

# Hadoop Enhancement

...

Team Space Jam

# Overview

- Explanation of Hadoop Enhancement
- Architectural changes caused by enhancement
- Components and interfaces that change
- Impacts of architectural change
- Approach to realize new feature
- Use case and sequence diagrams
- Limitations of reported findings
- Lessons Learned
- Conclusion

# Explanation of Problem

It can be very time consuming setting up a multi-node cluster.

- Java needs to be installed.
- PATH and JAVA\_HOME variables need to be set.
- User accounts need to be created.
- Nodes need to be mapped.
- Key based login needs to be configured.
- Hadoop needs to be installed on the slave servers and configured on the master server.
- A DataNode needs to be added to the cluster
- SSH access needs to be added to users
- etc.

# Explanation of Problem

- Most of these configurations are primarily done through bash commands and editing configuration files.
- It can be very confusing for a user to setup a multi-node cluster.
- Thus, this setup process can be shortened and made easier.

# Explanation of the Solution and Benefits

- Ease cluster setup by introducing a user interface to iterate through each setup phase.
- Ease cluster setup by saving instances of previous cluster histories.
- Prevents Setup Mistakes
- Reduces Setup Time
- Reduces User Confusion

# Approach #1: Hadoop Cluster Creation UI

- Create a user interface to simplify the process of cluster creation
- Allow users to simply “fill in the blanks”
- Avoids having to use command line
- Avoids having to edit certain files
- Streamlines process providing better understanding

# Approach #1: Hadoop Cluster Creation UI

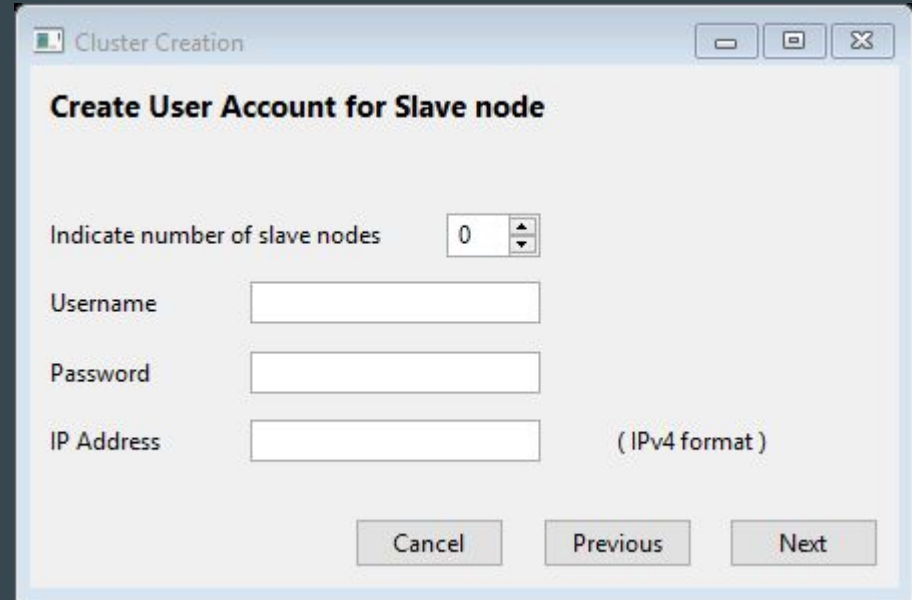
- First step create cluster name
- Create Master node user account
- Requires username, password, and IP address
- IP address used to automate process of setting up the server

The screenshot shows a 'Cluster Creation' window with the following fields and controls:

- Create Hadoop Cluster**
  - Cluster Name:
- Create User Account for master system**
  - Username:
  - Password:
  - IP Address:  (IPv4 format)
- Buttons: Cancel, Next

# Approach #1: Hadoop Cluster Creation UI

- Next frame sets up slave node accounts
- Requests for amount slave nodes being created
- Requests username, password, IP address



The screenshot shows a window titled "Cluster Creation" with standard Windows window controls (minimize, maximize, close). The main heading inside the window is "Create User Account for Slave node". Below this heading, there are four input fields:

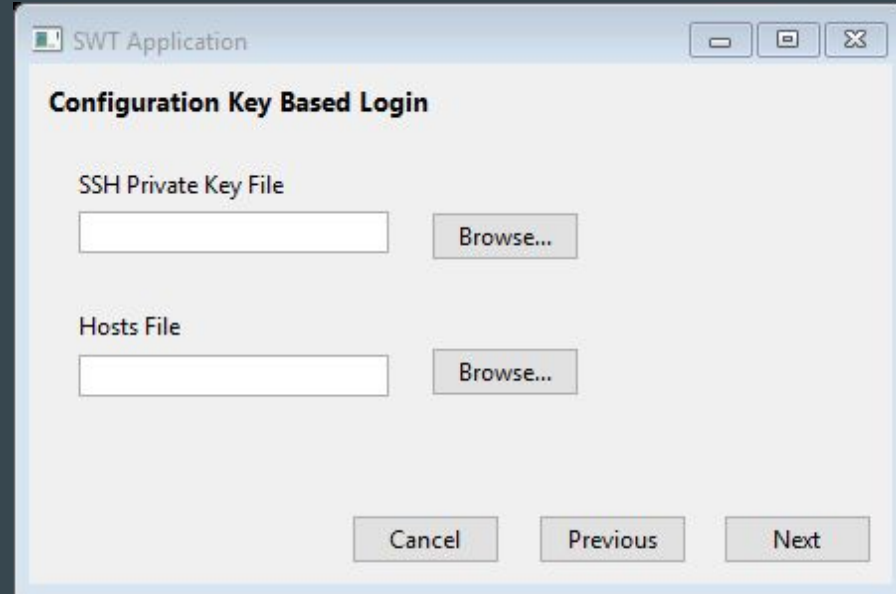
- "Indicate number of slave nodes" with a spinner box currently showing the value "0".
- "Username" with a text input field.
- "Password" with a text input field.
- "IP Address" with a text input field, followed by the text "( IPv4 format )".

At the bottom of the window, there are three buttons: "Cancel", "Previous", and "Next".



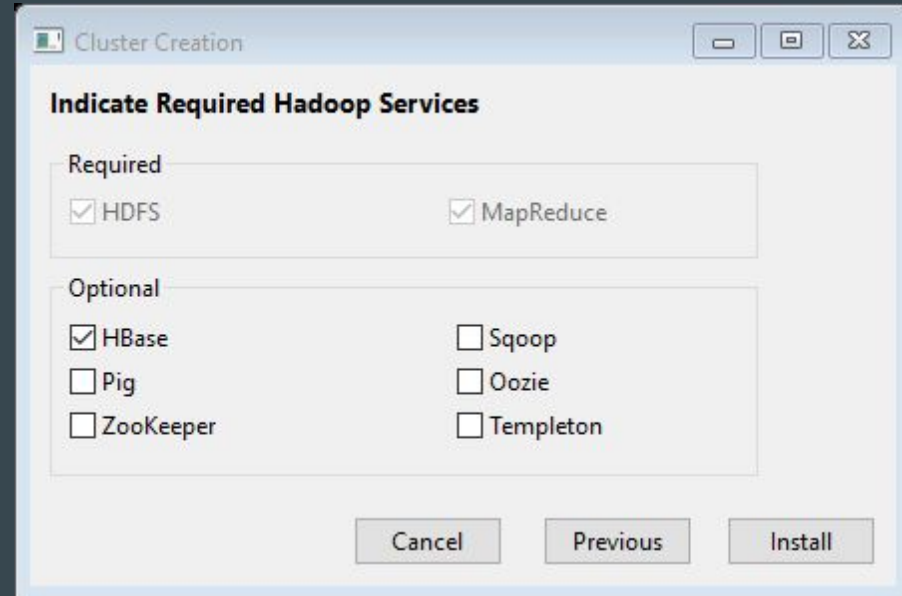
# Approach #1: Hadoop Cluster Creation UI

- Configuration Key Based Login
- Set up secure shell
- Allows nodes to communicate freely
- Private key used for root user
- Hosts file contains a list of all slave nodes
- Installs previous nodes



# Approach #1: Hadoop Cluster Creation UI

- HDFS and MapReduce are required and therefore must be checked
- Provides option of installing additional services
  - HBase - a non-relational database
  - Pig - Platforming to analyze large Data sets
  - ZooKeeper - Configuration management and Distribution Synchronization
  - Sqoop - Used to transfer bulk data
  - Oozie - Workflow system to manage jobs
  - Templeton - Webservice APIs for Apache Hadoop



# Approach #1: Hadoop Cluster Creation UI

## Advantages

- User friendly
- Streamlined process
- User will not have to edit files and use command line
- Available to a wider range of users

## Disadvantages

- May not be fully customizable for more experienced users
- May not contain certain available services to work with Hadoop
- Must be maintained

# Potential Software Architecture

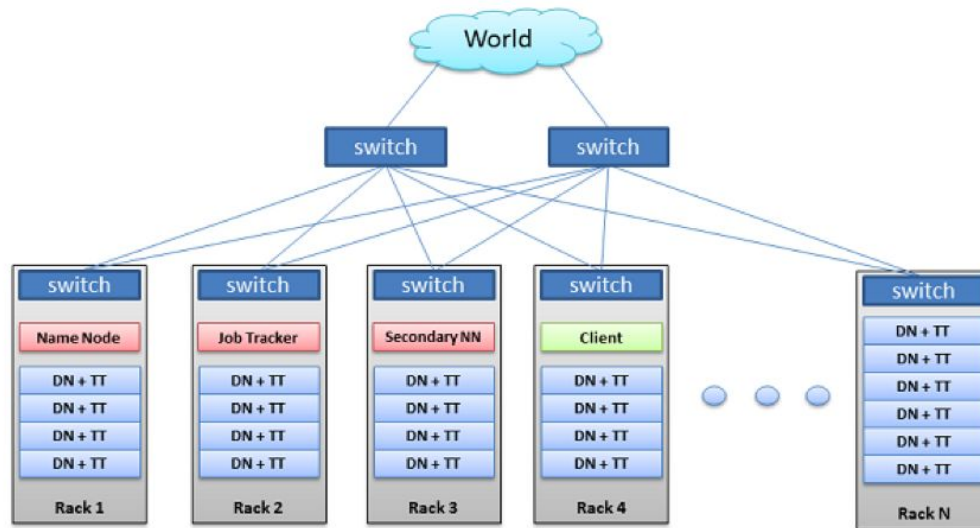
- Model View Controller : A software architecture, commonly used for interfaces, divides the application into three interconnected parts.
  - Model: manages the logic, rules and data.
  - View: Representation of information.
  - Controller: Takes inputs and converts them into commands for the view or model.



# Approach #2: Cluster History

- Many nodes deployed one at a time
- Client must specify each slave parameters individually through the command line interface
  - Linking the master, setting HostNames, adding DataNodes
- Can get tedious through multiple iterations

## Typical Hadoop Cluster Deployment



BRAD HEDLUND .com

**DN:** Data Node  
**TT :** Task Tracker  
**NN:** Name Node

# Approach #2: Cluster History

- A history of all the commonly used parameters of a slave system should be kept accessible to the master node
- Similar nodes can be created by copying the correct host login information, passwords, and server IPs to newly created data nodes so users don't have to manually type them in again
- Lots of content copying to slaves when a new node is created, this is a way to automate it

Execute the following on the master

```
mkdir -p $HOME/.ssh
chmod 700 $HOME/.ssh
ssh-keygen -t rsa -P '' -f $HOME/.ssh/id_rsa
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
chmod 644 $HOME/.ssh/authorized_keys
Copy the public key to new slave node in hadoop user $HOME directory
scp $HOME/.ssh/id_rsa.pub hadoop@192.168.1.103:/home/hadoop/
```

Execute the following on the slaves

Login to hadoop. If not, login to hadoop user.

```
su hadoop ssh -X hadoop@192.168.1.103
```

Copy the content of public key into file **"\$HOME/.ssh/authorized\_keys"** and then change the permission for the same by executing the following commands.

```
cd $HOME
mkdir -p $HOME/.ssh
chmod 700 $HOME/.ssh
cat id_rsa.pub >>$HOME/.ssh/authorized_keys
chmod 644 $HOME/.ssh/authorized_keys
```

Check ssh login from the master machine. Now check if you can ssh to the new node without a password from the master.

```
ssh hadoop@192.168.1.103 or hadoop@slave3
```

## Approach #2: Cluster History

- Linking new slave nodes to the master's HostName can be done automatically
- With these parameters stored within easy access, new slaves can be added with just a click of a single button!



```
ca. Command Prompt
cd $HOME
mkdir -p $HOME/.ssh
chmod 700 $HOME/.ssh
cat id_rsa.pub >>$HOME/.ssh/authorized_keys
chmod 644 $HOME/.ssh/authorized_keys

Would you like to link this new node to Master @ IP
1- 192.168.1.103
2- 192.168.1.99
3- Custom
>2

Would you like to use any of the data nodes found on
1- slave1
2- slave2
3- none
>3

slave3 machine
NETWORKING=yes
HOSTNAME=slave3.in
ssh -X Master@192.168.1.99
./bin/hadoop-daemon.sh start datanode
```

# Approach #2: Cluster History

## Advantages

- No longer setting up passwords and HostNames for each slave node
- Similar DataNodes can easily be copied over, modifications can be made to them as well
  - Create a DataNode that behaved like X, but does some stuff like Y

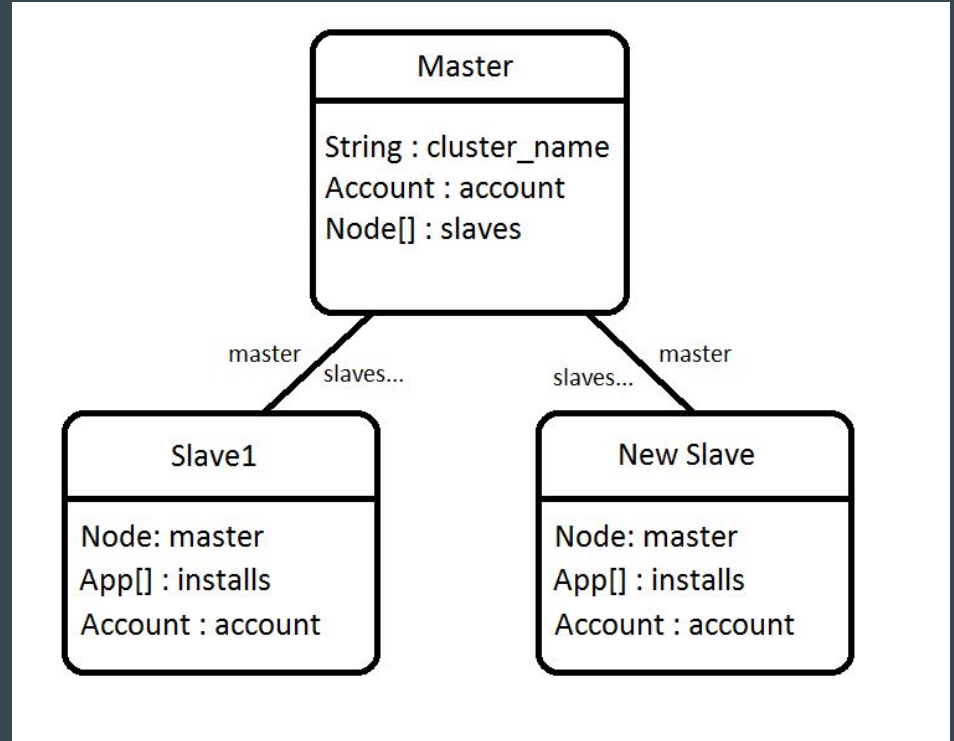
## Disadvantages

- Not all slave nodes are closely related
  - What if I wanted a new node that had nothing to do with the last few? Gotta type in those parameters by hand
- Privacy of input might be shared amongst other users operating on the same cluster



# Potential Software Architecture

- An addition of a new class to attach to the Master system to hold the name of currently created clusters, IP addresses, passwords, currently installed applications (on slave)
- Newly created slaves would be able to copy these parameters through a pointer from the Master node



# StakeHolders

Stakeholders	Non-functional requirements and priorities
Users	Ease of use, Performance, Reliability
Developers	Maintainability, Scalability, Bug Free
Customers	Reliability, Improves previous results

# Approach chosen is User Interface

- Due to its ease of use, Approach 1 has been selected to implement our new feature.
- User interfaces are much more user friendly for a beginner to use the system
- Reduces skill floor for first time users of Hadoop

# Architecture change



# Architecture Change

- The main architecture change is adding this new system which is dependent mainly upon HDFS
- This allows for the creation of master and slave nodes within HDFS that is represented visually by the user interface
- Subsystems will not be affected, this will just be an overlay on top of the system
- (Hey mina, maybe do one for both approaches, know what i mean, because the history solution will affect the architecture and subsystems) ps love you

# Limitations and Risks

- Removing or Deleting Cluster Services
  - Adding a service has been simplified by the enhanced features
  - How does one go about removing or deleting such services?
    - Identify the different components for the service
    - Identify the hosts where the components are running
    - Stop the service
      - Often fails to stop gracefully
      - Inability to locate hosts of components that fail to stop
      - GUI fails to manually query individual service components
    - Stop the components of service, if required
      - Unable to identify service components through GUI interface
    - Delete the service using DELETE API Call

# Limitations and Risks

- Limited ability to customize amongst the various Hadoop Distributions
  - Cloudera
  - MapR
  - Hortonworks
  - etc...

# Limitations and Risks

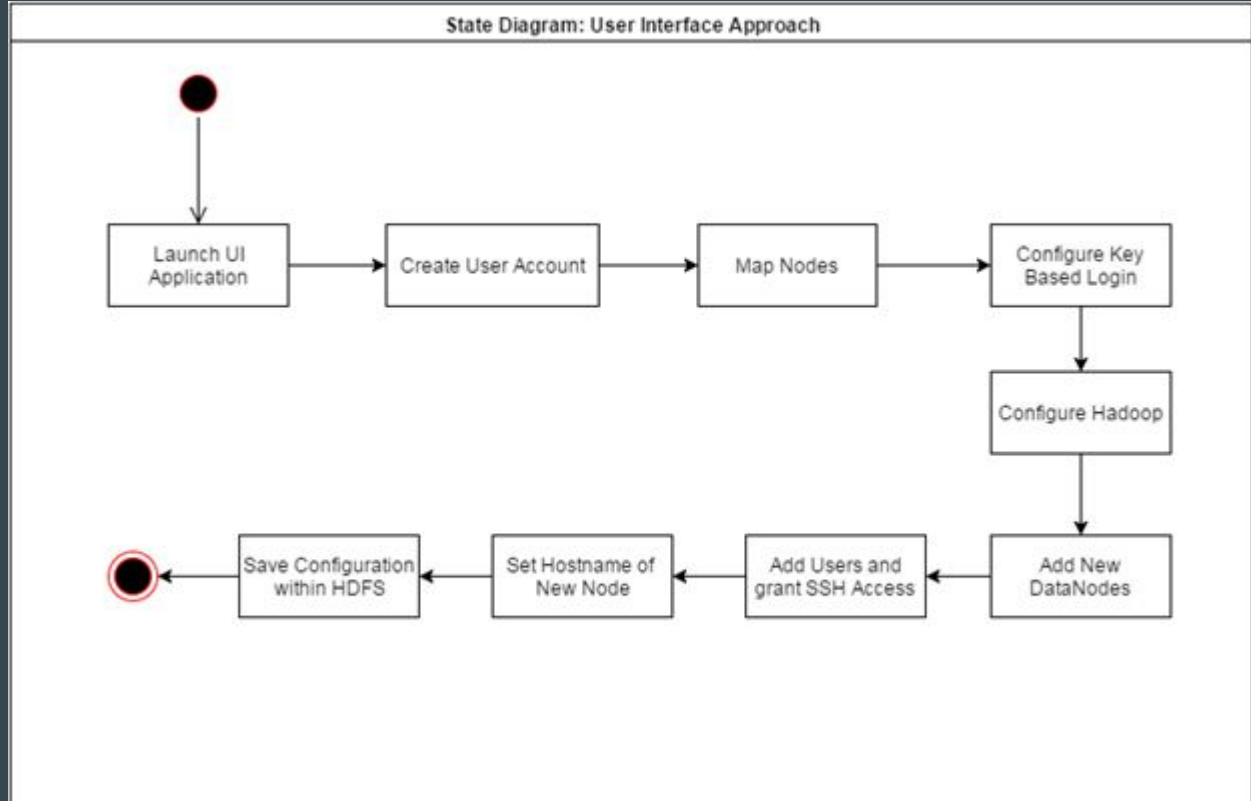
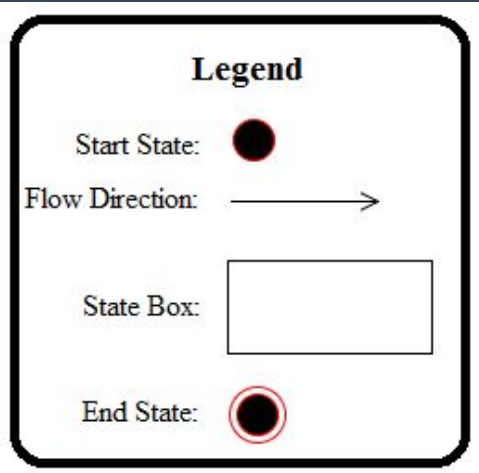
- Security
  - Even in a secure cluster, parts of the path are usually writeable, often leading to the unintentional creating of insecure paths
  - Permissions have to be explicitly asked for as there is no inheritance structure between nodes
  - Node permissions need to be locked down manually as to allow only authenticated users the ability to read and manipulate secret data written by specific services.



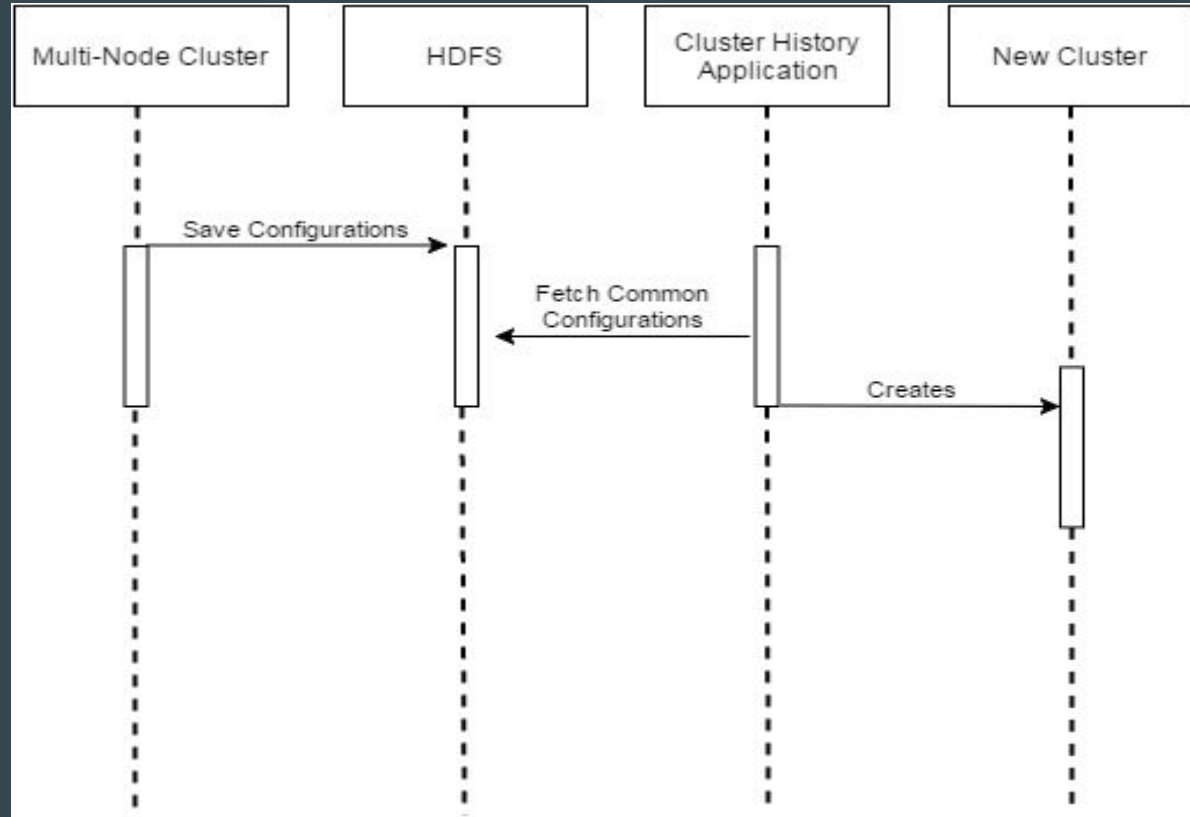
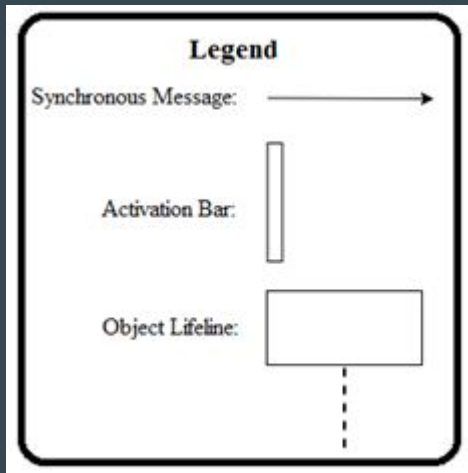
# Testing the Impact of Interactions

- Testing Security Concerns
  - MiniKDC
    - Integrated for Hadoop Security Unit Testing
  - The ability to run state analysis on Hadoop Clusters
  - Less Deterministic Test Results
  - Interpretation of Log Files is Difficult

# State Diagram: User Interface Approach



# Sequence Diagram: Cluster History



# The effects of concurrency and team issues

# Approach #1

- Through the intuitive user interface, the difficulty barrier can ease the task of delegating cluster creation.
- Possibility for redundancy, multiple users operating on the UI simultaneously could create a new cluster unaware of the other. Thus causing unnecessary bloat to the project, and resulting in a need to backtrack so that the issue may be fixed.
- Possible solution would be to lock new cluster creation while a user is in the process of doing so.

# Approach #2

- Multiple users cluster creation history will be available to others. This can potentially be harmful or helpful. Harmful, due to added clutter and potentially irrelevant histories for some users. Helpful due to increased interconnectedness between the team.
- Could be solved by attaching histories to specific users through their logins.
- History naming convention helps resolve team issues.

# Limitations of Reported Findings

- Difficult to find motivation for an enhancement when we're not part of the Apache development team.
- Most simple, yet effective, enhancements are already developed by hadoop or are in the process of being developed.
- If hadoop does not currently have this feature, most likely a third party company does.
- As a result, it is hard to create a feature or enhancement if one is not an advanced Hadoop user.
- Both approaches are not fully implemented, thus we can only vaguely describe their functionality and interaction with Hadoop.

# Lessons Learned

- Improving on an established product is more of a creative process than it is technical.
- Developing even a small feature is a demanding task.
- A more in depth knowledge of cluster creation within Hadoop.



# Conclusion

- Cluster creation is a tedious step within hadoop.
- Approach #1 attempts to solve this through a GUI.
- Approach #2 through the use of a history of previously created clusters.
- Approach #1 is adopted due to it's ease of use, and superior functionality.
- Possible impediments for this approach would be the failure to address existing security concerns and the inability to further simplify the removal process of cluster services.
- From this we learnt the difficulty of feature creation and that our brain childs are limited to predictions unless further developed.