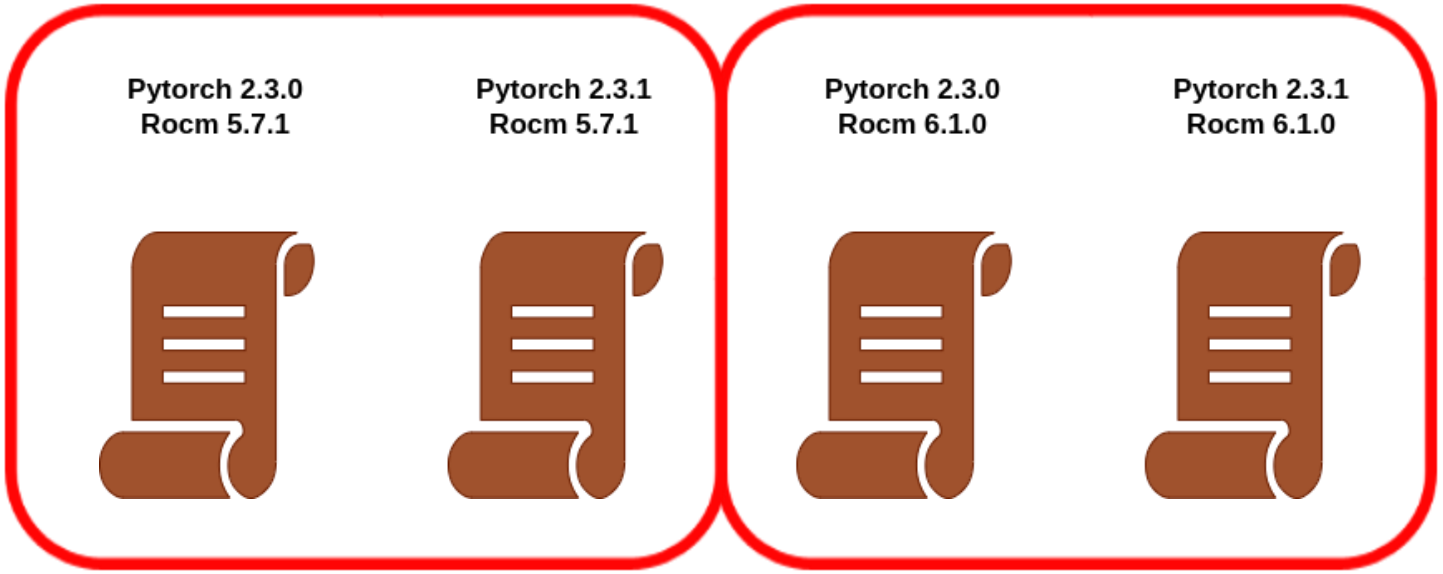
OAK RIDGE National Laboratory

Velocity

Maintenance of container build scripts on multiple systems, backends and distros.

U.S. DEPARTMENT OF ENERGY

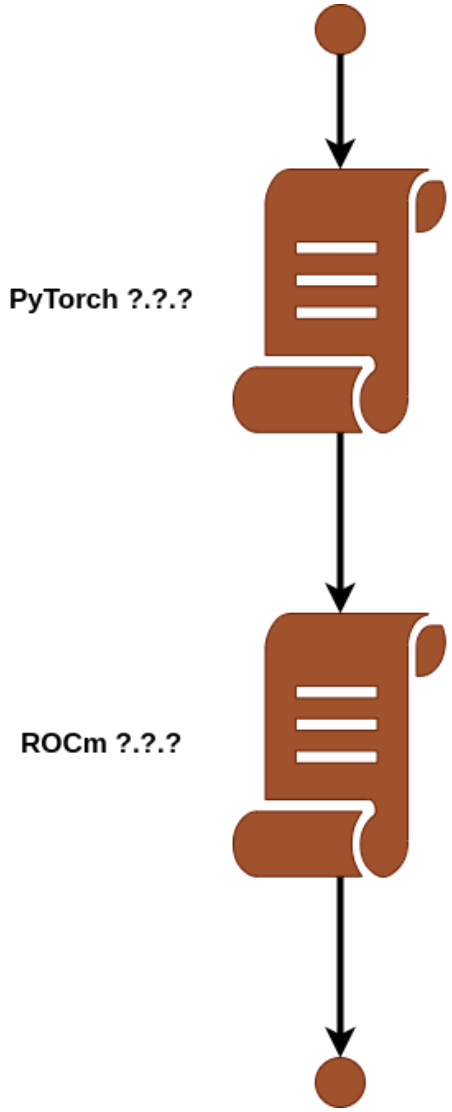
The Problem



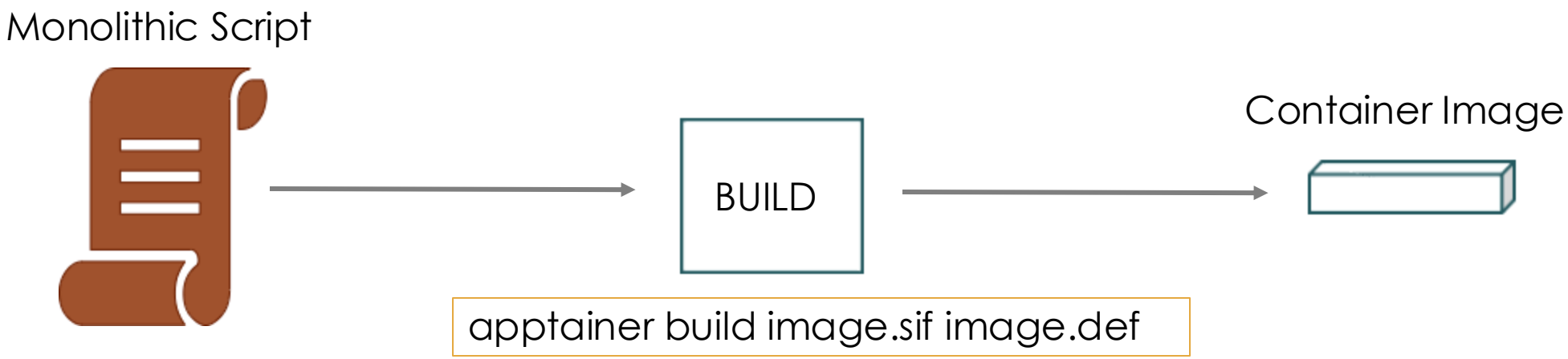
- Monilithic container build scripts maintained by hand
- Explosion of container combinations
- Multiple systems (e.g. Frontier, Andes)
- Multiple container frameworks (e.g. apptainer, podman)
- Multiple Linux distros (e.g. ubuntu, opensuse)

The Solution

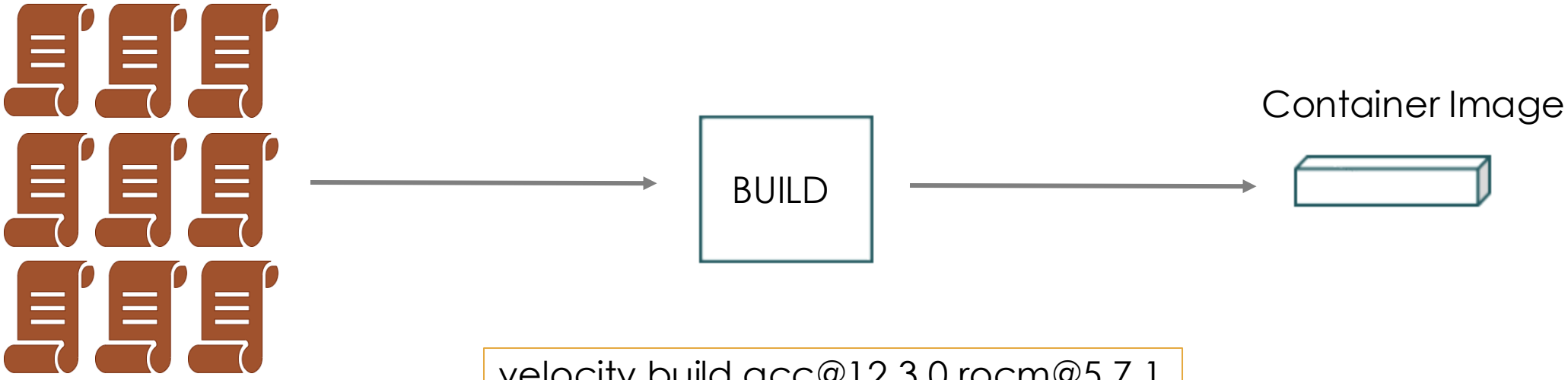
- Generic scripts that can build multiple versions.
- Dynamically chain scripts together for a target container.
- Support multiple container script formats (e.g. podman or apptainer) by using a unified template.
- Allow conditional statements in scripts.



Traditional Container Workflow



Velocity Workflow



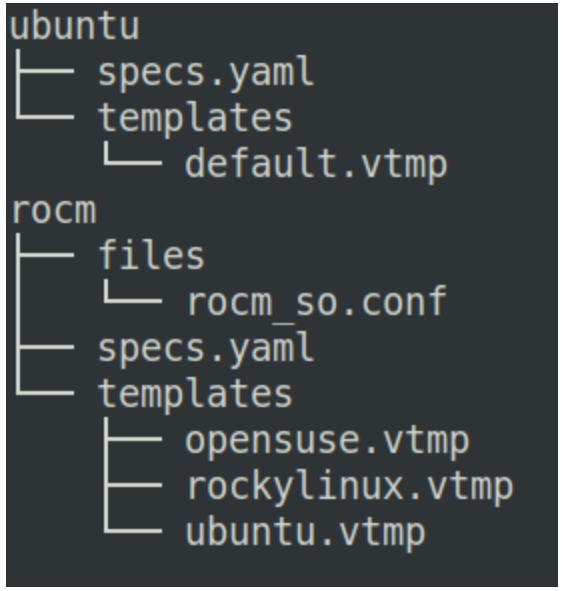

```
velocity build gcc@12.3.0 rocm@5.7.1
```

What is Velocity?

- Tool for organizing scripts & chaining container builds.
- Focuses on manual builds.
- Is not a container runtime (like docker) but a "compatability" layer on top of container runtimes.
- Does not replace having a basic knowledge of container runtime tools.

Defining Container Images

- Image definitions are kept in a directory structure
- Define image metadata (specs.yaml)
- Velocity Templates (*.vtmp)
- Static Files (in "files" directory)



Defining Image Metadata

- Sections:
  - versions
  - dependencies
  - templates
  - arguments
  - variables
  - files
  - prologs

GCC specs.yaml

```yaml
versions:
  - spec:
      - 12.3.0
      - 13.2.0
      - 14.1.0
dependencies:
  - spec: ubuntu
    when: distro=ubuntu
  - spec: opensuse
    when: distro=opensuse
  - spec: rockylinux
    when: distro=rockylinux
variables:
  - name: prefix
    value: "/opt"
```

Templating Build Script

GCC .vtmp

```
@from
    {{ __base__ }}

@run
    ?? distro=opensuse |> zypper --non-interactive install wget tar gzip gcc bzip2 gcc-c++ make && zypper clean --all ??
    ?? distro=ubuntu |> apt -y install wget bzip2 build-essential cmake && apt clean ??
    ?? distro=rockylinux |> dnf -y install wget bzip2 gcc gcc-g++ diffutils && dnf clean all ??
    wget --progress=bar:force https://gcc.gnu.org/pub/gcc/releases/gcc-{{ __version__ }}/gcc-{{ __version__ }}.tar.gz
    tar -xzf gcc-{{ __version__ }}.tar.gz
    cd gcc-{{ __version__ }}
    ./contrib/download_prerequisites
    ./configure --enable-languages=c,c++,fortran --disable-multilib --prefix {{ prefix }}/{{ __version__ }}
    make -j{{ __threads__ }}
    make install
    cd /
    rm -rf gcc-{{ __version__ }}.tar.gz gcc-{{ __version__ }}

@env
    PATH {{ prefix }}/{{ __version__ }}/bin:$PATH
    LIBRARY_PATH {{ prefix }}/{{ __version__ }}/lib64:$LIBRARY_PATH
    LD_LIBRARY_PATH {{ prefix }}/{{ __version__ }}/lib64:$LD_LIBRARY_PATH
    C_INCLUDE_PATH {{ prefix }}/{{ __version__ }}/include:$C_INCLUDE_PATH
    CPLUS_INCLUDE_PATH {{ prefix }}/{{ __version__ }}/include:$CPLUS_INCLUDE_PATH
    MANPATH {{ prefix }}/{{ __version__ }}/share/man:$MANPATH
```

Installation & Configuration

- Create python environment
- $ pip install olcf-velocity
- $ alias velocity="python3 -m velocity"
- "~/.velocity/config.yaml"

```
velocity:
  backend: apptainer
  system: frontier
  distro: ubuntu
  build_dir: /tmp/xjv/velocity
  image_path: /ccs/home/xjv/sw/velocity-images
```

- Can also use environment variables/CLI args

Live Tutorial

https://olcf.github.io/velocity/starting/tutorial.html

Links & References

- Repo: https://github.com/olcf/velocity
- Docs: https://olcf.github.io/velocity/index.html
- OLCF Image Definitions: https://github.com/olcf/velocity-images
- PyPI: https://pypi.org/project/olcf-velocity/
- Python Environments at OLCF: https://docs.olcf.ornl.gov/software/python/index.html#module-usage

Questions?

Thank you!