Project Requirements

**Overview:** This document contains subjective technical and quantitative requirements for the project according to feedback from intended end users (MITgcm-ECCO researchers, scientists, climate science graduate students, and others interested in using MITgcm model data). This document will be continuously updated as the author sees fit as new requirements are introduced, existing requirements modified, and trade-offs explored throughout the project design process. Since the project’s objectives are to design a product that is robust enough to run a wide array of tasks in the MITgcm-ECCO workflow, it is reasonable to design a framework that divides the tasks across multiple units of software (containers) to prevent one unit of software from taking on a number or intensity of tasks that exceeds memory allocation and crashing. In turn, this necessitates a need for the units of software to communicate the output of these tasks with one another in real time which can be accomplished using a container orchestration platform. Since both the container orchestration platform and its containers are intended to run on user’s host machines, then, there need be host machine hardware and software considerations in the design of the product. **Therefore, the requirements are assigned to four sub-systems integral to the design process, namely, a portable unit of software (containerization) system, a container orchestration system, host machine hardware, and host machine software. In turn, each requirement will be denoted as either a subjective technical requirement (requirement that is technical in scope but has no tangible quantitative values ascribed to it), or as a quantitative requirement (requirement that has quantitative bounds on its operation).**

**Subjective Technical Requirements (In Progress)**

Each of the project’s subjective technical requirements will be enumerated in sections according to their associated subsystem in the final product design as follows:

1. ***Portable Unit of Software (Containerization) System***
2. Containerization system shall be compatible with most frequently used host machine platforms, architectures, and operating systems
3. Containerization system shall come packaged with MITgcm and ECCO libraries and their dependencies for building, running, and interpreting results from model problems

* Where automation is not possible, containerization system shall be able to be easily configured by end user to communicate with host machine for completing tasks in workflow (for example, through optfiles, mpi-tasks etc.)

1. Containerization system shall be documented with user guide that enables straightforward replication of steps necessary to execute tasks in workflow
2. Containerization system shall be open source with readily available documentation for the end user
3. ***Container Orchestration System***
4. Container orchestration system shall be compatible with most frequently used host machine platforms, architectures, and operating systems
5. Container orchestration system shall come packaged with software libraries and their dependencies necessary for configuring communication between containers to run multiple tasks simultaneously

* Where automation is not possible, container orchestration system shall be able to be easily configured by end user to communicate with host machine for completing tasks in workflow (for example, through optfiles, mpi-tasks etc.)

1. Container orchestration system shall be documented with user guide that enables straightforward replication of steps necessary to execute tasks in workflow
2. Container orchestration system shall be open source with readily available documentation for the end user
3. ***Host Machine Hardware***
4. Containerization system shall be computationally inexpensive to run on end user host machines

* System shall reduce amount of RAM occupied by host machine when executing tasks in MITgcm-ECCO workflow
* System shall reduce CPU load of host machine when executing tasks in MITgcm-ECCO workflow

1. ***Host Machine Software***
2. Containerization system shall be computationally inexpensive to run on end user host machines

* System shall reduce compile time necessary to execute tasks in MITgcm-ECCO workflow
* System shall reduce runtime time necessary to execute tasks in MITgcm-ECCO workflow

**Quantitative Requirements (In Progress)**

As test runs of building, running, and analyzing output from MITgcm model problems are performed, document statistics on computational costs (runtime, compile time, CPU load, RAM, etc.) and use these properties to gauge thresholds and objectives for design requirements to minimize these computational expenses.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **System** | **Requirement** | **Thresholds** | **Objectives** | **Units** |
| Containerization Platform |  |  |  |  |
| Container Orchestration Platform |  |  |  |  |
| Host Machine Hardware |  |  |  |  |
| Host Machine Software |  |  |  |  |

**Design Compromise Schematic(s) (In Progress)**

As test runs of building, running, and analyzing output from MITgcm model problems are performed use documented statistics to determine where trade-offs exist in performance to develop design compromise schematics and decision matrices

**Decision Matrices (In Progress)**