DC3 SVN and C++ Namespace Reorganization

from DC3Management -> DC3StartupTasks.

References

The Data Management Work Breakdown Structure (WBS) guided the following layouts defined for both the SVN source repository and the C++ namespace. Refer to the following DOCUSHARE documents:

- Document-984 DM WBS Dictionary with element descriptions.
- Document-4687 DM WBS Responsiblities

DC2 transition to DC3

means svn '/trunk/' will be installed at that point in tree.

	DC2	DC3	Namespace	WBS	
	DC2	DMS			
Done	DC2/imageproc	DMS/ip/diffim#	lsst::ip::diffim	02.05.1.1.1	Difference Image Processing
	DC2/imageproc	DMS/ip/isr #	lsst::ip::isr	02.05.1.1.1	Image Subtraction Processing
Done	DC2/associate	DMS/associate#	lsst::ap	02.05.1.1.2	Association
Done	DC2/detection	DMS/detection#	lsst::detection	02.05.1.1.3	Detection
Done	DC2/movingobj	DMS/mops#	lsst::mops (nightmops)	02.05.1.1.4	Night Moving Object
	DC2/movingobj	DMS/mops#	lsst::mops (daymops)	02.05.1.1.4	Day Moving Object
		DMS/alert#	lsst::alert	02.05.1.1.5	Alert
		DMS/photocal#	lsst::photocal	02.05.1.2.1	Photometric Calibration
		DMS/astrocal#	lsst::astrocal	02.05.1.2.2	Astrometric Calibration
		DMS/coadd#	lsst::coadd	02.05.1.2.3	Image Coaddition
		DMS/classify#	lsst::classify	02.05.1.2.4	Image Classification
		DMS/deepdet#	lsst::deepdet	02.05.1.2.5	Deep Detection
		DMS/dqa#	lsst::dqa	02.05.1.2.6	Data Quality Assurance
Done	DC2/fw	DMS/afw#	lsst::afw	02.05.1.4	Application Framework
"	DC2/fw/image*	DMS/afw#/image	lsst::afw::image		
"	DC2/fw/WCS*	DMS/afw#/image/Wcs*	lsst::afw::image		
"	DC2/fw/formatters	DMS/afw#/formatters	lsst::afw::formatters		
"	DC2/fw/function*	DMS/afw#/math	lsst::afw::math		
"	DC2/fw/Kernel*	DMS/afw#/math	lsst::afw::math		
"	DC2/fw/DiaSource*	DMS/afw#/detection/	lsst::afw::detection		
Done	DC2/database	DMS/cat#	lsst::cat::srccat	02.05.1.5.1	Catalogs
		DMS/imagerep#	lsst::imagerep	02.05.1.5.2	Image Archive
		DMS/dui#	lsst::dui	02.05.1.5.3	Data User Interfaces
		DMS/opctrl#	lsst::opctrl	02.05.1.6	Operat'l Control and Monitoring
		DMS/calib#	lsst::calib	02.05.1.7	Calibration Products Pipeline
moved	DC2/dc2pipe	DMS/ctrl/dc2pipe#	lsst::ctrl::dc2pipe	02.05.2.1.1	Control and Management Service
		DMS/ctrl/orch#	lsst::ctrl::orch	02.05.2.1.1	Control and Management Service
Done	DC2/events	DMS/ctrl/events#	lsst::ctrl::events	02.05.2.1.1	Control and Management Service
		DMS/pcons#	lsst::pcons	02.05.2.1.2	Pipeline Construction Toolkit
		DMS/pex	lsst::pex	02.05.2.1.3	Pipeline Execution Services
Done	DC2/dps	DMS/pex/harness#	lsst::pex::harness	02.05.2.1.2	Pipeline Construction Toolkit
Done	DC2/mwi/exceptions	DMS/pex/exceptions#	lsst::pex::exceptions	02.05.2.1.3	Pipeline Execution Services
Done	DC2/mwi/trace	DMS/pex/logging#	lsst::pex::logging	02.05.2.1.3	Pipeline Execution Services

```
Done   DC2/mwi/logging  DMS/pex/logging#     lsst::pex::logging   02.05.2.1.3 Pipeline Execution Services
Done   DC2/mwi/policy   DMS/pex/policy#      lsst::pex::policy    02.05.2.1.3 Pipeline Execution Services
Done   DC2/mwi/data/Security  DMS/security   lsst::security       02.05.2.1.4 Security and Access Control

                       DMS/dbserv#          lsst::dbserv         02.05.2.2.1 Database Services
                       DMS/fsserv#          lsst::fsserv         02.05.2.2.2 Filesystem Services
                       DMS/qserv#           lsst::qserv          02.05.2.2.3 Query Services
                       DMS/catf#            lsst::catf           02.05.2.2.4 Catalog Construction Services
                       DMS/vo#              lsst::vo             02.05.2.2.5 Virtual Observatory Interfaces
                       DMS/daf              lsst::daf            02.05.2.2.6 Data Access Framework
Done    DC2/mwi/data    DMS/daf/data#        lsst::daf::data      02.05.2.2.6 Data Access Framework
Done    DC2/mwi/data    DMS/daf/base#        lsst::daf::base      Citizen, DataProperty, Persistable only
Done    DC2/mwi/persistence DMS/daf/persistence# lsst::daf::persistence   02.05.2.2.6 Data Access Framework
Done    DC2/mwi/utils   DMS/utils#           lsst::utils          (note: only the non-trace/logging routine he
Done    DC2/mwi/include/lsst/tr1  DMS/utils/include/lsst/tr1
Done    DC2/mwi/python/lsst/mwi/p_lsstSwig.i  DMS/utils/python/lsst/utils/p_lsstSwig.i

                       DMS/devenv#                               02.05.4.1  Development Environment and Tools
moved   DC2/deploy      DMS/devenv/build#
Done    DC2/editors     DMS/devenv/build#/editors
Done    DC2/BuildFiles  DMS/devenv/BuildFiles#
Done    DC2/SConsUtils  DMS/devenv/sconsutils#

Done    DC2/data/fwData DMS/testdata/afwdata#                     02.05.4.2.3 Test Data

Done    DC2/core        DMS/base#

Done    DC2/report      DMS/report/dc2report

Moved   DC2/analysis    DMS/analysis#
```

Miscellaneous DC2 Directories

```
Existing DC2 directories which need to be included or abandoned in DC3
-----------------------------------------------------------------------
DC2/community  -> DMS/community  (community outreach -- empty)
DC2/support    -> DC1/support   (3rd party support: PySextractor, PyWCSlib;   deadweight?)
DC2/das        -> DC1/das       (DC1 data access service: proto_ds; possibly deadweight)
DC2/dbingest   -> DC1/dbingest  (DC1 possibly deadweight)
DC2/consumer   -> DC1/consumer  (DC1 deadweight)
```

Include Transition from DC2 to DC3

```
.......... lsst::dps    -->  lsst::pex::dps
.......... 02.05.2.1.2  Pipeline Execution
dps/include/lsst/dps                --> pex/include/lsst/pex
dps/include/lsst/dps/InputQueue.h
dps/include/lsst/dps/Queue.h
dps/include/lsst/dps/Slice.h
dps/include/lsst/dps/Pipeline.h
dps/include/lsst/dps/Stage.h
dps/include/lsst/dps/Clipboard.h
dps/include/lsst/dps/OutputQueue.h

.......... lsst::events  -->  lsst::ctrl::events
.......... 02.05.2.1.1 Control and Management Services
events/include/lsst/events          --> ctrl/include/lsst/ctrl/events
events/include/lsst/events/EventSystem.h
events/include/lsst/events/Events.h
```

```
events/include/lsst/events/EventLog.h
events/include/lsst/events/EventTransmitter.h
events/include/lsst/events/EventReceiver.h
events/include/lsst/events/EventFormatter.h

.......... lsst::imageproc
.......... 02.05.1.1.1 Image Processing Pipeline
imageproc/include/lsst/imageproc
imageproc/include/lsst/imageproc/MaskPlanes.h
imageproc/include/lsst/imageproc/wcsMatch.h
imageproc/include/lsst/imageproc/ImageSubtract.h
imageproc/include/lsst/imageproc/PCA.h

.......... lsst::detection
.......... 02.05.1.1.2 Detection Processing Pipeline
detection/include/lsst/detection
detection/include/lsst/detection/Measure.h
detection/include/lsst/detection/Peak.h
detection/include/lsst/detection/BCircle.h
detection/include/lsst/detection/Measure.cc
detection/include/lsst/detection/Footprint.h

.......... lsst::ap
.......... 02.05.1.1.3 Association Pipeline
associate/include/lsst/ap
associate/include/lsst/ap/ZoneTypes.cc
associate/include/lsst/ap/Fifo.h
associate/include/lsst/ap/EllipseTypes.h
associate/include/lsst/ap/ChunkManager.h
associate/include/lsst/ap/Results.h
associate/include/lsst/ap/ChunkToNameMappings.h
associate/include/lsst/ap/Stages.h
associate/include/lsst/ap/ChunkManagerImpl.cc
associate/include/lsst/ap/Chunk.h
associate/include/lsst/ap/Chunk.cc
associate/include/lsst/ap/RectangularRegion.h
associate/include/lsst/ap/Match.h
associate/include/lsst/ap/DataTraits.h
associate/include/lsst/ap/Time.h
associate/include/lsst/ap/ZoneTypes.h
associate/include/lsst/ap/ChunkManagerImpl.h
associate/include/lsst/ap/CircularRegion.h
associate/include/lsst/ap/Bitset.h
associate/include/lsst/ap/Condition.h
associate/include/lsst/ap/SpatialUtil.h
associate/include/lsst/ap/CustomExceptions.h.m4
associate/include/lsst/ap/ScopeGuard.h
associate/include/lsst/ap/io
associate/include/lsst/ap/io/FileIo.h
associate/include/lsst/ap/io/ResultFormatters.h
associate/include/lsst/ap/Common.h
associate/include/lsst/ap/Mutex.h
associate/include/lsst/ap/Exceptions.h
associate/include/lsst/ap/Object.h
associate/include/lsst/ap/EllipseTypes.cc
associate/include/lsst/ap/Random.h
associate/include/lsst/ap/Utils.h

.......... lsst::mwi:persistence  --> lsst::daf::persistence
.......... 02.05.2.2.6 Data Access Framework
mwi/include/lsst/mwi/persistence        --> daf/include/lsst/daf/persistence/
mwi/include/lsst/mwi/persistence/DbStorageImpl.h
```

```
mwi/include/lsst/mwi/persistence/XmlStorage.h
mwi/include/lsst/mwi/persistence/LogicalLocation.h
mwi/include/lsst/mwi/persistence/Formatter.h
mwi/include/lsst/mwi/persistence/DbAuth.h
mwi/include/lsst/mwi/persistence/DateTime.h
mwi/include/lsst/mwi/persistence/Storage.h
mwi/include/lsst/mwi/persistence/FormatterRegistry.h
mwi/include/lsst/mwi/persistence/FitsStorage.h
mwi/include/lsst/mwi/persistence/FormatterImpl.h
mwi/include/lsst/mwi/persistence/StorageRegistry.h
mwi/include/lsst/mwi/persistence/Persistence.h
mwi/include/lsst/mwi/persistence/DbTsvStorage.h
mwi/include/lsst/mwi/persistence/DbStorageLocation.h
mwi/include/lsst/mwi/persistence/Persistable.h
mwi/include/lsst/mwi/persistence/BoostStorage.h
mwi/include/lsst/mwi/persistence/DbStorage.h

.......... lsst::mwi:data -> lsst::daf::data
.......... 02.05.2.2.6 Data Access Framework
mwi/include/lsst/mwi/data.h         --> daf/include/lsst/daf/
mwi/include/lsst/mwi/data           --> daf/include/lsst/daf/data/
mwi/include/lsst/mwi/data/SupportFactory.h
mwi/include/lsst/mwi/data/Provenance.h
mwi/include/lsst/mwi/data/LsstImpl_DC2.h  -> LsstImpl_DC3.h
mwi/include/lsst/mwi/data/FitsFormatter.h
mwi/include/lsst/mwi/data/LsstDataConfigurator.h
mwi/include/lsst/mwi/data/support.h
mwi/include/lsst/mwi/data/Citizen.h
mwi/include/lsst/mwi/data/LsstBase.h
mwi/include/lsst/mwi/data/ReleaseProcess.h
mwi/include/lsst/mwi/data/DataPropertyFormatter.h
mwi/include/lsst/mwi/data/DataProperty.h
mwi/include/lsst/mwi/data/LsstData.h

.......... lsst::mwi::data -> lsst::security
.......... 02.05.2.1.4 Security and Access Control
mwi/include/lsst/mwi/data/Security.h    --> security/include/lsst/security/

.......... lsst::mwi:policy  -> lsst::pex::policy
.......... 02.05.2.1.2  Pipeline Execution
mwi/include/lsst/mwi/policy                 --> pex/include/lsst/pex/policy
mwi/include/lsst/mwi/policy/PolicySource.h
mwi/include/lsst/mwi/policy/any.h
mwi/include/lsst/mwi/policy/PolicyFile.h
mwi/include/lsst/mwi/policy/PolicyParser.h
mwi/include/lsst/mwi/policy/xml
mwi/include/lsst/mwi/policy/Policy.h
mwi/include/lsst/mwi/policy/Dictionary.h
mwi/include/lsst/mwi/policy/PolicyParserFactory.h
mwi/include/lsst/mwi/policy/parserexceptions.h
mwi/include/lsst/mwi/policy/json
mwi/include/lsst/mwi/policy/json/JSONParser.h
mwi/include/lsst/mwi/policy/json/JSONParserFactory.h
mwi/include/lsst/mwi/policy/exceptions.h
mwi/include/lsst/mwi/policy/SupportedFormats.h

..........lsst::mwi:utils  -> lsst::utils
.......... 02.05.2.2.6 Data Access Framework
mwi/include/lsst/mwi/utils          --> utils/include/lsst/utils
mwi/include/lsst/mwi/utils/Demangle.h
mwi/include/lsst/mwi/utils/Utils.h
mwi/python/lsst/mwi/p_lsstSwig.i  --> utils/python/lsst/utils/p_lsstSwig.i
```

```
mwi/include/lsst/tr1/unordered_map.h  --> utils/include/lsst/tr1/

.......... lsst::mwi:exceptions  ->  lsst::pex::exceptions
.......... 02.05.2.1.3 Pipeline Execution Services
mwi/include/lsst/mwi/exceptions.h   --> pex/exceptions/include/lsst/pex/exceptions/
mwi/include/lsst/mwi/exceptions  --> pex/exceptions/include/lsst/pex/exceptions/
mwi/include/lsst/mwi/exceptions/Runtime.h.m4
mwi/include/lsst/mwi/exceptions/Exception.h

.......... lsst::mwi:logging  -> lsst::pex::logging
.......... 02.05.2.1.3 Pipeline Execution Services
mwi/include/lsst/mwi/logging        --> pex/logging/include/lsst/pex/logging/
mwi/include/lsst/mwi/logging/LogClient.h
.......... lsst::mwi:utils(trace)  -> lsst::pex::logging
mwi/include/lsst/mwi/utils/Trace.h   --> pex/logging/include/lsst/pex/trace/
mwi/include/lsst/mwi/utils/Component.h  --> pex/logging/include/lsst/pex/Component????
mwi/include/lsst/mwi/logging/LogRecord.h
mwi/include/lsst/mwi/logging/LogDestination.h
mwi/include/lsst/mwi/logging/ScreenLog.h
mwi/include/lsst/mwi/logging/LogFormatter.h
mwi/include/lsst/mwi/logging/Log.h
mwi/include/lsst/mwi/logging/DualLog.h

.......... lsst::fw --> lsst::afw
.......... 02.05.1.4  Application Framework
fw/include/lsst/fw/Framework.h      --> afw/include/lsst/afw/afw.h

.......... lsst::fw --> lsst::mops
.......... 02.05.1.1.4  Nightly Moving Object Pipeline
fw/include/lsst/fw/MovingObjectPrediction.h --> mops/include/lsst/mops/

.......... lsst::fw --> lsst::detection
.......... 02.05.1.2  Detection Pipeline
fw/include/lsst/fw/DiaSource.h      --> detection/include/lsst/detection/Source.h

.......... lsst::fw --> delete (should not have been in repository)
fw/include/lsst/fw/testmpa.cc
fw/include/lsst/fw/testmpa.h
fw/include/lsst/fw/vwimage.cc

.......... lsst::fw --> lsst::afw::image
.......... 02.05.1.4  Application Framework
fw/include/lsst/fw/fwExceptions.h.m4  --> afw/include/lsst/afw/ImageExceptions.h.m4
fw/include/lsst/fw/PixelAccessors.h      --> afw/include/lsst/afw/image/
fw/include/lsst/fw/MaskedImage.cc
fw/include/lsst/fw/MaskedImage.h
fw/include/lsst/fw/Image.cc
fw/include/lsst/fw/Image.h
fw/include/lsst/fw/Mask.h
fw/include/lsst/fw/Mask.cc
fw/include/lsst/fw/ImageUtils.h
fw/include/lsst/fw/Filter.h
fw/include/lsst/fw/Exposure.h
fw/include/lsst/fw/LSSTFitsResource.cc
fw/include/lsst/fw/LSSTFitsResource.h
fw/include/lsst/fw/DiskImageResourceFITS.h
fw/include/lsst/fw/WCS.h  -> afw/include/lsst/afw/image/Wcs.h

.......... lsst::fw  --> lsst::afw::formatters
.......... 02.05.2.2.6  Application Framework
fw/include/lsst/fw/formatters       -->afw/include/lsst/afw/formatters/
fw/include/lsst/fw/formatters/ImageFormatter.h
```

```
fw/include/lsst/fw/formatters/MaskFormatter.h
fw/include/lsst/fw/formatters/ExposureFormatter.h
fw/include/lsst/fw/formatters/MovingObjectPredictionFormatters.h
fw/include/lsst/fw/formatters/WcsFormatter.h
fw/include/lsst/fw/formatters/MaskedImageFormatter.h
fw/include/lsst/fw/formatters/Utils.h
fw/include/lsst/fw/formatters/DiaSourceFormatters.h _ afw/include/lsst/afw/formatters/SourceFormatters.h

.......... lsst::fw  -->  lsst::afw::math
.......... 02.05.1.4 Application Framework
fw/include/lsst/fw/Kernel                    --> afw/include/lsst/afw/math
fw/include/lsst/fw/Function.h       --> afw/include/lsst/afw/math/
fw/include/lsst/fw/FunctionLibrary.h
fw/include/lsst/fw/minimize.cc
fw/include/lsst/fw/minimize.h
fw/include/lsst/fw/Kernel.h
fw/include/lsst/fw/KernelFunctions.h

note: the following are automatically moved from Kernel to math when Kernel is renamed
fw/include/lsst/fw/Kernel/KernelFunctions.cc
fw/include/lsst/fw/Kernel/Kernel.cc
```

Comments Prior to Reconfiguration to DMS

Comments by KTL:

- Do we want to group DC3/dbserv, DC3/qserv, and DC3/catf, as they may in the end be provided by the same underlying software?
- ~~Should we still be prefixing everything with "DC3"? Doesn't it make more sense for DC3 to be a tag or version, eventually with its own branch, rather than a top-level structuring element?~~ (Done)
- ~~DC2/mwi/policy and DC2/mwi/utils (which is mostly Trace) should probably go under DC3/pexecute.~~ (Installed by RAA)
- ~~DC2/database is a set of schemas, not a set of database services. It corresponds to the catalogs in 02.05.1.5.1.*, but it was not broken down by catalog in DC2.~~ (Installed by RAA)
- ~~"DC2/mwi/*" should be "DC2/mwi/exceptions" as mapped to DC3/pexecute/exceptions.~~ (installed by RAA)
- ~~DC2/mwi should not be mapped to DC3/daf, as different parts go to pcontrol, pexecute, and daf.~~ (installed by RAA)
- ~~The whole DC2/mwi/policy -> DC3/daf/policy entry should not exist (this is in pexecute).~~ (installed by RAA)
- ~~DC2/mwi/tri -> DC3/daf/tr1 should really be DC2/mwi/tr1 (one, not i) -> DC3/devenv/tr1.~~ (installed by RAA)

Comments by JPK:

- ~~I would change pcontrol to control, as it will more than likely end up with more control services than just pipelines (e.g. Data transfers)~~ (installed by RAA)
- ~~We may wish to divide the pipelines (e.g DC3/Imageproc) between nightly and dataRelease (per the WBS), check with apps folks~~ (see Comments by RAA below)
- All of KT's comments are fine, except I am not sure dbserv and qserv will only be based on one set of software. We may enable some additional query services that go beyond SQL stuff.

Comments by RAA:

We might have a chicken and egg issue wrt dependencies:

- daf(persistence) depends on pex(policy) and pex(exceptions) depends on daf(data)
    - *KTL replies:*

      This may not be an issue as they are separate subcomponents of each package. On the other hand, it could be indicating that daf(persistence) and daf(data) really should be split up.

- Jeff and I talked about his comment on dividing the pipelines between nightly and dataRelease. He is OK with maintaining the layout as specified above. However, we need to decide where and how the policy files (.paf) for Telescope (mountain top), Nightly (base), and dataRelease (archive) will be located for development, build, and operation. I think they should be included in the appropriate pipeline module, for example (names to be decided):
    - imageproc/policies/nightly/*
    - imageproc/policies/archive/*
    - imageproc/policies/mountain/*

- ~~The current component known as DC2 should morph into an enduring software component stack (named DM); this component would retain its top level structuring characteristic. A final tag should be created for DC2 and thence for each subsequent data challenge stack. The DM structuring element serves to isolate LSST generated code from the other elements in the Root level of the SVN repository.~~ (Done)

Comments by SRP:

- ~~Agree about using tags for DC3 rather than prefixing with DC3. Generally in SVN trees this is how versioning is done.~~ (OK)
- Suggestion: When I've done this to source trees in other projects, I've found it to be beneficial to go in stages to be sure things still worked. Make the change to the existing structure, get that to build, and tag it before merging in new DC3 code/changes. That way we'll start DC3 from a known to be working base under the new structure, rather than trying to change the structure and merge changes all at once.

Comments by RayPlante:

I appreciate the attempt to align the packages with the WBS structure, but I feel there are areas where this is to the day-to-day disadvantage of the developer; my detailed comments below try to highlight these areas. I've never had the experience that the architectural organization set down at the beginning of a project survived completely intact; consequently, I expect that the divide between the WBS organization and what actually makes sense programmatically will grow with time. Rather than suffer from this, we should consider (1) adapting the UML organization over time, and/or (2) loosening the one-to-one correspondence between code packages and the WBSes.

Here are some detailed comments/questions:

- At the start of DC2, we said that we felt it was important to encourage the use of short-ish namespace component names. I think the change from `mwi` to `pexecute` goes against this notion. I like the more explanatory length of the third level (e.g. logging), but shorter names above that are nice.

♦ *Robyn replies:*

> Install your short-form names into the document.

- ~~The dps package is not conceptually part of Pipeline Construction; it is clearly part of Pipeline Execution. Pipeline Construction refers to components that automatically assemble and configure pipelines for particular purposes. In particular, smart components are envisioned for assembling pipelines on the fly that reconstruct previously calculated data products based on provenance. Currently, we have no software that falls in the WBS of Pipeline Construction.~~ (Done)

- One thing that is not clear to me is the level at which code is versioned and released as a package. Is it assumed that each path under the DC3 column contains a "trunk" subdirectory?
  ♦ *KTL replies:*

  > For DC2, this was always at the top level below DC2. Presumably in the future this would always be below DM[S]. That's one reason to go to a finer-grained split at this level.

  ♦ *Robyn replies:*

  > Yes. Modules at this level were felt to be an integrated entity. See another reply lower.

- If the answer to the previous question is "yes", does it really make sense to have a separate package for each kind of catalog? Do we envision any commonly shared bits? (Would that be part of "DC3/catf"?)

- Are we specifically suggesting we split mwi and fw each into smaller packages? This will certainly give us more flexibility in versioning their capabilities. Are there overhead issues? Will it complicate our dependency management?
  ♦ *Robyn replies:*

  > MWI has been parcelled out to 4 or 5 major components; fw remains intact. Further division should be discussed. We need to be concerned with potential for dependency circular loop.

  ♦ *Ray replies:*

  > Based on the replies from KT and Robyn, I'm still confused as to where I expect to find the trunk directories for the separately versioned packages.

- I think the existence of our mwi package suggests that we need another WBS entry that is analogous to 02.05.1.4 (Application Framework) but called "Middleware Interface". This is consistent with the recent work on the project organization.

- ~~I believe we proposed at our DC3 post mortem to merge trace and logging.~~ (Done)

- ~~I recommend that we do *not* create package directories until we actually have things to put in them. They only clog/confuse our view and maintenance of the software stack, and our notion of the best organization my change over time.~~ (Second that)

I would like to float the idea of grouping our packages together more without necessarily a corresponding change in the namespace. For example, under the root software directory we would have:

```
apps                applications (WBS 02.05.1)
  apps/imageproc
  apps/associate
  ...
  apps/dqa
  apps/afw
  apps/db              (short for database)
mw                  middleware (WBS 02.05.2)
  mw/pipeexec          packages related to pipeline execution
     mw/pipeexec/pfw   (formerly dps)
  mw/pipeconst
  mw/control
  mw/daf
     mw/daf/data
     mw/daf/persistence
  mw/mwi               packages exposing middleware capabilities to the pipeline implementations
     mw/mwi/policy
     mw/mwi/logging
     mw/mwi/utils
hwi                 infrastructure (pnemonic: hardware infrastructure; WBS 02.05.3)
devenv              development environment (WBS 02.05.4.1)
     devenv/build
     devenv/sconsUtils
```

The idea is to reduce the number of items in a directory, thereby giving a less cluttered, functionally organized view of the software stack. (We may want to group apps packages more, but I leave that to the preference of the app developers.) The namespaces, however, can remain much as they are now. One exception is that I would consider inserting a "`db`" field in each of the catalog packages.

- *KTL replies:*

  I think it's useful to have the namespaces (both C++ and Python) reflect the directory hierarchy. Having an additional grouping level above the current package level, while perhaps nice organizationally, would make it more difficult to navigate and to find source files.
- *Ray replies:*

  That is, if I'm interpreting your comment correctly, if you are staring at a piece of code that is using classes from some other namespace, you know exactly where the code for those classes are located. If we insert other directories into the directory structure that are not reflected in the namespace, then the location becomes a bit more mysterious.

  - *Robyn comments:*

    It all comes down to a choice of where the SVN package boundaries are placed and hence, how easily it is for a developer to extract and work on a particular SVN package. Early on we found that having the entire software stack in a single CVS module caused problems with coordination between developers. The division of packages for DC2 was generally OK except for FW and MWI --which bundled too many disparate entities within a single SVN package. The visual layout of the SVN repository should not drive our organization model.
  - *Robyn notes:*

Remember changes to the C++ namespace need to maintain consistency with the DCx model being implemented. During reverse engineering, we create an EA package for each namespace grouping; the reverse engineering process is much simpler if the EA packages align with the C++ namespace.

~~Finally, on the subject of the root directory, I agree that we should make our working branch generic - I would recommend "DMS". When a data challenge is complete, we make a copy of DMS and call it DC*n*.~~ (Done)

Comments by Russell Owen:

I would like the fw layout to look more like vw (because it seems logical and we might as well emulate it). Thus:

- ~~Use namespace math instead of function (I think function is too narrow). Then it obviously includes the minimizer.~~ (Done)
- Add a core namespace for things such as the header file that includes the types we are explicitly instantiating and defines our convention for the position of the center of a pixel.
    - *Robyn comments:*

      Tell me the explicit headers to which you are referring.
    - *Russell replies:* There is no such header at present but we need it. Some of the data is in other headers and some does not yet exist. Contents will include typedefs for the supported types of mask (presently in Mask.h), image (does not exist) and kernel (does not exist) and a constant for our definition of the position of the center of the 0,0 pixel (presently in ImageUtils?.h).
    - *Russell adds:* another possible place for this header is afw/trunk/includes/Image.h (the header file that brings in *all* the contents of afw/trunk/includes/image, not the header file that defines the Image class, which unfortunately will probably also be named Image.h but will be one level down). Then we would not need any extra "core" directory. I lean towards this.

- ~~Perhaps ditch namesapce kernel and put kernels under image. I don't have strong feelings about this, but in mild support: kernels are basically just images and it reduces ambiguity as to where the convolve and apply functions should go (they could also go under math, but in vw they are in image).~~ (Done)

Also, should the ds9 interface continue to go into display? (Well it is currently Display but now is the time to switch to lowercase.) It seems a bit specialized but I've not thought of anything better.

Comments by TSA:

- ~~I don't think it's useful to split up DC2/database. What's in there are the schema definitions as .sql files,~~

and a few utility scripts. There are no namespaces to split up. (Done)

- ~~Shouldn't there be a module for the proposed middleware orchestration layer? Have I missed it?~~ (placed in DMS/ctrl/orch)

- ~~I think kernel should go into math with function, not into image~~ (Done)
    - *Russell replies:* why this departure from VW's layout? I guess it's no big deal but my inclination is to either use image/ (like vw) or put Kernel into a kernel subdirectory. The latter is probably justified by the complexity of the kernel classes.

Comments by RHL

- ~~I agree with Robyn's comments on nightly and dataRelease pipelines — the code is basically the same so a split would be counterproductive. Her proposed Policies subdirectories seems fine to me (but shouldn't it be policies?)~~ (Done)

- Russell asks, "should the ds9 interface continue to go into display?". I'd say no, that's basically history. We might want to rename it to `display` at the same time (`import lsst.fw.display as display; display.zoom(4)`). I assume that `Catalog` will be simply deleted.
    - *Russell comments:* Just to be sure I understand: you are proposing that there be a package named lsst.fw.display and it will contain the various classes and subroutines that permit sending images to ds9. If so this means renaming python/.../Display to python/.../display and then making the init.py file import all the "good stuff", right?

- ~~The database namespace seems a little shallow; I'd prefer~~ ~~`lsst::catalog::srcca`~~ ~~(or something) to~~ ~~`lsst::srcca;`~~ (Fixed editing error)

- in fact, maybe all the ex-mwi namespaces should be moved down a level into a small number of namespaces (e.g. do we really want `lsst::dbserv` and `lsst::fsserv`). Why don't I say the same about e.g. `lsst::imageproc`? I think that it's because a developer will probably only be working on e.g. imageproc, whereas I expect a (different) developer to be working more-or-less at once on all of the services. Maybe it's too early to be making these distinctions, but I'd rather over-divide than vice-versa.

- KT writes, "If we insert other directories into the directory structure that are not reflected in the namespace, then the location becomes a bit more mysterious." I agree (Pan-STARRS did this and it confuses me to this day). You can always build tags files, but why make things hard? This implies that if the previous notes on namespaces are adopted we should change the svn layout too.

- ~~Can we get rid of directories called core? It scares me that there'll be a cron script out there that'll delete them, and at the very least it's confusing.~~ (Renamed it 'base'; can be changed to something else)
    - *Russell replies:* could you expand on this? What "core" directories are deleted by cron jobs? (I am curious because I proposed "core" for afw as a place to put the header file that contained the chosen types for Image, etc. and it never occurred to me that directories would be at risk. In any case I'm leaning towards not having that core directory anymore.)

Comments by SRP

- all references to "DMS/ctrl/orch" should be "DMS/ctrl/orca"