Title	CWL in Practice: Adopting Common Workflow Language for existing research projects and academic computing clusters
Authors	*Dan Leehr*, Alejandro Barrera, Hilmar Lapp
Affiliation	https://www.genome.duke.edu
Contact	dan.leehr@duke.edu
URL	https://github.com/Duke-GCB/GGR-cwl
License	MIT

The Common Workflow Language provides a blueprint for architecting workflows. With open source implementations and a comprehensive specification, it lowers the barrier to building reusable components and reproducible pipelines for bioinformatics. These pipelines can be run on any implementation supporting the language, and thus are a desirable product for reproducible research.

At Duke Center for Genomic and Computational Biology (GCB), we're using CWL in support of the Genomics of Gene Regulation project. This project aims to comprehensively characterize and better understand the first 12 hours of the glucocorticoid response (GCR). Using gene editing techniques to turn on and off genes, researchers in the Reddy lab plan to elucidate the network of gene regulation involved in the GCR by comparing the results of a number of genomic experiments in each condition. Our task at hand is to develop pipelines for processing experimental data that can be confidently reused and customized.

By building workflows that conform to the CWL standard, not only do we have flexibility to easily modify or replace parts of the GGR workflows, but we'll be able to distribute reproducible pipelines along with publications and data sets. Additionally we benefit from the ability to reuse components shared in the analysis of ChIP-seq, RNA-seq, and DNase-seq experiments.

During the research and development of these CWL pipelines, we've met two discrete challenges: translating existing logic to CWL and integrating this workflows with our high-performance computing resources. In this presentation, we will discuss our approaches to addressing these challenges. We'll cover the details of translating existing control flow and imperative scheduling logic to CWL's declarative language. This includes pragmatic approaches to reusing components for manageable workflows.

We'll also discuss the practical aspects of adopting CWL in an academic HPC environment. Our center operates a 40-node HPC cluster running Slurm for computational genomic workloads. As it is a shared compute environment, it is not suitable to run Docker or virtual machines favored by CWL. However, publishing Docker images makes it easier for consumers to use our workflows. We build Docker images and load modules with helmod, both providing the necessary environment for CWL tools to run. Finally, we'll address how we're extending toil, a workflow engine that supports CWL with Slurm integration, allowing us to scale up CWL workflows to our available compute resources.