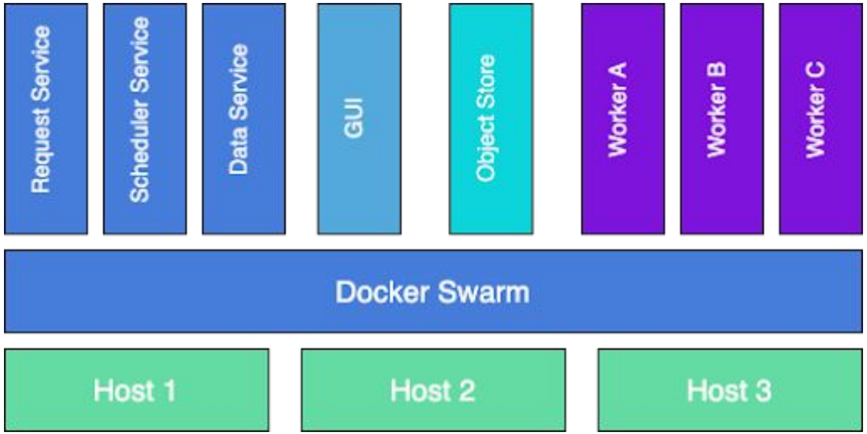
Leveraging the Distributed Model on Demand Platform to Work With Community Models in the Next Generation Water Resources Modeling Framework

Robert Bartel [1], Austin Raney [1], Christopher Tubbs [1], Nels J. Frazier [1], Trey Flowers [2], Shengting Cui [1]

1. *Lynker Technologies, LLC; NOAA Affiliate*
2. *NWS/Office of Water Prediction/National Water Center*

OFFICE OF WATER PREDICTION

Background: What is DMOD?



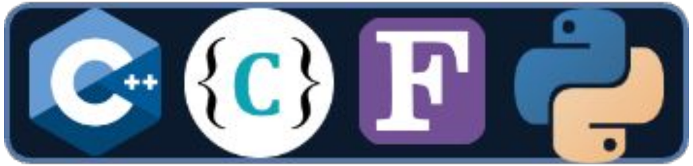
- Prototype platform for facilitating running ngen
- **Key Feature: abstract and manage compute infrastructure and environments**

Full details can be found at: https://github.com/NOAA-OWP/DMOD

What is in an ngen Compute Environment?

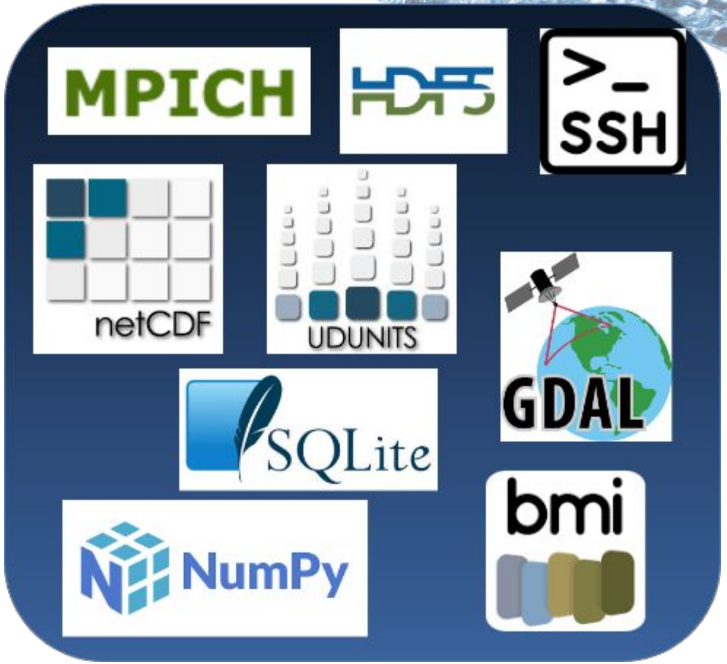
What is in an ngen Compute Environment?

Language Support



What is in an ngen Compute Environment?

Language Support



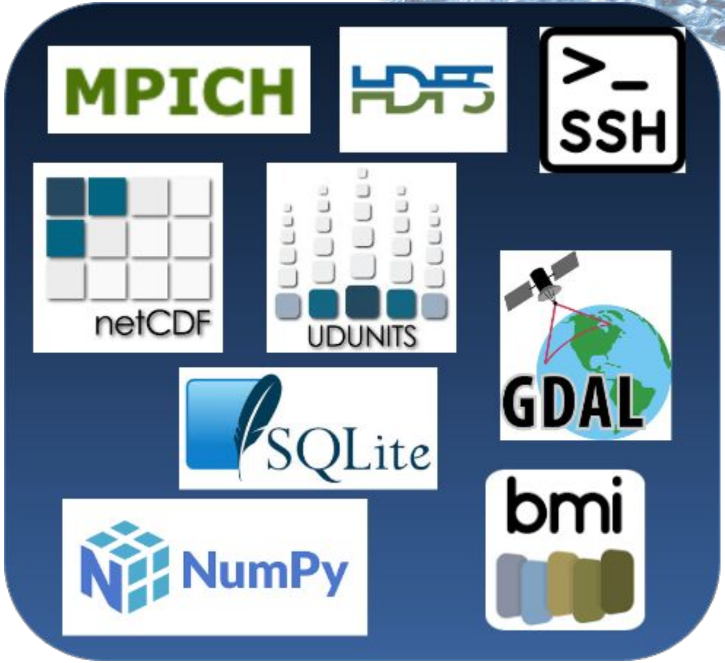
Dependencies

What is in an ngen Compute Environment?

Language Support

git
GCC
CMake

Software Tools and Builds



Dependencies

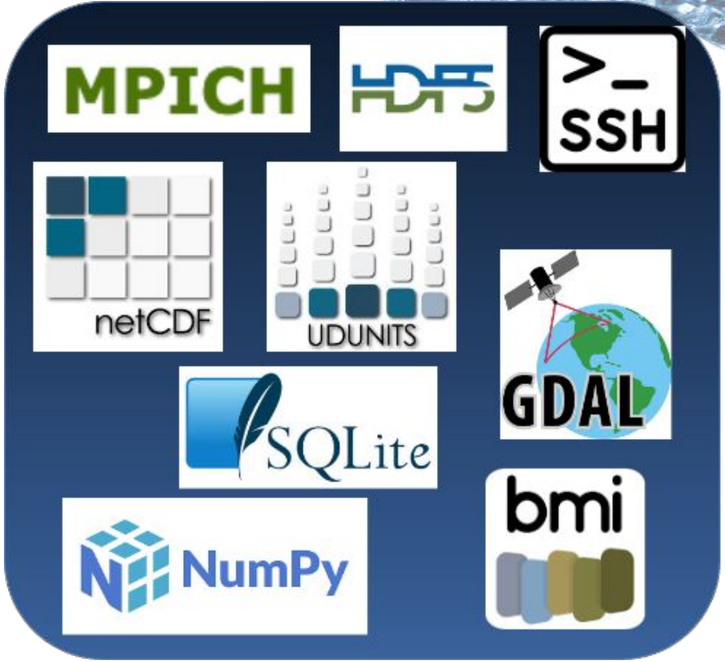
What is in an ngen Compute Environment?

Language Support

DMOD was designed to handle some of this burden for users.

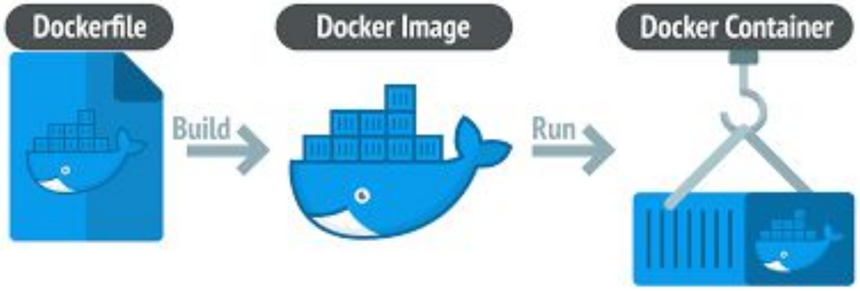
git
GCC
CMake

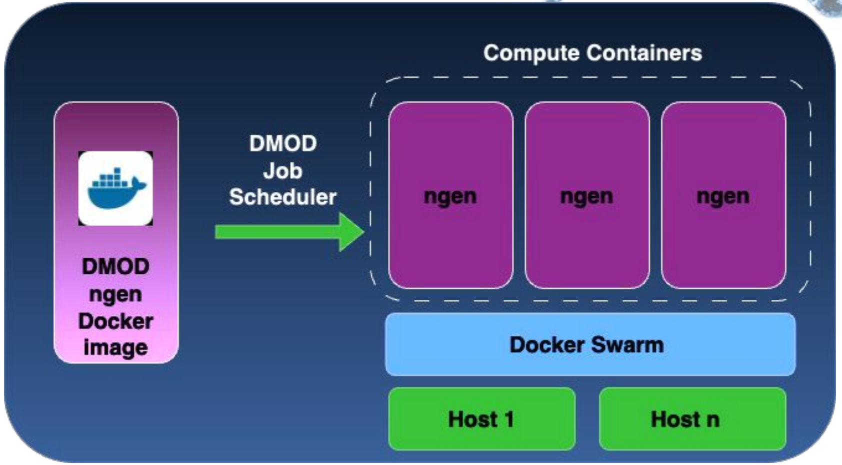
Software Tools and Builds



Dependencies

Package Environments in Docker Containers





- Compute infrastructure as source code
- Tools to *locally* build images and run ngen in containers
- No figuring out how to compile ngen from scratch
- Consistent, reproducible environments

What About BMI Modules?

- Similar dependency and install needs as ngen
- Image source code bundled with some OWP-developed BMI modules
- Works well for many use cases
- Eventually more will be needed

What About BMI Modules?

- Similar dependency and install needs as ngen
- Image source code bundled with some OWP-developed BMI modules
- Works well for many use cases

bmi

What About BMI Modules?

- Similar dependency and install needs as ngen
- Image source code bundled with some OWP-developed BMI modules
- Works well for many use cases
- Eventually more modules will be needed
- ngen designed to work with community-developed modules
- DMOD needs to provide the same flexibility

What About BMI Modules?

- Image source code bundled with some OWP-developed BMI modules
- Works well for many use cases
- Eventually more will be needed
- Why?

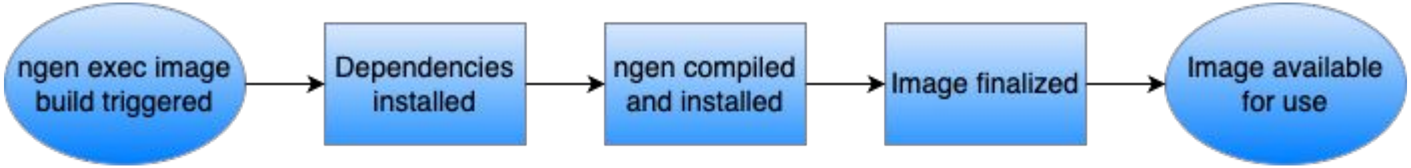
The Solution: Guided Customization

- Not practical to bundle every possible module
- Instead give users ways to customize Docker image
- Lets users insert BMI modules **they** need
- Three different guided methods supported
- Utilizes a special directory in local DMOD file structure
  - `docker/main/ngen/customize`
  - A “bucket” for customization files
  - Ignored by Git
  - Includes a README.md file with documentation

Providing Guided Customization

- Not practical to bundle every possible module
- Instead give users ways to customize Docker image
- Lets users insert BMI modules **they** need

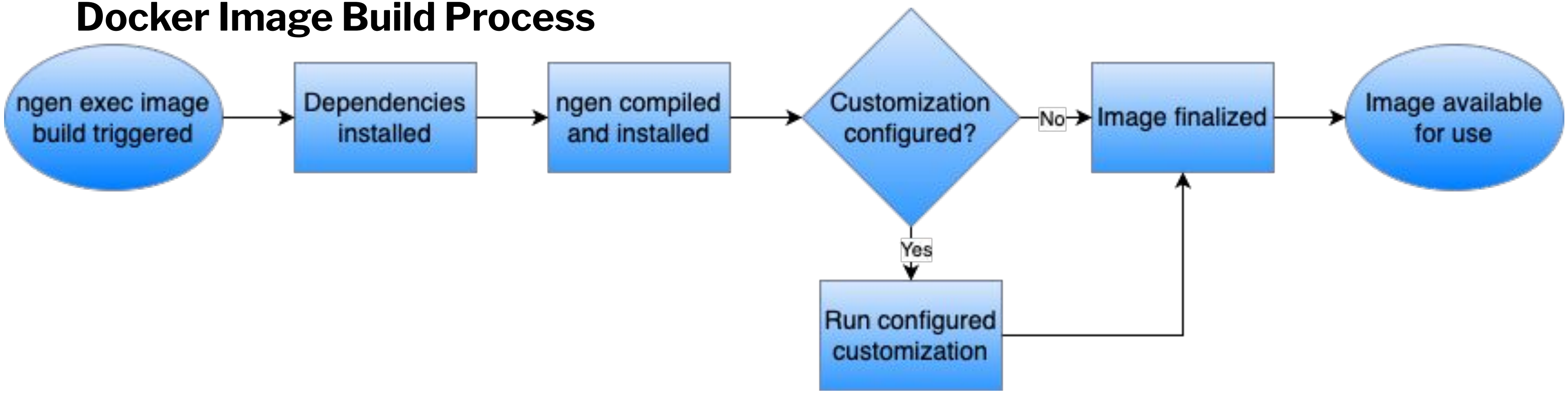
Docker Image Build Process



Providing Guided Customization

- Not practical to bundle every possible module
- Instead give users ways to customize Docker image
- Lets users insert BMI modules **they** need

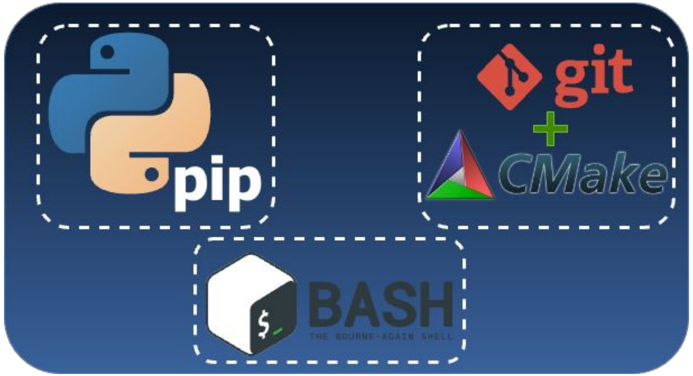
Docker Image Build Process


ngen exec image build triggered
Dependencies installed
ngen compiled and installed
Customization configured?
No
Image finalized
Image available for use
Yes
Run configured customization

Configuring Customization

Available Methods

- A pip requirements file of Python packages to install
- A list of Git repos to auto-build with CMake
- A Bash script for fine-grained control



Configuring Customization

Customize Directory

- Files placed in special directory in local DMOD file structure
  - `docker/main/ngen/customize`
- Ignored by Git
  - Only exists if locally added by user

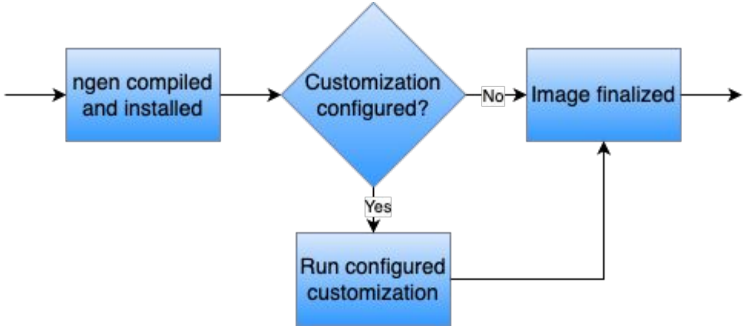
How it Works

- User adds customization file(s) to directory in local copy of DMOD
- When present, contents used automatically by DMOD image build tools
  - No other specialized action needed
  - Presence triggers execution of customization steps at specific build stage

Configuring Customization

Customize Directory

- Files placed in special directory in local DMOD file structure
  - `docker/main/ngen/customize`
- Ignored by Git
  - Only exists if locally added by user



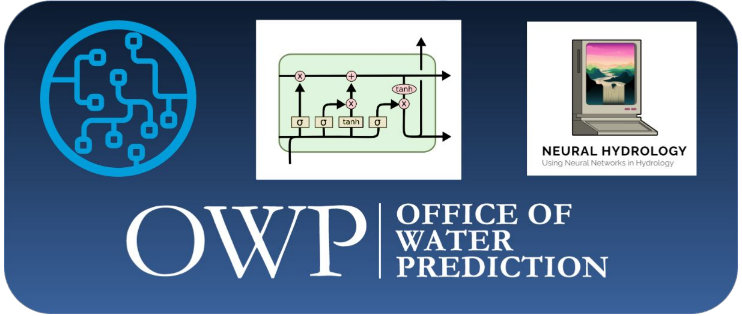
ngen compiled and installed
Customization configured?
No
Image finalized
Yes
Run configured customization

How it Works

- User adds customization file(s) to directory in local copy of DMOD
- When present, contents used automatically by DMOD image build tools
  - No other specialized action needed
  - Presence triggers execution of customization steps at specific build stage

An Example: A Fork of LSTM

- As an example, consider an LSTM BMI module
- Source code available here:
  - https://github.com/robertbartel/owp_lstm
- Analogous to arbitrary community BMI module
  - Not directly from OWP public repo
  - Not pre-integrated into image
- Because some special handling of dependencies is needed, we will use the Bash script



The Steps

Step 1

Create `docker/main/ngen/customize/customize.sh`

Step 2

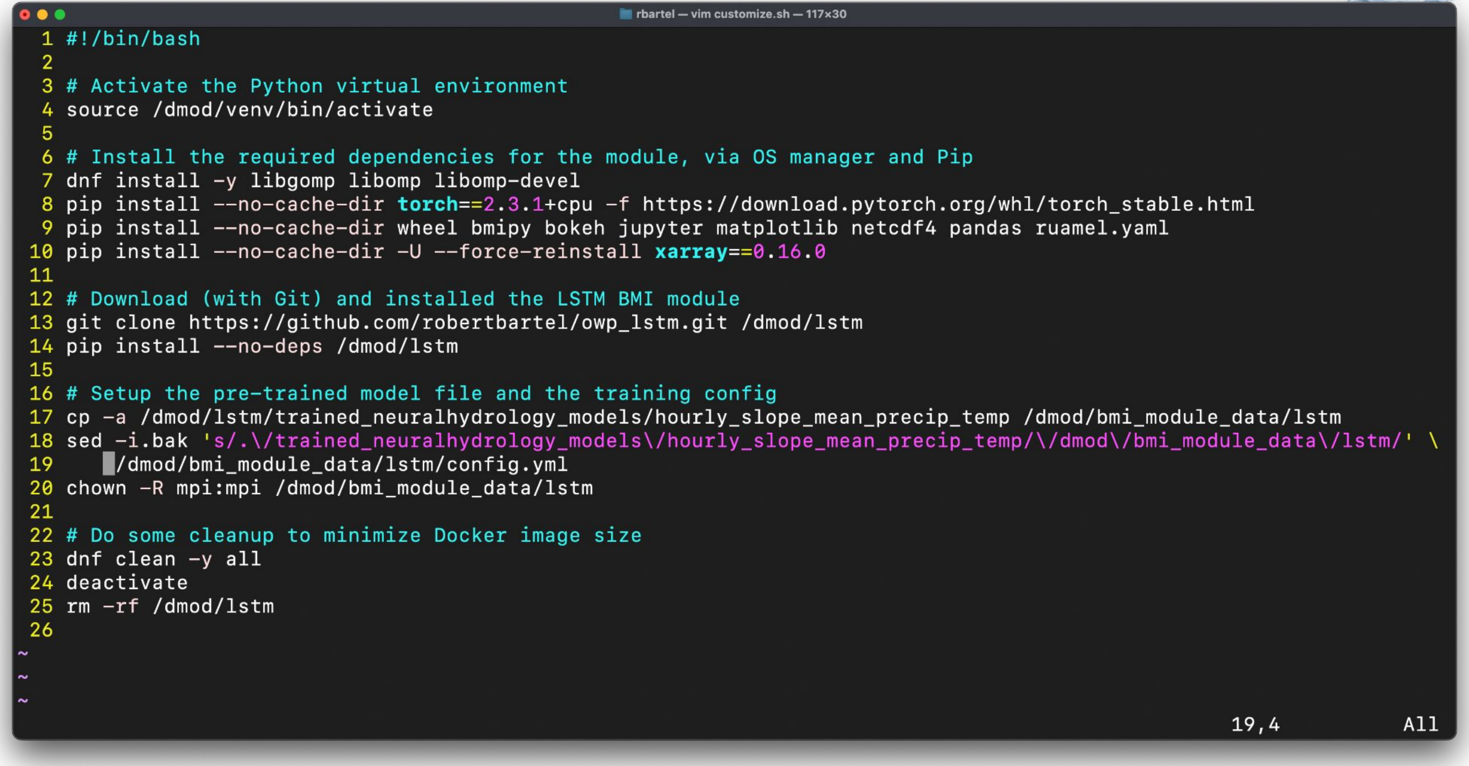
(Re)build the Docker image using DMOD’s *control_stack.sh* helper script:

```
./scripts/control_stack.sh --build-args “ngen” main build push
```

Step 3

Run a DMOD ngen job as normal, using the module in formulations

An Example customize.sh File



```
#!/bin/bash

# Activate the Python virtual environment
source /dmod/venv/bin/activate

# Install the required dependencies for the module, via OS manager and Pip
dnf install -y libgomp libomp libomp-devel
pip install --no-cache-dir torch==2.3.1+cpu -f https://download.pytorch.org/whl/torch_stable.html
pip install --no-cache-dir wheel bmipy bokeh jupyter matplotlib netcdf4 pandas ruamel.yaml
pip install --no-cache-dir -U --force-reinstall xarray==0.16.0

# Download (with Git) and installed the LSTM BMI module
git clone https://github.com/robertbartel/owp_lstm.git /dmod/lstm
pip install --no-deps /dmod/lstm

# Setup the pre-trained model file and the training config
cp -a /dmod/lstm/trained_neuralhydrology_models/hourly_slope_mean_precip_temp /dmod/bmi_module_data/lstm
sed -i.bak 's/.\/trained_neuralhydrology_models\/hourly_slope_mean_precip_temp/\/dmod\/bmi_module_data\/lstm/' \
    /dmod/bmi_module_data/lstm/config.yml
chown -R mpi:mpi /dmod/bmi_module_data/lstm

# Do some cleanup to minimize Docker image size
dnf clean -y all
deactivate
rm -rf /dmod/lstm

~
~
~
```

The Steps (continued)

Step 2

(Re)build the Docker image using DMOD’s *control_stack.sh* helper script:

```
./scripts/control_stack.sh --build-args “ngen” main build push
```

Step 3

Run a DMOD ngen job as normal, using LSTM in formulations.

High Level Results

- Ran a simple ngen job through DMOD using LSTM
- Trained model params:
  - Slope, Elevation, Precipitation, Temperature
- 1 month
  - January 2016
- Hydrofabric VPU-01
  - ~18,000 catchments
- 16 CPU cores
  - Intel 13th Gen i7
- Available memory: ~90 GB memory
  - Max used was ~80 GB
- Job time: ~16 minutes

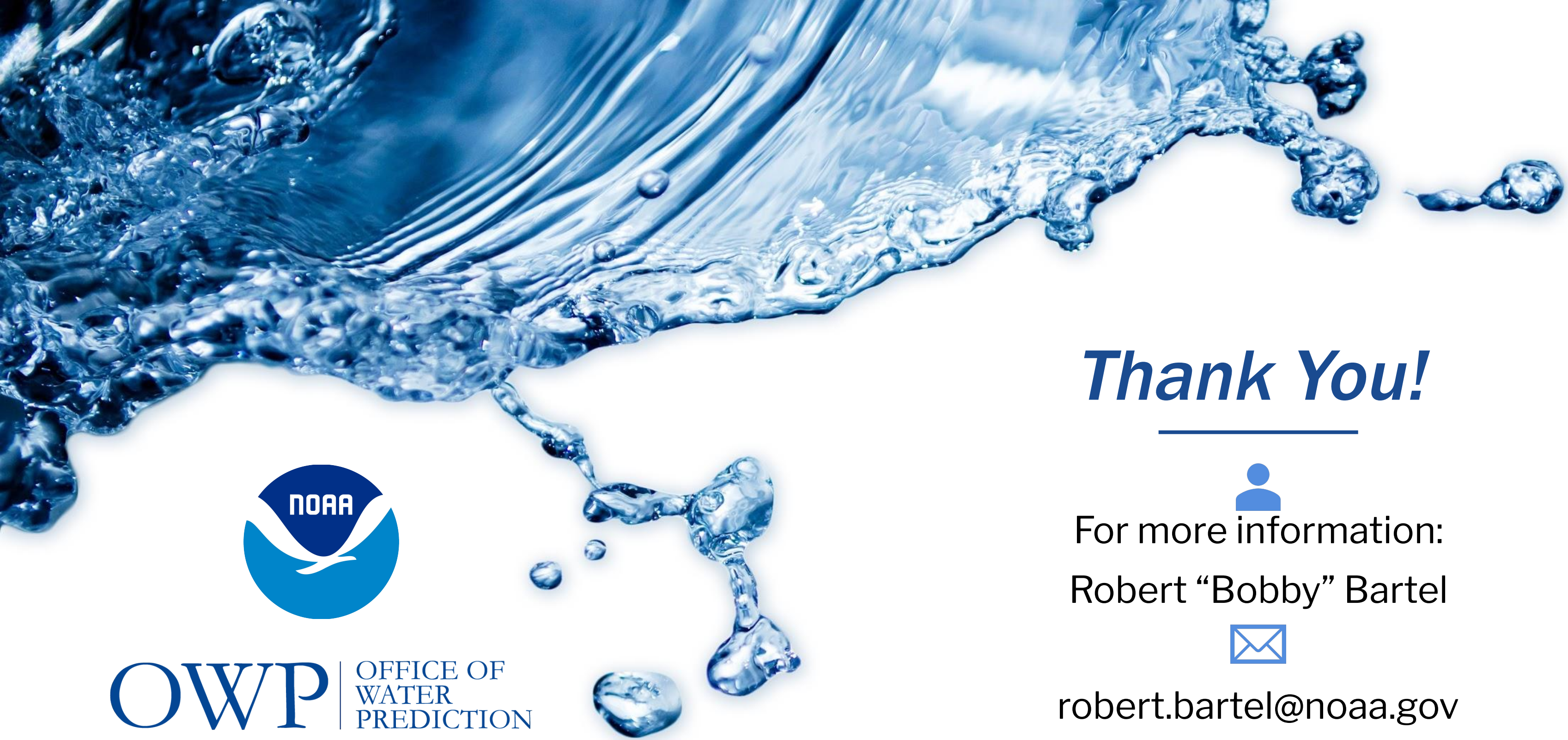
Summary and Future

What DMOD Does

- Automates steps to create consistent compute environments via Docker images and containers
- Enables guided local customization to integrate external BMI modules into images without modifying distributed source code

Future Ideas

- More tools and standards/conventions to “package” models
- Separate image building from DMOD and unify with other work



Thank You!

For more information:

Robert "Bobby" Bartel

robert.bartel@noaa.gov

https://water.noaa.gov

NOAA

OWP | OFFICE OF WATER PREDICTION