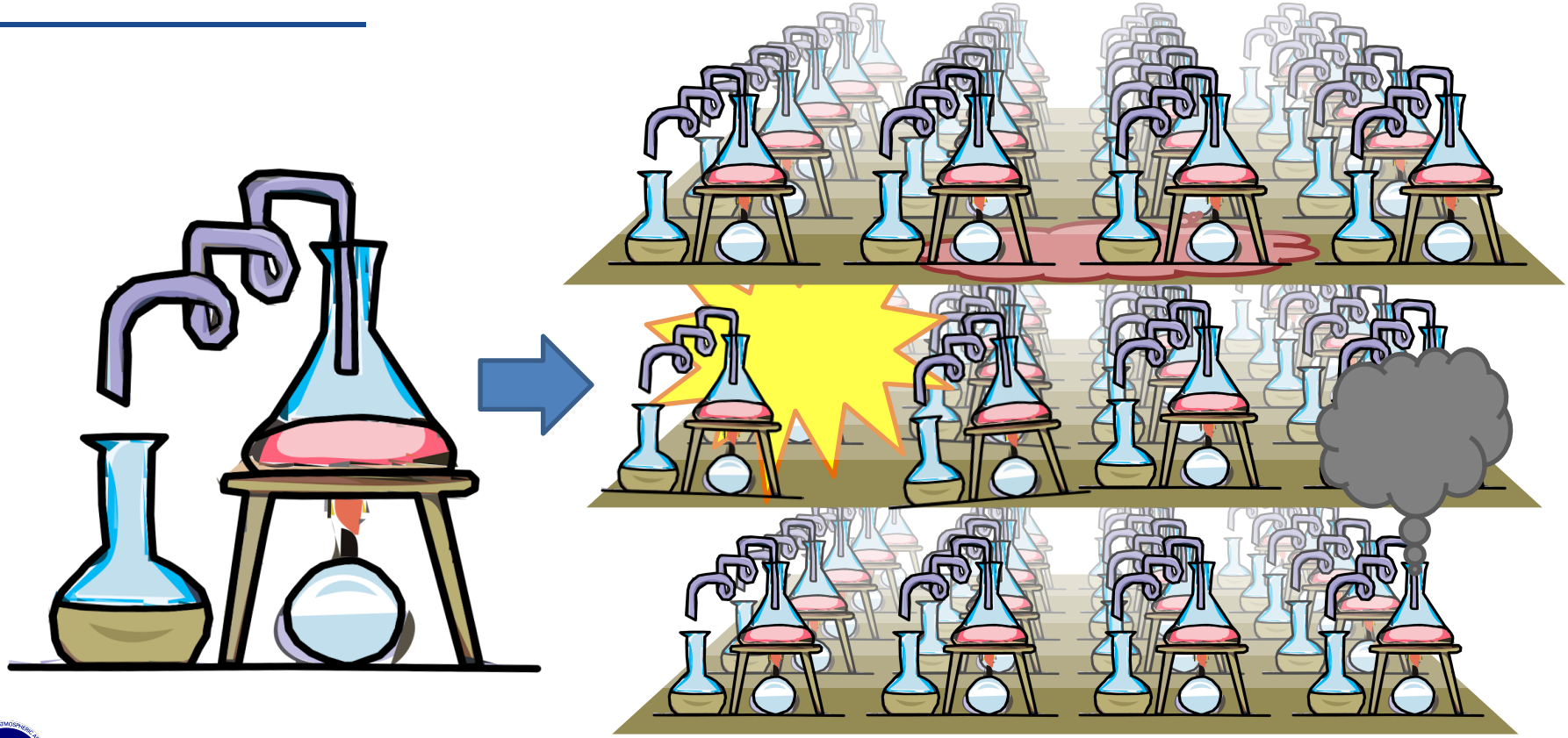
Matthew Williamson, Nels J. Frazier, Trey Flowers, Fred L. Ogden, Robert J. Bartel, Donald Johnson, David Mattern, Shengting Cui, Christopher Tubbs

OWP | OFFICE OF WATER PREDICTION

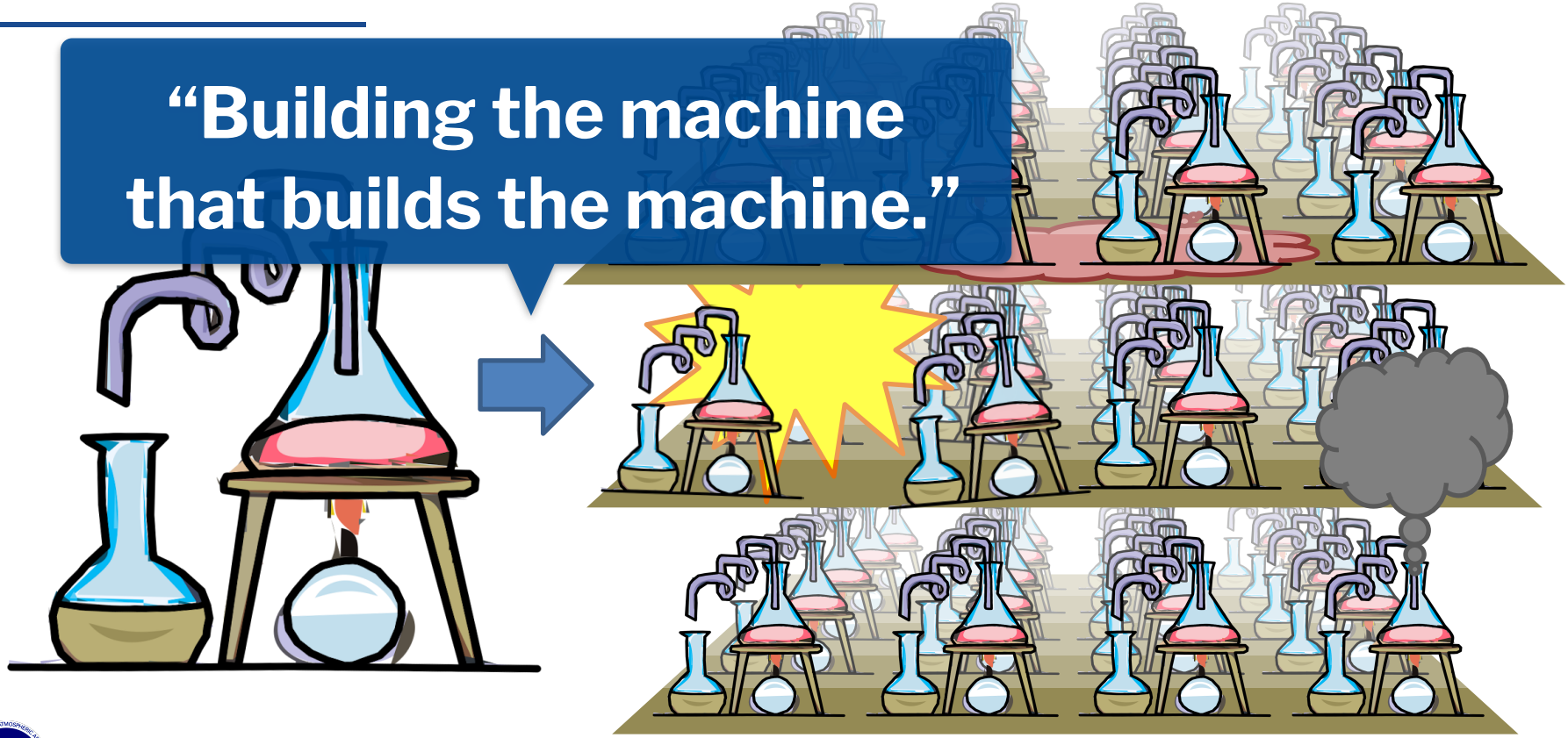
Next Generation Water Resources Modeling

Modern, Open Source Development Strategies for Building an Extensible Research to Operations Framework

Science at Scale



Science at Scale



Our Solution

- Open Source Project
- Iterative, Agile development
- Modularity emphasis
- Open, standard interfaces (HY_Features, BMI)
- Modern software development best practices

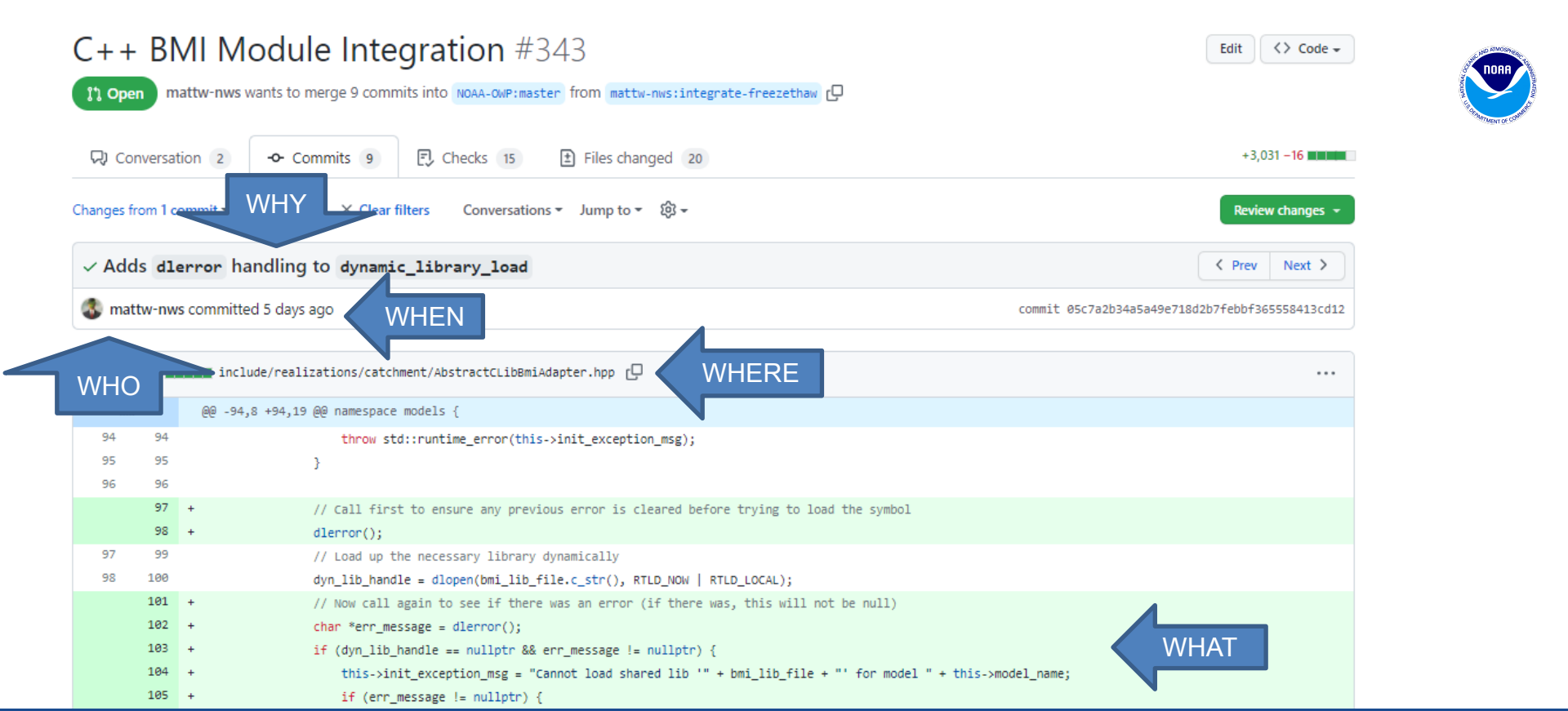
Tools

- Git
  - Version Control
- GitHub
  - Shared Version Control / Open Source Development
  - Issue Tracking / Feature Requests
  - Developer Workflow (Pull Requests / Merge Strategy)
- GoogleTest
  - Test writing and mocking framework
- GitHub Workflows
  - Test Automation / Continuous Integration
- Standard interfaces and libraries
  - BMI
  - HY_Features
  - Boost
  - UDUNITS-2

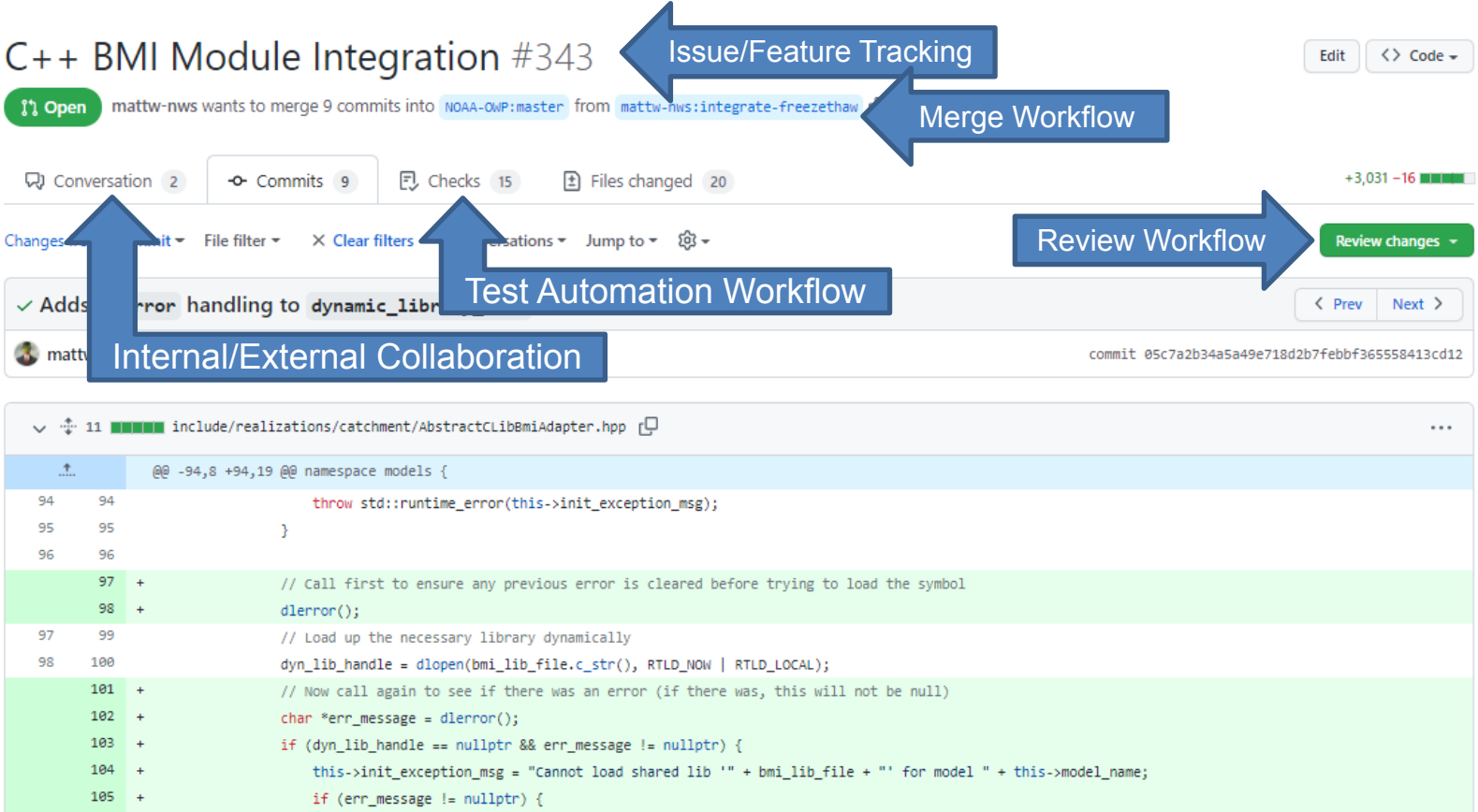
Modularity and Reuse

- BMI (Basic Model Interface)
  - Uniform, generalized interface for model implementation
  - Allows the framework to be completely ignorant of the models it is running
  - Model modules written in any of 4 languages and can reside in their own libraries
  - https://csdms.colorado.edu/wiki/BMI
- HY_Features
  - OGC Standard
  - Describes graph-based hydrofabric structure and its features
  - Will provide an interface for hydrofabric and routing modules
  - http://opengeospatial.github.io/HY_Features/
- Others
  - Boost
  - UDUNITS-2

Why borrow when you can steal?

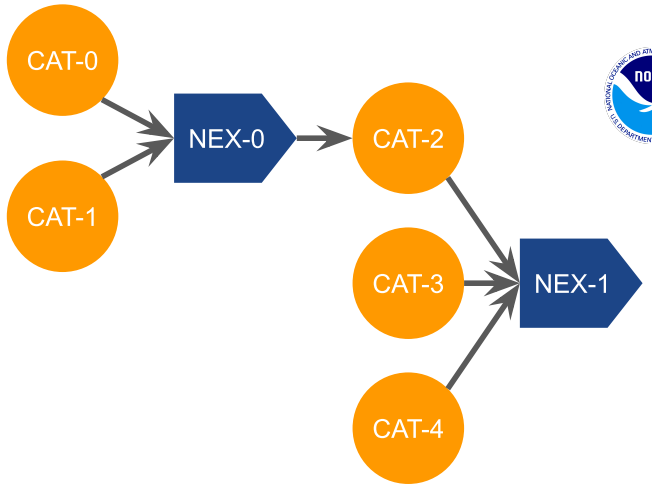


Version Control / Source Code Control (Git)



Collaboration and Developer Workflow (GitHub)

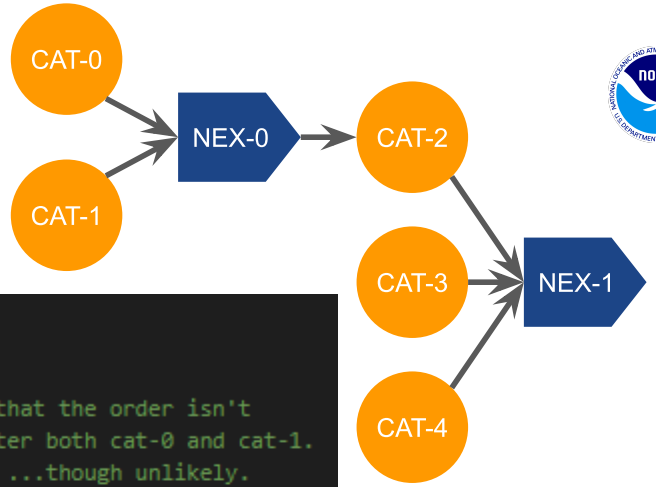
```
132
    Nels Frazier, 7 months ago | 1 author (Nels Frazier)
133 class Network_Test2 : public Network_Test, public ::testing::Test{
134 public:
135   Network_Test2(){}
136   void SetUp(){
137     this->add_catchment("cat-0", "nex-0");
138     this->add_catchment("cat-1", "nex-0");
139     this->add_nexus("nex-0", "cat-2");
140     this->add_catchment("cat-2", "nex-1");
141     this->add_catchment("cat-3", "nex-1");
142     this->add_catchment("cat-4", "nex-1");
143     this->add_nexus("nex-1");
144     n =  Network(this->get_fabric());
145   }
146 };
147
```



Automated Unit Testing (GoogleTest)

```
132
        Nels Frazier, 7 months ago | 1 author (Nels Frazier)
133     class Network_Test2 : public Network_Test, public ::testing::Test{
134     public:
135       Network_Test2(){}
136       void SetUp(){
137         this->add_catchment("cat-0", "nex-0");
138         this->add_catchment("cat-1", "nex-0");
139         this->add_nexus("nex-0", "cat-2");
140         this->add_catchment("cat-2", "nex-1");
```

```
291
292   TEST_F(Network_Test2, test_catchments_filter)
293   {
294     //This order IS IMPORTANT, it should be the topological order of catchments.  Note that the order isn't
295     //guaranteed on the leafs, the only guarantees for this test is that cat-2 comes after both cat-0 and cat-1.
296     //Technically, this is a valid topological order: cat-0, cat-3, cat-1, cat-4, cat-2 ...though unlikely.
297     auto catchments = n.filter("cat");
298     //auto catchments_it = catchments.begin();
299     for(const auto& id : n) std::cout<<"topo_id: "<<n.get_id(id)<<std::endl;
300     for(const auto& id : catchments) std::cout<<"id: "<<id<<std::endl;
301     auto cat0_it = std::find(catchments.begin(), catchments.end(), "cat-0");
302     auto cat1_it = std::find(catchments.begin(), catchments.end(), "cat-1");
303     auto cat2_it = std::find(catchments.begin(), catchments.end(), "cat-2");
304     std::cout << "cat-0 to cat-2 distance: " << std::distance(cat0_it, cat2_it) << std::endl;
305     std::cout << "cat-1 to cat-2 distance: " << std::distance(cat1_it, cat2_it) << std::endl;
306     ASSERT_TRUE( std::distance(cat0_it, cat2_it) > 0);
307     ASSERT_TRUE( std::distance(cat1_it, cat2_it) > 0);
308     ASSERT_TRUE( cat2_it != catchments.end() );
309   }
```



Automated Unit Testing (GoogleTest)

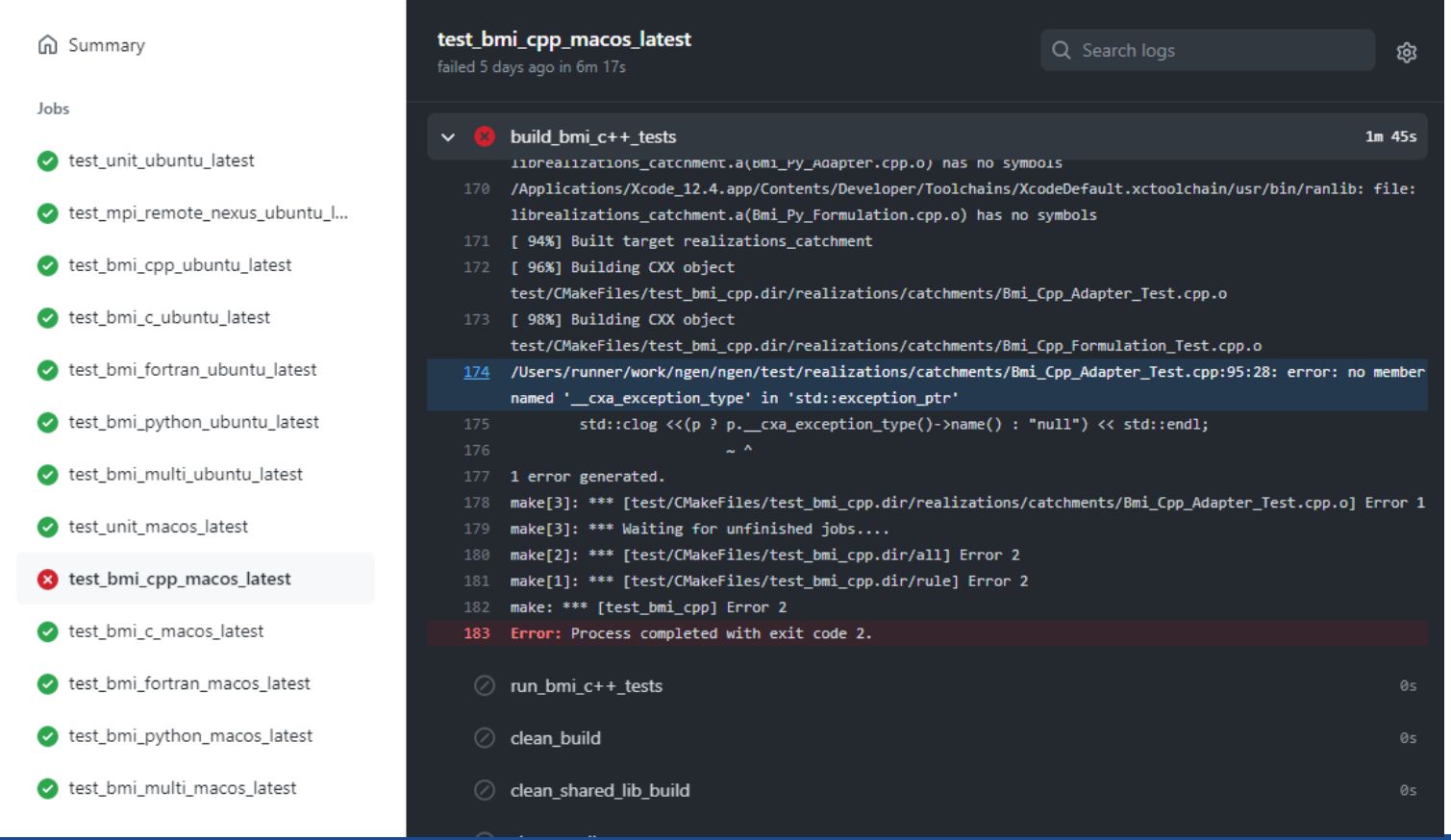
```
const std::string EXAMPLE_1 = "{ "
    "\"global\": { "
      "\"formulations\": [ "
          "{"
            "\"name\": \"tshirt\", "
            "\"params\": {"
              "\"maxsmc\": 0.81, "
              "\"wltsmc\": 1.0, "
              "\"satdk\": 0.48, "
              "\"satpsi\": 0.1, "
              "\"slope\": 0.58, "
              "\"scaled_distribution_fn_shape_parameter\": 1.3, "
              "\"multiplier\": 1.0, "
              "\"alpha_fc\": 1.0, "
              "\"Klf\": 0.0000672, "
              "\"Kn\": 0.1, "
              "\"nash_n\": 8, "
              "\"Cgw\": 1.08, "
              "\"expon\": 6.0, "
              "\"max_groundwater_storage_meters\": 16.0, "
              "\"nash_storage\": [ "
                "1.0, "
                "1.0, "
                "1.0, "
                "1.0, "
                "1.0, "
                "1.0, "
                "1.0, "
                "1.0 "
              "], "
              "\"soil_storage_percentage\": 1.0, "
              "\"groundwater_storage_percentage\": 1.0, "
              "\"timestep\": 3600, "
              "\"giuh\": { "
                "\"giuh_path\": \"./test/data/giuh/GIUH.json\", "
                "\"crosswalk_path\": \"./data/crosswalk.json\" "
              "} "
```

```
TEST_F(Formulation_Manager_Test, basic_run_1) {
    std::stringstream stream;
    stream << fix_paths(EXAMPLE_1);

    std::ostream* raw_pointer = &std::cout;
    std::shared_ptr<std::ostream> s_ptr(raw_pointer, [](void*) {});
    utils::StreamHandler catchment_output(s_ptr);

    realization::Formulation_Manager manager = realization::Formulation_Manager(stream);

    this->add_feature("cat-52");
    this->add_feature("cat-67");
    manager.read(this->fabric, catchment_output);

    ASSERT_EQ(manager.get_size(), 2);

    std::map<std::string, std::map<long, double>> calculated_results;

    pdm03_struct pdm_et_data;
    pdm_et_data.scaled_distribution_fn_shape_parameter = 1.3;
    pdm_et_data.vegetation_adjustment = 0.99;
    pdm_et_data.model_time_step = 0.0;
    pdm_et_data.max_height_soil_moisture_storerage_tank = 400.0;
    pdm_et_data.maximum_combined_contents = pdm_et_data.max_height_soil_moisture_storerage_tank / (1

    std::shared_ptr<pdm03_struct> et_params_ptr = std::make_shared<pdm03_struct>(pdm_et_data);

    double dt = 3600.0;

    for (std::pair<std::string, std::shared_ptr<realization::Formulation>> formulation : manager) {
        if (calculated_results.count(formulation.first) == 0) {
            calculated_results.emplace(formulation.first, std::map<long, double>());
        }
```

Automated Integration Testing (GoogleTest)

```yaml
43
44  - name: Build surfacebmi module
45    run: |
46      cmake -B extern/noah-mp-modular/cmake_build cmake_build -S extern/noah-mp-modular
47      cmake --build extern/noah-mp-modular/cmake_build --target surfacebmi -- -j ${{ env.LINUX_NUM_PROC_CORES }}
48
49  - name: Build cfebmi module
50    run: |
51      cmake -B extern/cfe/cmake_build -S extern/cfe/
52      cmake --build extern/cfe/cmake_build/ --target cfebmi -- -j ${{ env.LINUX_NUM_PROC_CORES }}
53
54  - name: Build iso_c_fortran_bmi module
55    run: |
56      cmake -B extern/iso_c_fortran_bmi/cmake_build -S extern/iso_c_fortran_bmi
57      cmake --build extern/iso_c_fortran_bmi/cmake_build --target iso_c_bmi -- -j ${{ env.LINUX_NUM_PROC_CORES }}
58
59  - name: cmake_init_build
60    run: |
61      export BOOST_ROOT="$(pwd)/boost_1_72_0"
62      [ ! -d "$BOOST_ROOT" ] && echo "Error: no Boost root found at $BOOST_ROOT" && exit 1
63      cmake -B cmake_build -DBMI_C_LIB_ACTIVE:BOOL=ON -DBMI_FORTRAN_ACTIVE=ON -DNGEN_ACTIVATE_PYTHON:BOOL=OFF -S .
64
65  - name: Build ngen
66    run: cmake --build cmake_build --target ngen -- -j ${{ env.LINUX_NUM_PROC_CORES }}
67    timeout-minutes: 15
68
69  - name: Run surfacebmi plus cfebmi
70    run: ./cmake_build/ngen data/catchment_data.geojson "cat-27" data/nexus_data.geojson "nex-26" data/example_bmi_multi_realization_config.json
```

Automated Integration Testing (GitHub Workflows)



GitHub Workflow Test Automation

Summary

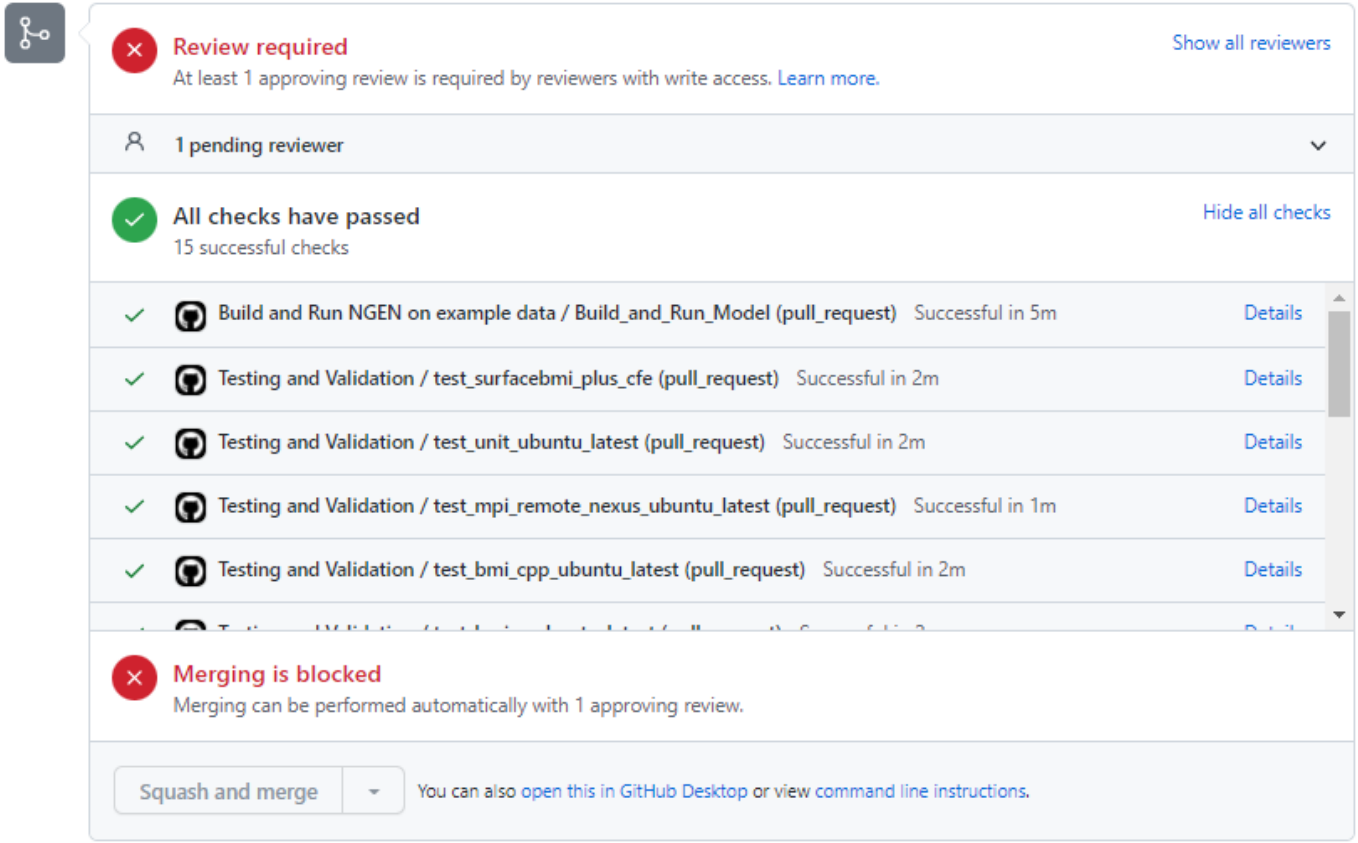
Jobs

- test_unit_ubuntu_latest
- test_mpi_remote_nexus_ubuntu_l...
- test_bmi_cpp_ubuntu_latest
- test_bmi_c_ubuntu_latest
- test_bmi_fortran_ubuntu_latest
- test_bmi_python_ubuntu_latest
- test_bmi_multi_ubuntu_latest
- test_unit_macos_latest
- test_bmi_cpp_macos_latest
- test_bmi_c_macos_latest
- test_bmi_fortran_macos_latest
- test_bmi_python_macos_latest
- test_bmi_multi_macos_latest

“What tests? That’s what the users are for!”

Brady Holt – no changes– CC3.0 – https://creativecommons.org/licenses/by/3.0/

GitHub Workflow Test Automation



Semi-Automated “Peer Review” Before Merge

```
255
256     // Check for and remove any stored sends requests that have completed
257 ▶ 2/2 192   for ( auto i = stored_sends.begin(); i != stored_sends.end(); )
258         {
259 ▶ 2/4 60          MPI_Handle_Error( MPI_Test(&i->mpi_request, &flag, &status) );
260
261 ▶ 1/2 60          if (flag)
262             {
263                 // remove this object from the vector
264 60              i = stored_sends.erase(i);
265             }
266             else
267             {
268 X               ++i;
269             }
270         }
271 132 }
272
273 X   long HY_PointHydroNexusRemote::get
274     {
275 X       return time_step;
276     }
277
278 X   int HY_PointHydroNexusRem
279     {
280 X       return world_rank;
281 ▶ 2/4 72  }
282
```

Donald Johnson 2:43 PM
268 not be reached is interesting

that means we never checked for a send before it arrived

Matthew Williamson - NOAA Federal 2:43 PM
A test on get_timestep might be valuable even if the method isn't needed... to ensure that the timestep is being internally set correctly.

Ah I was unsure what that branch indicated.

Donald Johnson 2:44 PM
thats is why I said we should either add tests or remove the functions

yeah 268 is where we check to see if a recieve has completed and it had not

does that need some comments

Test Coverage Reports

QUIS CUSTODIET IPSOS CUSTODIES?

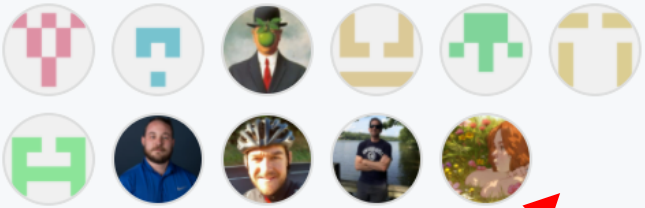
Open Source Development

NOAA-OWP / ngen Public

generated from NOAA-OWP/owp-open-source-project-template

Code Issues 40 Pull requests 9

Contributors 13



See you on GitHub!

YOUR IMAGE HERE

Thank You!

Matt Williamson

matthew.williamson@noaa.gov

https://water.noaa.gov

OWP | OFFICE OF WATER PREDICTION