Version 0.01

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 17 Nov. 2016 | 0.01 | Created | John Hetrick |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 References 1

1.4 Definitions, Acronyms, and Abbreviations 1

1.4.1 Acronyms 1

1.4.2 Terms and Definitions 1

1.5 Overview 1

2. System Overview (Black Box Context 2

2.1 Actors 2

2.2 Data Exchanges 3

2.2.1 Post Process Msg, 3

2.2.2 Reduction Cmnd 3

2.2.3 Reduction Status Msg 3

2.2.4 Experiment Information 4

2.2.5 Workflow Information 4

2.2.6 Workflow Description Information 4

2.2.7 Heartbeat Msg 4

2.2.8 User Reduction Cmnds 4

3. Interfaces & Protocols 5

3.1 POSTPROCESS.DATA\_Ready 5

3.2 Other Protocol Messages 5

4. Use-Case View (High Level Functional Requirements) 5

4.1 Use Case: Setup Workflow System (NEEDS WORK) 6

4.2 Use Case: Reduce Data Set (NEEDS WORK) 6

4.3 Use Case: Add (CRUD) Workflow Description (Procedure) (NEEDS WORK) 7

4.4 Use Case: Perform New Reduction on Existing Data Set (NEEDS WORK) 7

4.5 Use Case: Rerun Existing Data Set (NEEDS WORK) 8

4.6 Other Relevant Information 8

5. System Architecture (Abstract Components) 9

6. Use-Case Realizations 9

6.1 Use Case: Setup Workflow System (NEEDS WORK) 9

6.1.1 Scenario 1: Prototype (Test) Startup Sequence 9

6.2 Use Case: Reduce Data Set (NEEDS WORK) 10

6.2.1 Scenario 1: First Data Set for New Experiment 10

6.3 Use Case: Rerun Existing Run Set (NEEDS WORK) 11

6.3.1 Scenario 1: 11

6.4 Use Case: Perform New Reduction on Existing Data Set (NEEDS WORK) 11

6.4.1 Scenario 1: 11

6.5 Use Case: Add (CRUD) Workflow Description (NEEDS WORK) 11

6.5.1 Scenario 1: 12

7. Design 12

7.1 WorkflowSystem 12

7.2 WorkflowEngine 12

7.3 WorkflowDescription 12

7.4 Workflow 13

7.4.1 Workflow 13

7.4.2 GroupWF 13

7.4.3 SeqWF 13

7.4.4 TaskStatus 13

7.4.5 WorkflowRepo 13

7.5 Experiment 14

8. Design Patterns, Rules, & Guideline 14

8.1 ID Strategies 14

8.1.1 Data IDs (Meta Data) 14

8.1.2 Experiment IDs 15

8.1.3 Group IDs 15

8.1.4 Sequence IDs 15

8.1.5 Workflow Description IDs 15

8.1.6 Task IDs 17

8.1.7 Workflow IDs 17

8.2 Architectural/Design Rules 17

9. Thoughts, Questions, Notes 18

9.1 Stateless workflow 18

9.2 Workflow State 18

10. To Do 18

10.1 Component Interfaces 18

10.2 Factories 18

10.3 Multi-Process 19

11. Appendix: Raw Model Diagrams 20

11.1 Workflow System Class Diagram 20

11.2 Workflow Engine Class Diagram 21

11.3 Workflow Class Diagram 21

11.4 Workflow Description Class Diagram 22

11.5 Task Class Diagram 22

11.6 Action Class Diagram 23

# Introduction

[The introduction of the **document subject** provides an overview of the entire document. It includes the purpose, scope, definitions, acronyms, abbreviations, references, and overview of this **document**. Adding text here is optional.]

## Purpose

[Specify the purpose of this document]

The purpose of this document is to capture the architecture and design of the . This is a living document which will be updated as needed to further define and clarify key decisions?

## Scope

[A brief description of the scope of this **document**; what topics it is associated with, and anything else that is affected or influenced by this document.]

This document is specific for the project. This document covers the high level architecture of the system and the design of the system along with architecture/deisgn rules that are to be used in the development of the system.

This document does cover some of the system requirements but is not intended to cover them comprehensively. The requirements are largely functional and in the form of Use Cases and or scenarios that are to be implemented. These provide the basis implementing the design.

## References

1. Presentation: Overview of current workflow, Mathieu Doucet, Arv2.pptx
2. Book: Design Patterns Elements of Reusable Object-Oriented Software, By Eric Gamma, Richard, Helm, Ralph Johnson, John Vlissides
3. Reference, <link>

## Definitions, Acronyms, and Abbreviations

[This subsection provides the definitions of all terms, acronyms, and abbreviations required to properly interpret the **document**.  This information may be provided by reference to the project’s Glossary.]

This section contains the terms, acronyms and their definitions that are specific to understanding Automated Reduction Workflow.

### Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| AR | Automated Reduction |
|  |  |
|  |  |
|  |  |

### Terms and Definitions

| **Term** | **Definition** |
| --- | --- |
| Data Set | Any bound set of data created. In this context it is a set of data produced by an instrument for an experiment that is to be reduced |
|  |  |

## Overview

[This subsection describes what the rest of the **document** contains and explains how the document is organized.]

<Brief overview of the Automated Reduction Workflow>

# System Overview (Black Box Context

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

This section represents the context of the system under development. Of importance is the identification of the system, the actors that interact with the system, and anything that is exchanged with the system.

<context diagram>

A screenshot of a cell phone

Description automatically generated

The System is the system that manages the reduction of the data that is created as a result of neutron spallation experiment.

## Actors

This is a list of actors to the system. This includes any people, other computer systems, and processes that interact with this system.

**Translation Service** – The Translation Service is entity that tells the System there is a file of experiment run data (data set) from an instrument ready for processing. The following is information is exchanged by the actor:

* Message containing the location of the file to be processes

**Reducer (Post Processing)** – The Reducer is a post processing system responsible for performing the actual reduction of a data set. It takes a series of messages for a data set that direct different aspects of the reduction process.

**Monitor System** – The monitor system is the system that someone who is interested in monitoring the instruments experiments are run on and/or the experiments themselves. Currently the provides heat beat messages about its health and the status of reductions to the Monitor.

**Experiment Repo** – The Experiment Repo (repository) is a notional repository that holds information about experiments. The gets experiment information from the repository and updates the repository with workflow information.

**Workflow Description Source** – The Workflow Description Source is a notional external source where Workflow Descriptions are held. The gets the workflow descriptions from this source to be paired with data set in a workflow. It is not clear if this will be CSV file(s), JSON file(s) or a more persistent storage like a database. It is also unclear at this time if there will be a system to help create workflow descriptions.

**User (TBD)** – This actor is very abstract and notional at this time. Currently the has not user interface. However, it is yet unclear where the initiation of some behavior required of the system will come from. Specifically the user cases that involve the re-reducing of data sets. This is a place holder actor until this is better understood.

## Data Exchanges

This section identifies the sets of information that is exchanged with the . In the model these are referred to as I/O Entities. I/O Entities are <<sent>> by a actor to the system, <<receive>> by an actor from the system, or <<sent/receive>> by and actor to/from the system.

### Post Process Msg,

Actor: TranlationService,

Action: <<send>>

The Post Process Data Msg entity is a message from the Translation Service to the telling the there is a data set to process (reduce)

### Reduction Cmnd

Actor: Reducer (Post Processor)

Action: <<receive>>

The Reduction Cmnd entity is a message from the to the Reducer system telling the Reducer to perform a specific task in the reduction process. Currently 3 different reduction commands

* CATALOG.DATA\_READY
* REDUCTION.DATA\_READY
* REDUCTION\_CATALOG.DATA\_READY

### Reduction Status Msg

Actor: Reducer (Post Processor)

Action: <<send>>

The Reduction Status Msg entity is a message from the Reducer to the indicating the status of executing a specific reduction task. Currently identified reduction status messages are:

* CATALOG.STARTED
* CATALOG.COMPLETE
* REDUCTION.STARTED
* REDUCTION.COMPLETE
* REDUCTION\_CATALOG.STARTED
* REDUCTION\_CATALOG.COMPLETE

### Experiment Information

Actor: Experiment Repo

Action: <<send>>

The Experiment Information entity is experiment information from the Experiment Repo that the needs as part of executing a workflow. .

### Workflow Information

Actor: Experiment Repo

Action: <<receive>>,

The sends experiment workflow information to the Experiment Repo for persisting

### Workflow Description Information

Actor: Workflow Description Source

Action: <<send>>,

The Workflow Description Information entity is raw workflow descriptions from the Workflow Description Source that guides on how to execute a workflow.

### Heartbeat Msg

Actor: Monitor System

Action: <<receive>>,

The Heartbeat Msg entity is message that is received by the Monitor System from the as a mechanism to let the Monitor system know the is alive. It is not yet determined what if any other information is conveyed in the Heartbeat Msg

### User Reduction Cmnds

Actor: User (TBD)

Action: <<send>>,

The User Reduction Cmnds entity is a notion message sent by the User (TBD) actor that communicates a re-reduction of a set of experiment data. This entity has yet to be fully understood as the usage scenarios associated with re-reduction of experiment data has not yet been fleshed out.

# Interfaces & Protocols

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

<primary text>

## POSTPROCESS.DATA\_Ready

This message is generated by the Translation Service and sent to the AR Workflow System

This message can come in two formats: text string or JSON. Both formats contain the same basic information. This information is actually a path to a file on a file system. The path is not a UNC it does not contain a hostname of any kind. The assumption is that the file system is mounted to the both the system that generated the file and to the AR Workflow System. This means the path is immediately useable to get to the raw data file.

The following is the data that is contained in the path:

* Facility
  + SNS or
  + HIFR
* Instrument (ID)
* Experiment (ID)
* Run Number
* File Name

## Other Protocol Messages

Other messages contain the PATH a field for error and a field for info strings (not sure if this is one or more yet.)

* <bullet list: level 1>

<example bullet list>

* <bullet list: level 1>
  + bullet list: level 2>

# Use-Case View (High Level Functional Requirements)

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

This section captures high level functional requirements. One mechanism used for this system is Use Cases.

A picture containing food, rain

Description automatically generated

## Use Case: Setup Workflow System (NEEDS WORK)

*Technically this is a terrible use case, but it is an important behavior for white box realization.*

The use case begins when the Admin starts the . The performs all necessary activities to be ready to receive Post Process Data Msgs and execute workflows. Part of those activities includes collecting any relevant Experiment Information from the Experiment Repo.

Scenarios:

* Power On
  + <tbd>

## Use Case: Reduce Data Set (NEEDS WORK)

This use case begins when the Translation Service sends a Post Process Data Msg message to the that a new data set is ready to be reduced. On reception of the message the must determine the context of the data set from the meta data, determine which workflow is to be used to reduce the data set. The does not actually do any of the processing of the data set. Instead the sends messages to the Reducer systems (post processing systems) to do the actual reduction

Scenarios:

* Process 1 Data Set at a time
  + First Data Set for a new Experiment and new Run Set
  + First Data Set for known Experiment, but new Run Set
  + Additional Data Set for know Experiment and known Run Set
  + [For any of the steps above] Task (step) requires multiple incoming message to be successful/complete
* Process Multiple Data Sets for Same Experiment at the same time
  + Not that the system needs to handle this, but the Use Case, and hence the system is invoked for 1 Data Set at a time. The Translation Service’s request is for one Data Set
* Process Multiple Data Sets for Multiple Experiments at the Same Time
  + Not that the system needs to handle this, but the Use Case, and hence the system is invoked for 1 Data Set at a time. The Translation Service’s request is for one Data Set

## Use Case: Add (CRUD) Workflow Description (Procedure) (NEEDS WORK)

This use case begins when a USER (TBD) needs to add a new Workflow Description to the inventory of Workflow Descriptions. The must recognize this and add the new workflow description to its knowledge of workflow descriptions.

(It is not yet clear if the will be responsible for helping users create workflow descriptions or just reading them in from a source once they are created.)

Considerations:

* Is the Workflow System going to provide a GUI for this?
* Is the new Workflow Description contained in a file (JSON or other format)?
* Need to consider the Tasks (Steps) and Actions etc. and how new ones are created/added as well (again CRUD)
  + All aspects of a Workflow Description need to be considered
* Does a new step require new code, or can it really be something the system can excute without adding new code?

Scenarios:

* <bullet list: level 1>

## Use Case: Perform New Reduction on Existing Data Set (NEEDS WORK)

This use case begins when a <user> wants to perform a new reduction on a set of existing data that was previously reduced. The system will create new instances of the workflows involved but will use the latest version of the Workflow Descriptions that are indicated by the MetaData.

(It is unclear at this point how this will take place. This is a place holder for the required behavior without knowing the detailed interaction with an actor.)

Consideration:

* <bullet list: level 1>

Scenarios:

* <bullet list: level 1>

## Use Case: Rerun Existing Data Set (NEEDS WORK)

This use case begins when a <user> wants to rerun a run set that was run previously. The system will create new instances of the workflows involved but will use the exact same workflow descriptions that are specified in the existing Run Set and associated Runs.

(It is unclear at this point how this will take place. This is a place holder for the required behavior without knowing the detailed interaction with an actor.)

Considerations:

* How does the system allow the user to identify the Run Set to be rerun?
* The Runs associated with the Run set can possibly be reduced in parallel. The Run Set Workflow Description would govern this.

Scenarios:

* <bullet list: level 1>

## Other Relevant Information

Notional Use Case Survey:

* Reduce Run – The Translation Service sends a message (containing a DataSet) to the Workflow System to be processed
  + The system creates determines which Procedure (recipe) to use from the MetaData
  + The system instantiates a new workflow to bind the DataSet and the Procedure(s) together
    - This needs to happen at the RunSet and a Run level
    - Consider creating a factory to do this
  + There are two different types of Recipes for the RunSet
    - One to process and stitch each run as they come in to be processed
    - One to stitch after all runs have come in and been processed
* Rerun Existing Workflow – A “user” wants to rerun an old workflow again
  + This should use the same RunSet and Runs from the original
  + This will create a new instance of a workflow. (Not overwrite the prvious)
  + This implies that there is some mechanism for a user to identify an old workflow
  + This implies there is a mechanism to tell the Workflow system to rerun the identified workflow.
    - Need to determine if there are additional runs should they be included or not? Probably yes
* Run another reduction on existing data – A run set needs to be processed again
  + This generally occurs if the Procedure has changed.

Special Circumstances

* Goal is to be able to change a Procedure, but the not the MetaData of a data set that identifies which Procedure to use.
  + Thinking of a strategy that always uses the latest “version”
  + This implies there are a set of standard Procedures that only changes by version
  + Need to determine what to do if a DataSet needs to use a truly new Procedure. I would think the MeatData would have to change in this case

# System Architecture (Abstract Components)

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

<primary text>

A screen shot of a map

Description automatically generated

<example bullet list>

* <bullet list: level 1>
  + bullet list: level 2>

# Use-Case Realizations

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

<primary text>

## Use Case: Setup Workflow System (NEEDS WORK)

See Use Case description above

### Scenario 1: Prototype (Test) Startup Sequence

This is only for initial exploration of concepts

A screenshot of a cell phone

Description automatically generated

## Use Case: Reduce Data Set (NEEDS WORK)

See Use Case description above

### Scenario 1: First Data Set for New Experiment

<description>

A screenshot of a computer screen

Description automatically generated

## Use Case: Rerun Existing Run Set (NEEDS WORK)

See Use Case description above

### Scenario 1:

## Use Case: Perform New Reduction on Existing Data Set (NEEDS WORK)

See Use Case description above

### Scenario 1:

## Use Case: Add (CRUD) Workflow Description (NEEDS WORK)

See Use Case description above

### Scenario 1:

# Design

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

The system is comprised of the following major pieces (components/packages):

* **WorkflowSystem** – This component represents the entire workflow system. This is the black box. It contains all the other pieces (components) of the systems
* **WorkflowEngine** – The workflow engine is the part of the system responsible for receiving incoming messages and invoking the correct workflow to process the message. It is also responsible for sending any actions from a workflow that are to be executed by an actor (e.g. a Reducer).
* **WorkflowDescription** – The WorkflowDescription component is the set of Tasks that have been orchestrated to processes (recipes, procedures). There are multiple WorkflowDescriptions for each representing a different set of orchestrated tasks that can be used to process data from an experiment.
* **Workflow** – Workflows are created dynamically and bind a data set (or the reference to a data set) with a WorkflowDescription. It is this Workflow that holds the state of a data set that is waiting to be, is being, or has been processed. For the AR Workflow System there are two specializations of the Workflow; one for processing a group of data sets (runs) and another for processing the data sets (runs) themselves.
* **Experiment** – The experiment is all the information about an experiment. This includes all the groups of runs (run sets), all the information and data about each specific run and any associated state information. Each workflow will be included with the experiment. Note that this Experiment is be used and referenced by systems other than the AR Workflow System. It is likely that this system will only use part of the full set of experiment information, but that is out of scope for this system.
* **Actors** – The actors package is a container for components that can act as stubs for the real systems and users of the AR workflow system. These components are mostly used for testing.

## WorkflowSystem

<model diagram>

This component represents the entire workflow system. This is the black box. It contains all the other pieces (components) of the systems

## WorkflowEngine

<model diagram>

The workflow engine is the part of the system responsible for receiving incoming messages and invoking the correct workflow to process the message. It is also responsible for sending any actions from a workflow that are to be executed by an actor (e.g. a Reducer).

## WorkflowDescription

<model diagram>

The WorkflowDescription component is the set of Tasks that have been orchestrated to processes (recipes, procedures). There are multiple WorkflowDescriptions for each representing a different set of orchestrated tasks that can be used to process data from an experiment.

## Workflow

<model diagram>

The workflow component contains the set of classes and behavior to bind data sets (or the reference to a data set) with workflow descriptions. As workflow descriptions, along with their tasks, have no state, the workflow provides the state for the processing of the data sets.

The Workflow implements the Composite pattern. This allows the workflow engine to simply invoke a generic workflow.

### Workflow

Workflows are created dynamically and bind a data set (or the reference to a data set) with a WorkflowDescription. It is this Workflow that holds the state of a data set that is waiting to be, is being, or has been processed. For the AR Workflow System there are two specializations of the Workflow; one for processing a group of data sets (runs) and another for processing the data sets (runs) themselves.

The workflow maintains most of the state while the specializations handle the specialized workflow processing. Currently, workflows sequentially walk through a set of tasks from a workflow description. However, it is the Tasks that understand how to preform the processing. As the Tasks are stateless the workflow may pass itself or part of itself to the Task so the task has context in which to operate. (Details below.)

### GroupWF

The GroupWF is a workflow handles the workflow for a group of sequences (runs). The group workflow is responsible for ensuring all sequences that belong to the group are processed appropriately. The GroupWF contains other workflows, child workflows. The child workflows are sequence workflows (SeqWF) for processing each sequence that belongs to the group.

While the workflow engine will invoke the parent workflow it is expected that the WorkflowEngine will actually be invoking the group workflow and the group workflow will invoke the appropriate child workflow.

### SeqWF

The SeqWF specializes Workflow and handles the workflow for a sequence (run).

### TaskStatus

The TaskStatus is a structure to hold the state of a task. TaskStatus are expected to be collected into a table that represents the set of status for the Tasks that belong to the workflow description.

### WorkflowRepo

The WorkflowRepo represents a workflow repository. It is the responsibility of the WorkflowRepo to return the correct workflow based on information contained in the meta data of a data set.

If necessary a workflow is created if one does not already exist.

It is anticipated that this will/could be a front end class to some sort of persistent storage of workflows. The type of persistent storage to be used has not yet been considered so the design for how this class works in that context has not yet been done.

## Experiment

<model diagram>

The experiment is all the information about an experiment. This includes all the groups of runs (run sets), all the information and data about each specific run and any associated state information. Each workflow will be included with the experiment. Note that this Experiment is be used and referenced by systems other than the AR Workflow System. It is likely that this system will only use part of the full set of experiment information, but that is out of scope for this system.

# Design Patterns, Rules, & Guideline

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

<primary text>

## ID Strategies

There are a number of places where items in the system require an ID or a key. This section covers those cases and the strategy for creating the IDs and/or keys.

The following is a corollary document that contains some sample ID & keys used in testing: testData.xlsx.

Unless otherwise stated all of the following IDs and keys will be created, at least in part, from the values in the meta data associated with a data set. To that end a common strategy will be used where by sets of values will be concatenated and delimited by a forward slash “/”. This is consistent with the raw reference to a data set which is a location on a file system (presumably Linux hence the forward slash).

* Example:
  + instrumentID +
  + “/” + experimentID +
  + “/” + groupID
* ID/Key = “instrumentID/experimentID/groupID”

IDs are generally a single identifier for the specific piece of information such as a group or a workflow description.

Keys are most often used for searches or keys for storing the piece of information in something like a hashTable.

### Data IDs (Meta Data)

*NOTE: At this point I know these pieces of information are contained in the meta data, but I don’t know their real structure and format. All references to this data are considered abstract at this point till their real representation can be established.*

For the purposes of the prototype I have used the following convention for representing these pieces of data. There is a prefix that identifies the information type and a number (#) that identifies the ID of the information.

* Instrument ID (INST-<#>)
  + Ex: Instrument number 4: INST-4
* Experiment ID (EXP-<#>)
  + Ex: Experiment number 3054: EXP-3054
* Group ID (GRP-<#>)
  + Ex: Group number 0: GRP-0
  + Note this is the Sequence ID from the meta data
* Sequence Number (SEQ-<#>)
  + Ex: Sequence number 0: SEQ-0
* Data Type Number (DT-<#>)
  + Ex: Sequence number 0: DT-0

### Experiment IDs

The experiment ID is simply the experiment ID from the meta data.

* Experiment ID (EXP-<#>)
  + Ex: Experiment number 3054: EXP-3054

The fully qualified key would be:

* Instrument ID + Experiment ID
* Ex: INST-4/EXP-3054

### Group IDs

The group ID is simply the group ID from the meta data.

* Group ID (GRP-<#>)
  + Ex: Group number 0: GRP-0

The fully qualified key would be:

* Instrument ID + Experiment ID + Group ID
* Ex: INST-4/EXP-3054/GRP-0

### Sequence IDs

The sequence (or run) identifier is simply a number (integer) starting from zero (0). ID is simply the group ID from the meta data.

The fully qualified key would be:

* Instrument ID + Experiment ID + Group ID + Sequence Number
* Ex: INST-4/EXP-3054/GRP-0/SEQ-0

### Workflow Description IDs

Note: How to uniquely identify workflow descriptions is not yet fully figured out. Each workflow description, once used, must be retained for historical purposes. The issue is that it is yet not clear exactly how the meta data for a data set is used to identify the specific workflow description to be used. Here are some rules that have been discussed

Workflow Description for each experiment. It is expected that there will be a unique workflow for each experiment. The scientist conducting the experiment is likely to author the processing (reduction) of the data set.

Versions of Workflow Descriptions. It is likely there will be versions of the workflow descriptions for each experiment. This is because a scientist may not like the results they are currently getting and want to tweek the processing (reduction) algorithm in some small way. For instance, the formula used in a particular reduction step. Also, at some later time some scientist may want to run a group of sequences through a different workflow completely.

Default Workflow Descriptions. Currently the notion of a default workflow has been discussed where if there is a problem locating a specific workflow description for a data set, a default workflow description can/should be used be used. The discussed details are as follows. Look for a workflow using the fully qualified name of the sequence data type. If not found use the fully qualified name of the group. If not found, use the fully qualified name of the experiment.

There are actually two different types of workflows. One type for groups and one type for sequences. (See design above for details.)

#### Groups

At its core the Workflow Description ID for a group will be an identifying prefix and unique number. How to handle versions has yet to be determined.

* Group workflow description prefix: WDG-<#>
* The unique number is TBD, but current thinking is it will simply be some globally incrementing number.

The composition of the prefix is as follows:

* **WD** stands for workflow description
* **G** indicates group

Group keys are the fully qualified id of the group as follows:

* Ex: INST-4/EXP-3054/GRP-0

Group Completion Criteria are a bit different of a strategy. It is anticipated there are a limited number of group workflow descriptions – 2. These two completion criteria are as follows:

* **REDUCE&STITCH** – This is to indicate behavior to reduce the incoming data set and immediately stitch it together with the other data sets.
* **REDUCE&WAIT** – This is to indicate behavior to reduce all of the data sets before stitching them together.

#### Sequences

Each sequence does not get it own unique workflow description. The workflow description to be used on a set of sequences is expected to be derived from the data type of the data set. For this reason the workflow description ID for a group of sequences will include an identifying prefix and a unique number. How to handle versions has yet to be determined.

* Sequence workflow description prefix: WDT-<#>
* The unique number is TBD, but current thinking is it will simply be some globally incrementing number.
* Each experiment is likely to have

The composition of the prefix is as follows:

* **WD** stands for workflow description
* **T** indicates data type

Sequence keys are the fully qualified id of the group plus the data type as follows:

* Ex: INST-4/EXP-3054/GRP-0/DT-0

### Task IDs

Tasks are simple elements and don’t belong to anything until the are group in a workflow description. The identifiers reflect this with a simple prefix and a unique number. However, there are two different prefixes, one for tasks that belong to group workflow descriptions and tasks that belong to sequence workflow descriptions.

* Group Task Prefix: GTSK-<#>
* Sequence Task Prefix: TSK-<#>

### Workflow IDs

Each workflow contains a unique ID. The ID base will be created from the following pieces of data:

* Instrument ID (INST-<#>)
* Experiment ID (EXP-<#>)
* GroupID (GRP-<#>

A group workflow and a sequence workflow will vary by a prefix plus a number. The number represents which workflow it is in a possible set of workflows. The numbers start at zero (0) and increment as more workflows are added for the group or sequence. This makes the number equivalent to the index in an array as well. The prefix for each will be as follows:

* Group Workflow prefix: WFG-
* Sequence Workflow prefix: WFS-

Example:

* Group Workflow prefix: WFG-

## Architectural/Design Rules

Facades and Proxies

There a basic philosophy to follow. All objects should interact as if they are all in the same program on the same computer. This can be achieved by using interfaces, facades and proxies. The idea is that if an object need to interact with another object, the calling object calls an interface for the called object. The called object interface hides whether the called object is there or remote. All the communication to the called object is to be hidden behind the interface. This allows the core behavior of the program to be created and the complications brought on by scaling and distribution to be hidden. It also allows the those technologies to be more easily replaces over time without affecting the core code.

FLESH THIS OUT LATER - JXH

Stuff

# Thoughts, Questions, Notes

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

<primary text>

## Stateless workflow

I need to read more about stateless workflows. However, a thought is that the states of a workflow could be classes that are called. The “State” class could be specialized for any activity. Events may trigger behavior in the activitity or a transition to another “State”. Some thing to think about. Maybe another solution to this AR workflow system.

## Workflow State

Currently I have some level of workflow state going to a workflow Task. In this way the Task updates the state without having to persist the state itself. This allows for the Task to be used/reused multiple times without having to create a new instant of a Task that is stateful. However, I don’t do this with the WorkflowDescription. Should I? Currently the Workflow manages the activity and behavior. Why do this? Instead would it work for the WorkflowDescription to have the behavior to process the tasks? As with Tasks the workflow state could be passed into the WorkflowDescription.

Strategy Pattern

Investigate the Strategy Pattern as a way to handle different algorithms for processing data sets. Can the WorkflowDescriptions use this pattern as the algorithm for data reduction may change over time.

# To Do

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

This section contains a set of enhancements, improvements or just implementations to do at some point.

## Component Interfaces

The set of components that comprise the system have solidified. Interfaces for the different components have been identified as well. However, the interfaces have not been implemented as they came about later in the architecture. Formalizing the interfaces will expose only the essential methods that should external users (other components) should have access to. Most of the primary classes have other methods that are used for processing within the component and have no need to be exposed to the outside world

## Factories

There are several key places where it makes sense to apply one of the factory or builder patterns. (Which one is TBD.) In sort any parts of the system that will reside in a repository concept would benefit from using a factory pattern for creating the contents of that repository to be used by the systems. The following may or would benefit from having a factory to build them:

* Workflows – Hiding the creation of a workflow behind a factory will allow for different implementations of workflows and hide their complexity. Establishing a factory, if done correctly, will allow for implementations that allow for workflows to be created within the same process (program), or different processes (threads), or even have them created in containers in a cloud.
* Workflow Descriptions – Where the information for creating workflow descriptions is not yet determined. It may be from a flat file that a CSV or JSON file. Ulitmately it may be prudent to put the workflow descriptions in database. Creating a factory will allow the hiding of creating the workflow descriptions from the rest of the system. This will allow for flexibility in chainging how and where the raw workflow description information is being stored.
* Experiment – (NOTE: This is controversial at this time.) A workflow should be part of an experiment. The context of execution of a workflow is an experiment. The AR Workflow System is built with the premise that there is an experiment context. I believe this should be tied to a (the) current repository that holds experiment information. There are two conditions where a factory will be useful for the experiment in the AR Workflow system
  + Anytime a new data set for an existing experiment needs to be processed. A representation of that experiment needs to be created. Assuming the existing experiment is in an external store, the factory would know how to retrieve the relevant information and constitute a representation of it in the system
  + Any time a new data set is receive for processing and there is no existing experiment. This systems should create a representation of the experiment in the system and also in the external store.

## Multi-Process

The architecture of the system was done so that different parts could be separated into different processes. Ultimately the goal would be to all this system to scale by creating new processes on demand as the number of data sets being processed increases. This vision is to stand up containers in a cloud on demand that will handle some or all of the processing of data sets.

An initial step will be to create threads for processing of the workflows. From there perhaps spawning processes on the same computer. Then finally working with containers in a cloud. It is not clear how working with containers will change the architecture. I am hoping that can be hidden from the core behavior, and code, of the AR workflow system. This is where interfaces and factories will be essential.

# Appendix: Raw Model Diagrams

[This subsection is the start of the primary document content. Note that bullet and number lists should be of text style Normal. Normal is single spaced] It is recommended to place the captions at the top of tables and graphics.

<description>

## Workflow System Class Diagram

A screenshot of a cell phone screen with text

Description automatically generated

## Workflow Engine Class Diagram

A close up of a screen

Description automatically generated

## Workflow Class Diagram

A picture containing clock, hanging, mounted, black

Description automatically generated

## Workflow Description Class Diagram

A screen shot of a social media post

Description automatically generated

## Task Class Diagram

A picture containing clock, dark, screen, monitor

Description automatically generated

## Action Class Diagram

A picture containing clock, black, dark, mounted

Description automatically generated