“run the pipeline”

1. use pipeline tasks [how the developers do it]

CASA <20> hif_makeimlist(field=“GG Tau”,spw=“13,15”,specmode="cont”)

>> best to understand task parameters and experiment

“run the pipeline”

1. use pipeline tasks [how the developers do it]
2. use a script of pipeline tasks [how PIs do it]

CASA <20> execfile(“casa_pipescript.py”)

```
h_init()
try:
   hifa_importdata(dbservice=False, vis=['uid___A002_Xa2ce2e_X7bd'],
session=['session_1'])
   hifa_flagdata(pipelinemode="automatic")
   hifa_fluxcalflag(pipelinemode="automatic")
   hif_rawflagchans(pipelinemode="automatic")
   hif_refant(pipelinemode="automatic")
   hifa_tsyscal(pipelinemode="automatic")
   hifa_tsysflag(pipelinemode="automatic")
   hifa_antpos(hm_antpos='file')
 .....
```

“run the pipeline”

1. use pipeline tasks [how the developers do it]
2. use a script of pipeline tasks [how PIs do it]
3. run from PPR [supported by Kern et al for operations]

CASA <2> eppr.executeppr(“PPR…..xml”)

>> needs specific environment variables, and specific directory structure

“run the pipeline”

1. use pipeline tasks [how the developers do it]
2. use a script of pipeline tasks [how PIs do it]
3. run from PPR [supported by Kern et al for operations]

CASA <2> eppr.executeppr(“PPR…..xml”)

>> needs specific environment variables, and specific directory structure

linux> . /lustre…./pipeline_env_C4R1.sh
linux> pipelineMakeRequest uid://mous/uid intents_hifa.xml procedure_hifa.xml

>> creates
/lustre/naasc/sciops/comm/YOU/pipeline/root/2013.1.00576.S_2016_05_05T15_40_33.280/
>> creates PPR…xml

** procedure_hifa.xml is the “recipe” – easy to change e.g. procedure_hifa_inform.xml

“run the pipeline”

1. use pipeline tasks [how the developers do it]
2. use a script of pipeline tasks [how PIs do it]
3. run from PPR [supported by Kern et al for operations]
4. run wrapper script calibPipeIF.py [written by JAO to do “fixes” etc -> what JAO uses]
   calibPipeIF-NA.py [modified by me from JAO’s]

- sets environment
- runs pipelineMakeRequest
- executes PPR (just importdata)
- runs fixes (none right now)
- executes PPR (rest of processing)

[if using my informative imaging:
  - move calibration products aside
  - run informative imaging]

- cleanup, write fixes to casa_pipescript.py

>> read the output, learn to recognize _where_ its failing – in pipeline makeRequest, in first eppr.executeppr, in second eppr.executeppr, etc

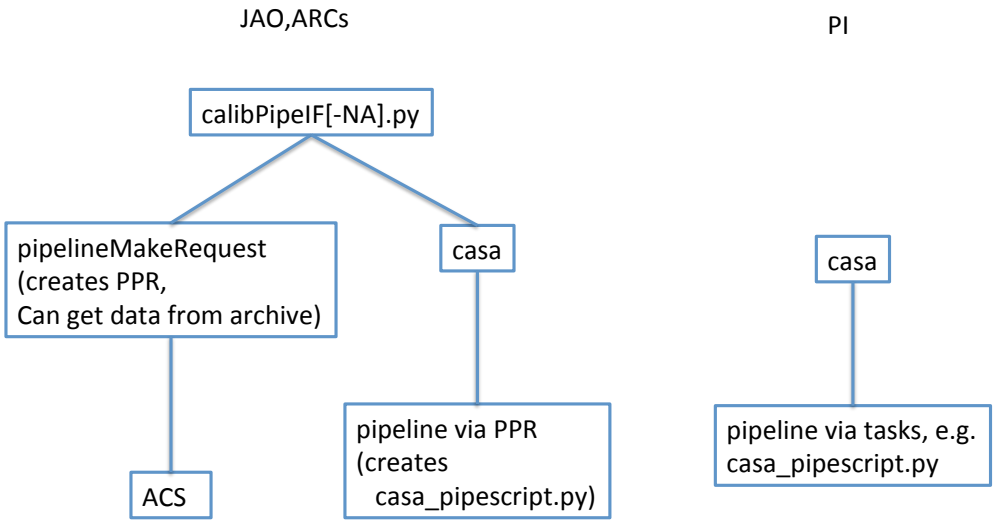
“run the pipeline”

1. use pipeline tasks [how the developers do it]
2. use a script of pipeline tasks [how PIs do it]
3. run from PPR [supported by Kern et al for operations]
4. run wrapper script calibPipeIF.py [written by JAO to do “fixes” etc -> what JAO uses]
   calibPipeIF-NA.py [modified by me from JAO’s]

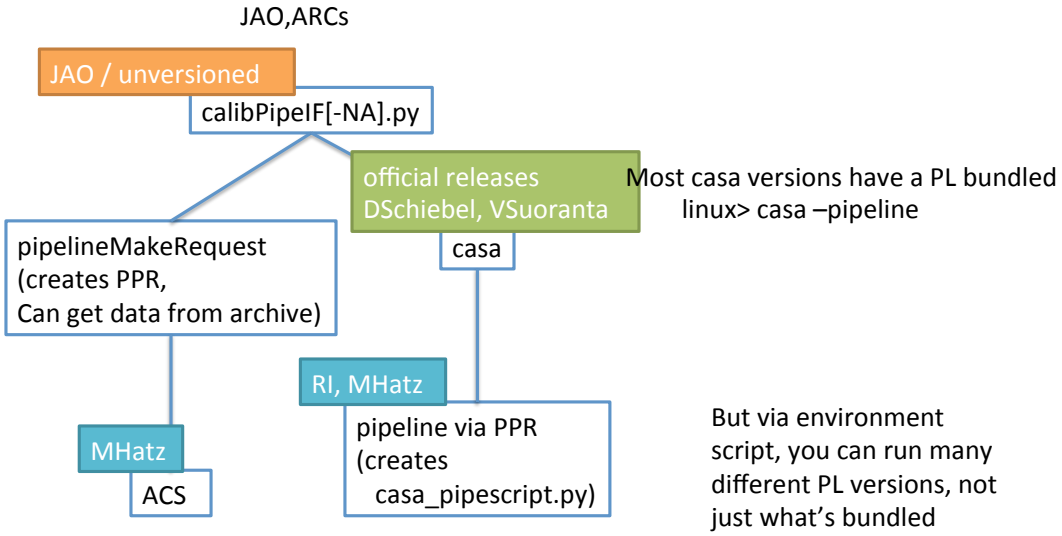
version used depends on what I have coded at the moment (C3R4)
--test invokes C4R1
--image runs imaging after calibration
--inform runs informative imaging (all but final cubes) after calibration

1. --mous=uid://MOUS/uid = “from scratch”
2. --flag=/dir/with/PPR/and/flagtemplates
3. --flag=/dir/with/just/flagtemplates –mous=uid://MOUS/uid > will make new PPR

“run the pipeline”



Versions



Versions

Most casa versions come bundled

```
linux> casa --pipeline

linux> casa -r 4.5.0-el6 –pipeline
CASA <2>: pipeline.revision
  Out[2]: 'r35156 (Pipeline-Cycle3-R3-B)'
```

Version Control

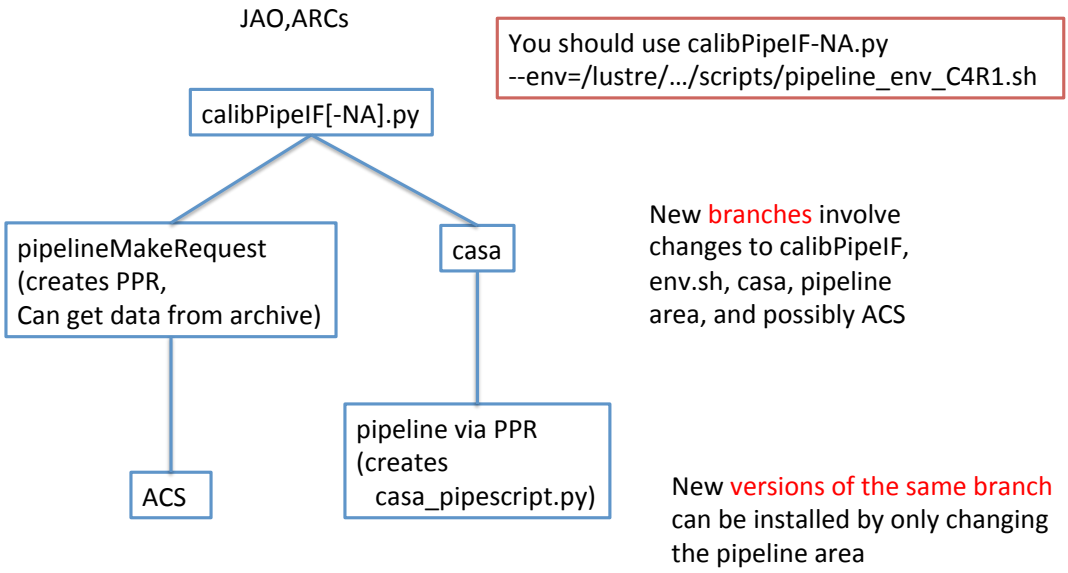
different pipeline versions can be used by setting the appropriate environment e.g.

```
linux> . /lustre/naasc/sciops/comm/rindebet/pipeline/scripts/pipeline_env_C4R1.sh

linux> casa

CASA <2>: pipeline.revision
  Out[2]: '36945M (Pipeline-Cycle4-R1-B)’
```

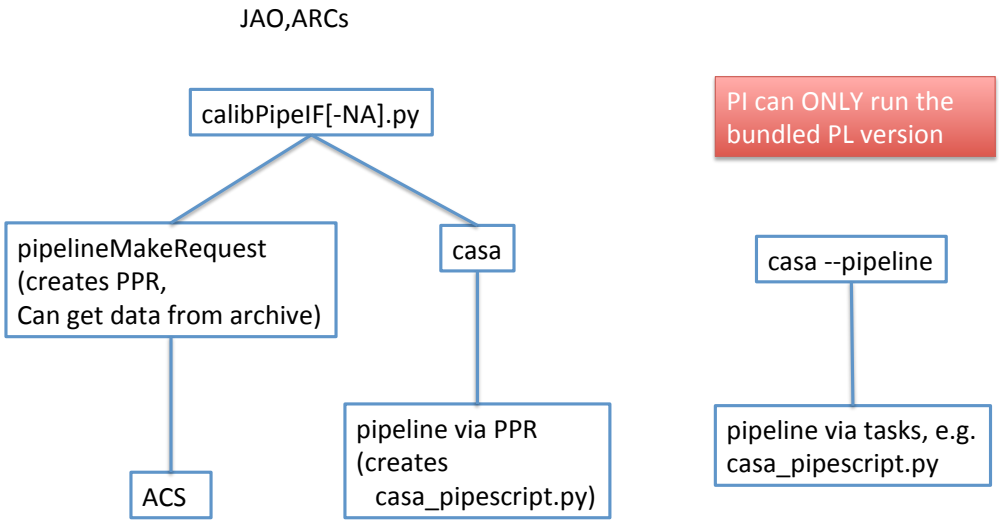
>> when making queries, specify not just the casa version, but the pipeline branch and revision as well.
>> info is in the weblog

https://wikis.alma.cl/bin/view/DSO/PipelineVersionTracker

Version Control



Versions



What do you get - weblog

Imaging [current workflow]

JAO runs calibration pipeline
JAO packages (keeps weblog, caltables, casa_restorescript.py)

DA reconstitutes (get ASDMs, run casa_restorescript.py)

[hopefully soon]
- DA runs informative, or full imaging using script method (just imaging tasks)
- assess imaging weblog

ARC does manual imaging

- casa_restorescript *should* be backwards-compatible (not extensively tested; and of course the actual processing that was done may be outdated)
- There is NO restriction on what is used for imaging, once you have a calibrated MS