www.csiro.au

The ASKAP Telescope Operating System (TOS) – Lessons Learned

Juan Carlos Guzman (on behalf of the ASKAP Computing Team)
ASKAP TOS Development Group Leader

SKA M&C CoDR, Pune, India, 8 – 10 Nov 2011

CSIRO

Outline

1. ASKAP Specifications
2. Developing the ASKAP TOS
3. ASKAP TOS Status
4. Lessons Learned

What is ASKAP?

ASKAP - CASS

CSIRO

Astronomy and Space Science (formerly ATNF)

- **The largest single astronomical institution in Australia (~300 staff)**
- **Sydney HQ + Parkes (the 'Dish'), Narrabri (Compact Array), and Mopra, near Coonabarabran.**
- **Operates NASA CDSCC (Tidbinbilla)**
- **CASS is designing and building the Australian SKA Pathfinder ASKAP**

What is ASKAP?

Australian

Square

Kilometre

Array

Pathfinder

Location

Science

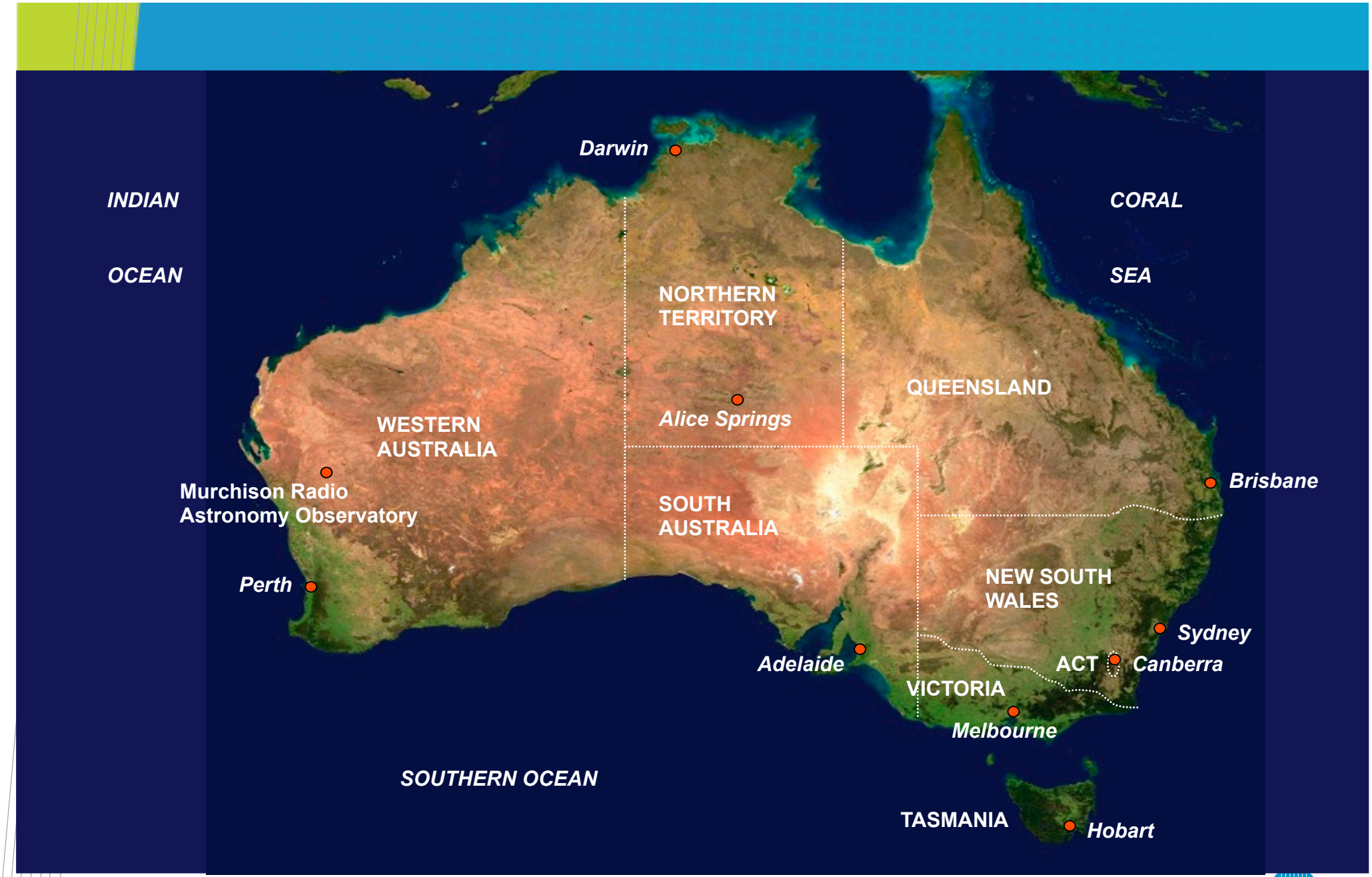
Technology

CSIRO

ASKAP Design Specifications

• Number of antennas	36 (630 baselines)
• Antenna diameter	12 m
• Collecting Area	4072 m^2
• Maximum baseline	6 km
• Cont. Angular resolution	10 arcsec
• Sensitivity	65 m^2/K
• Frequency range	700 – 1800 MHz
• Focal plane phased array	188 elements (92 dual pol)
• Field of view	30 deg^2
• Processed bandwidth	300 MHz
• Number of channels	16,384
• Correlator integration time	5 sec

CSIRO

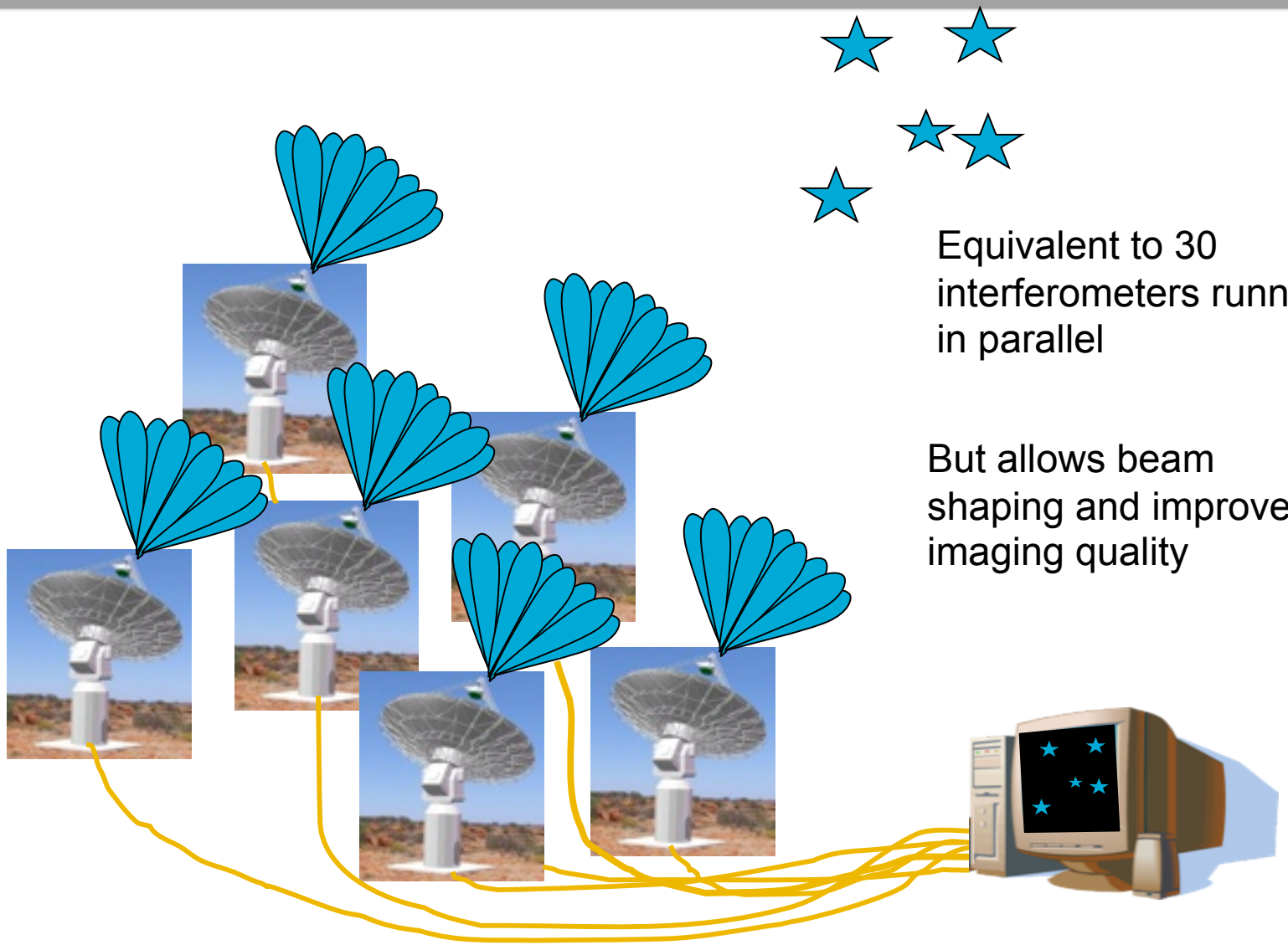

Darwin
INDIAN
OCEAN
CORAL
SEA
NORTHERN
TERRITORY
QUEENSLAND
Alice Springs
WESTERN
AUSTRALIA
Murchison Radio
Astronomy Observatory
Brisbane
SOUTH
AUSTRALIA
Perth
NEW SOUTH
WALES
Sydney
Adelaide
ACT
Canberra
VICTORIA
Melbourne
SOUTHERN OCEAN
TASMANIA
Hobart

CSIRO

Lake Macleod
Carnarvon
WESTERN AUSTRALIA
Monkey Mia
Murchison
Boolardy
ASKAP
350 km
Geraldton

CSIRO

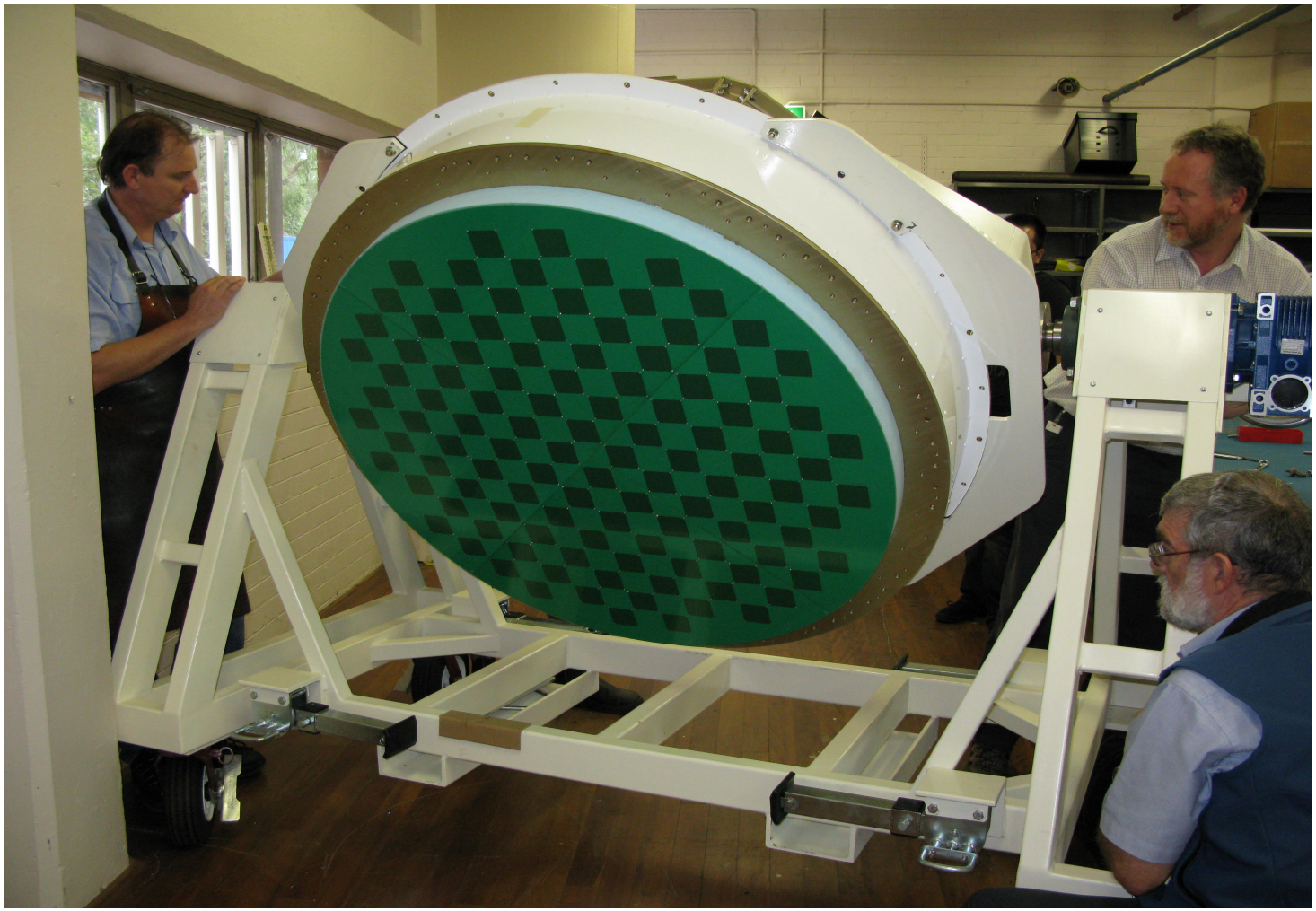
ASKAP Phased Array Feed (PAF) Technology



Equivalent to 30 interferometers running in parallel

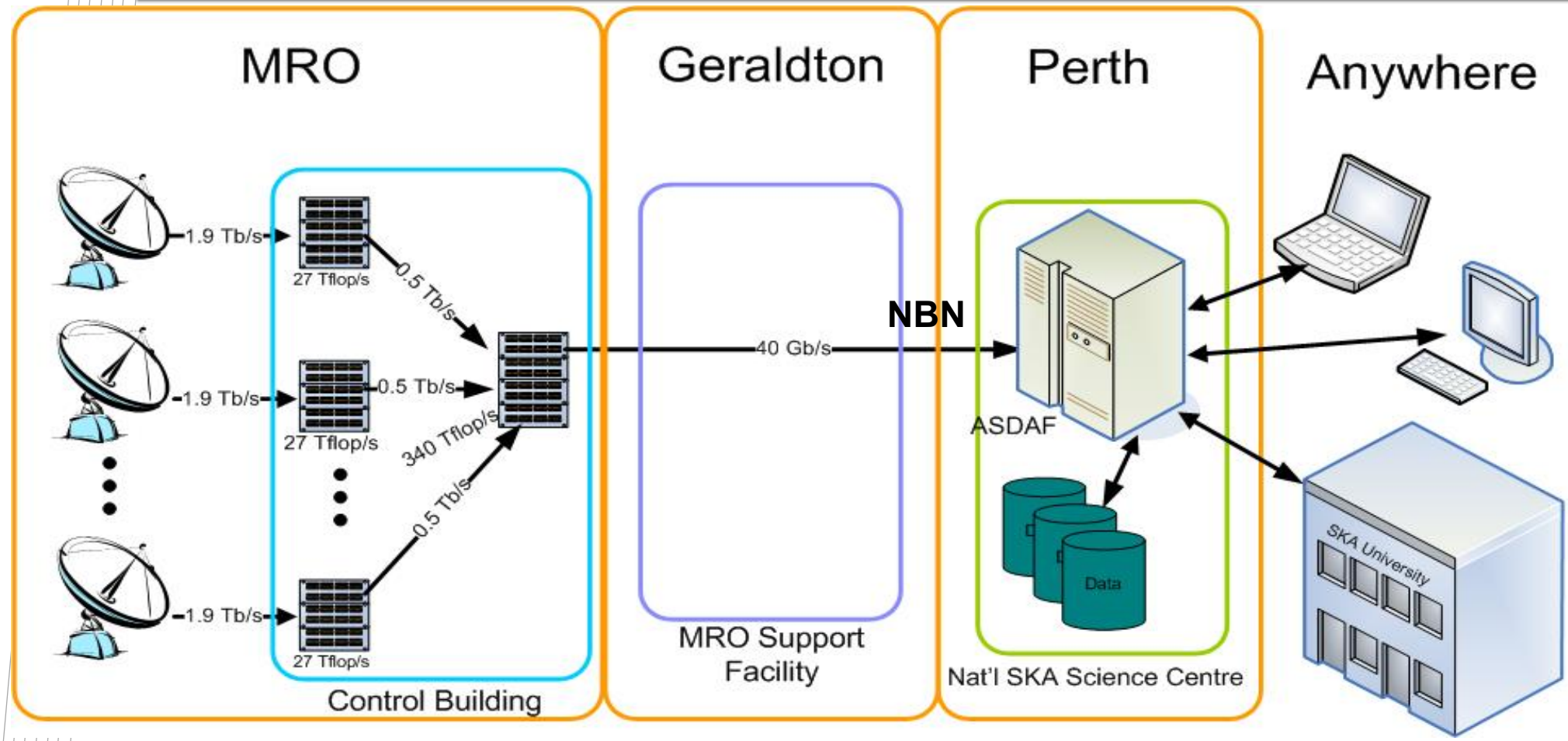
But allows beam shaping and improved imaging quality

ASKAP PAF R&D



CSIRO

ASKAP System and Data Flow



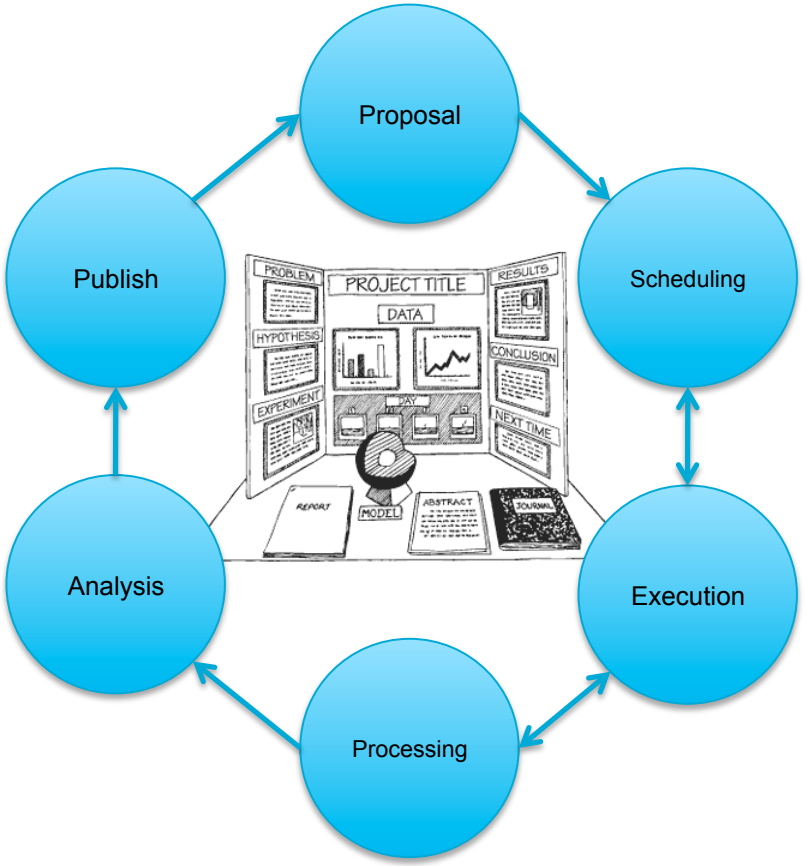
CSIRO

Developing the ASKAP Telescope Operating System

CSIRO

Key Project Requirements

- End-to-End Data Flow System (observation life cycle “wheel”)
  - Trend in new large astronomical facilities
- Survey instrument
  - But targeted observations also need to be supported
- Queue observing mode (semi-automated)
- Remote operations
- Commensal observing
- Instrument capability delivered incrementally
- Support degraded operations
- Support sub-arrays (desirable)

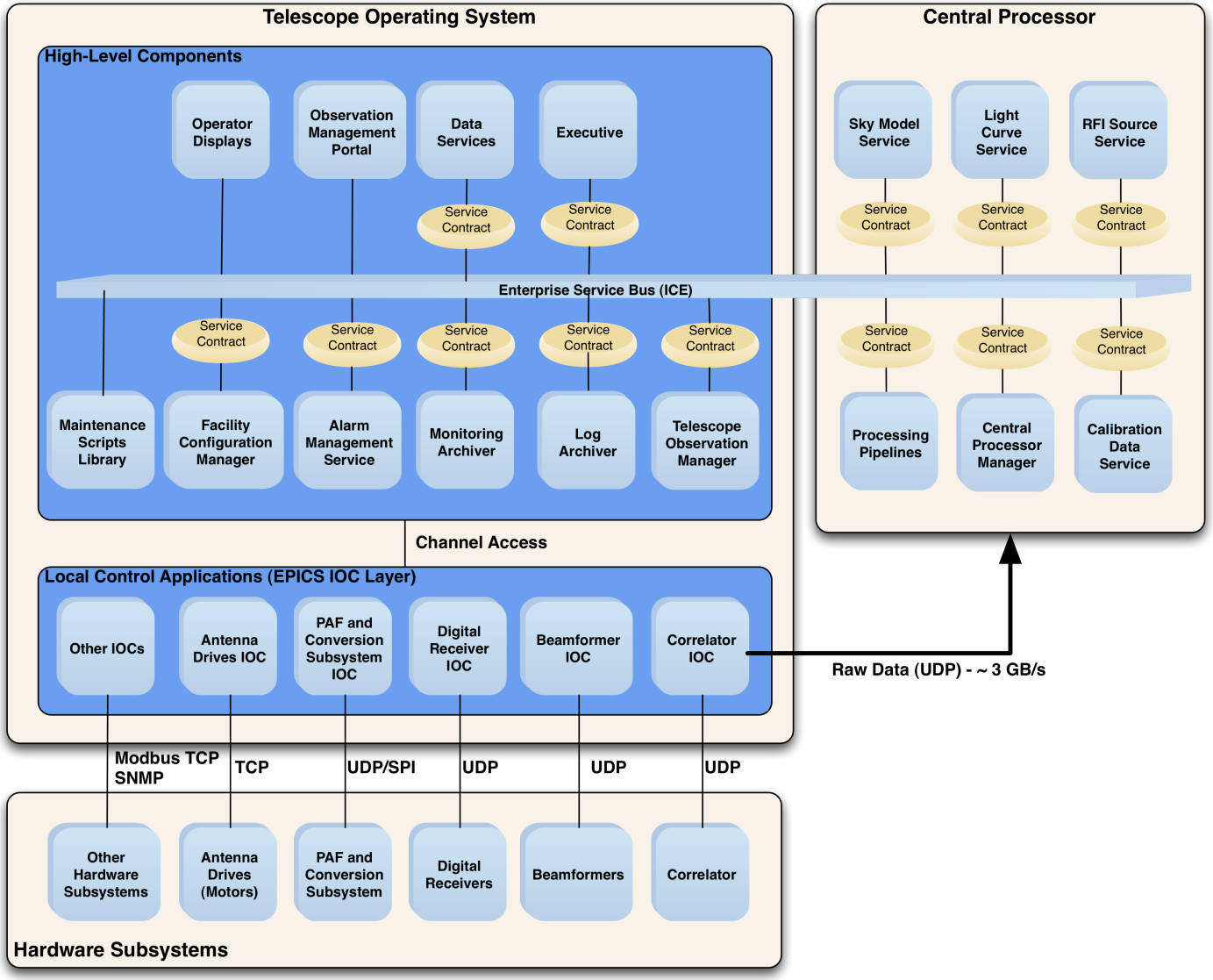


CSIRO

Architectural Goals

- The structure of the system
  - Top-level decomposition of the software system
  - Assignment of responsibilities to components
- Definition of interactions between components
- Loosely coupled components
  - Hardware platform independence
  - Language independence
  - Implementation independence
  - Location and server transparency

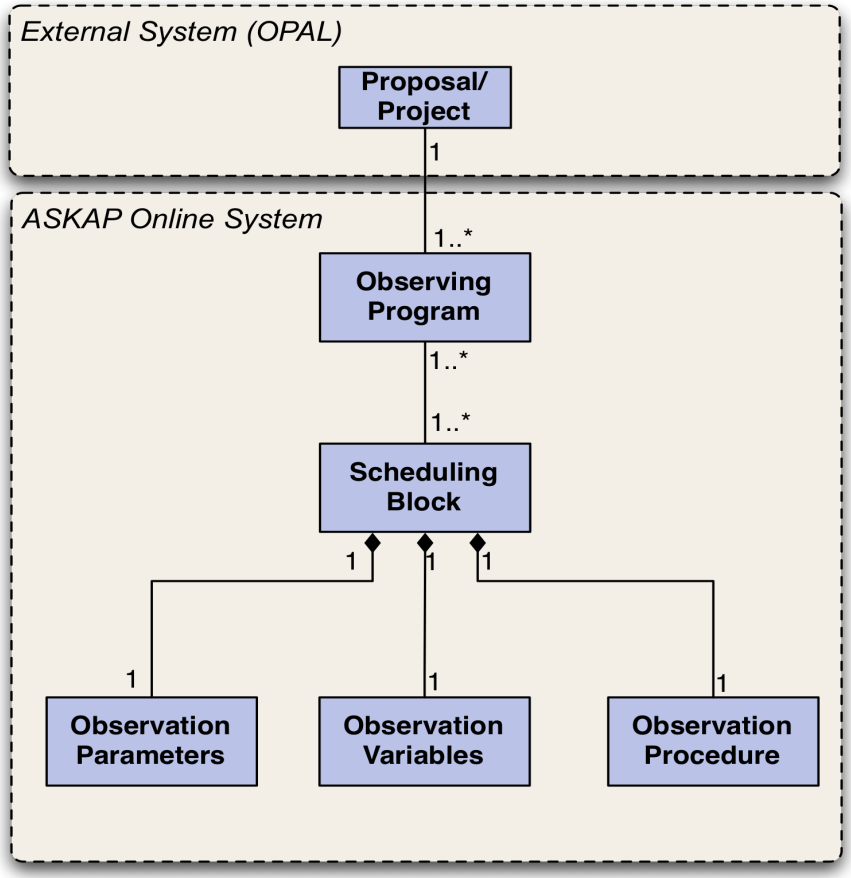
ASKAP Software Architecture



CSIRO

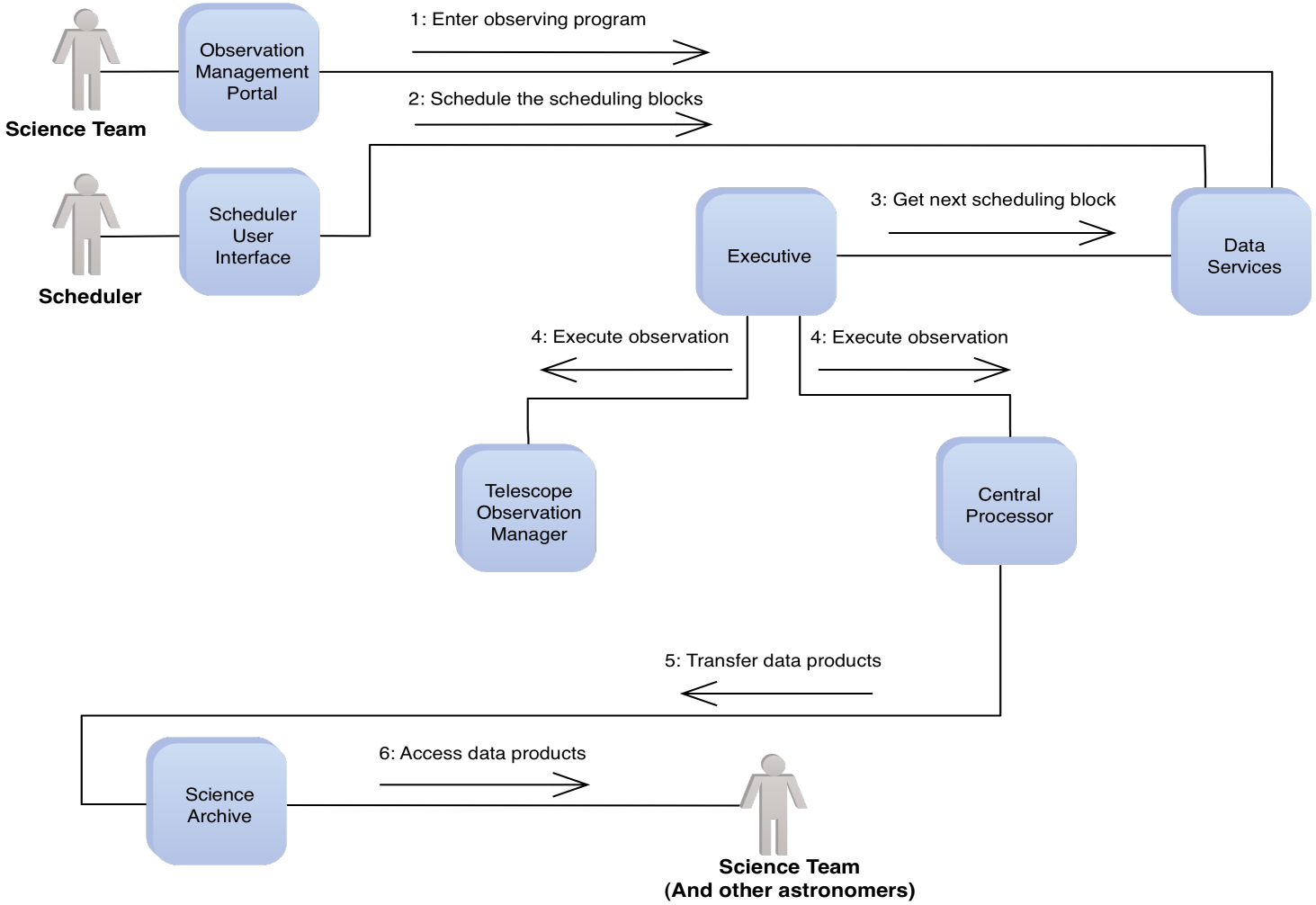
Scheduling Blocks

- The scheduling block is the primary object of execution for the telescope operating system, central processor and archiving components
- Contains information necessary to carry out an observation:
  - Observation Procedure
  - Observation Parameters
- Plus some state
  - Observation Variables
- Can be asssociated to multiple observing programs



CSIRO

Process View



CSIRO

Key Technologies
ICE

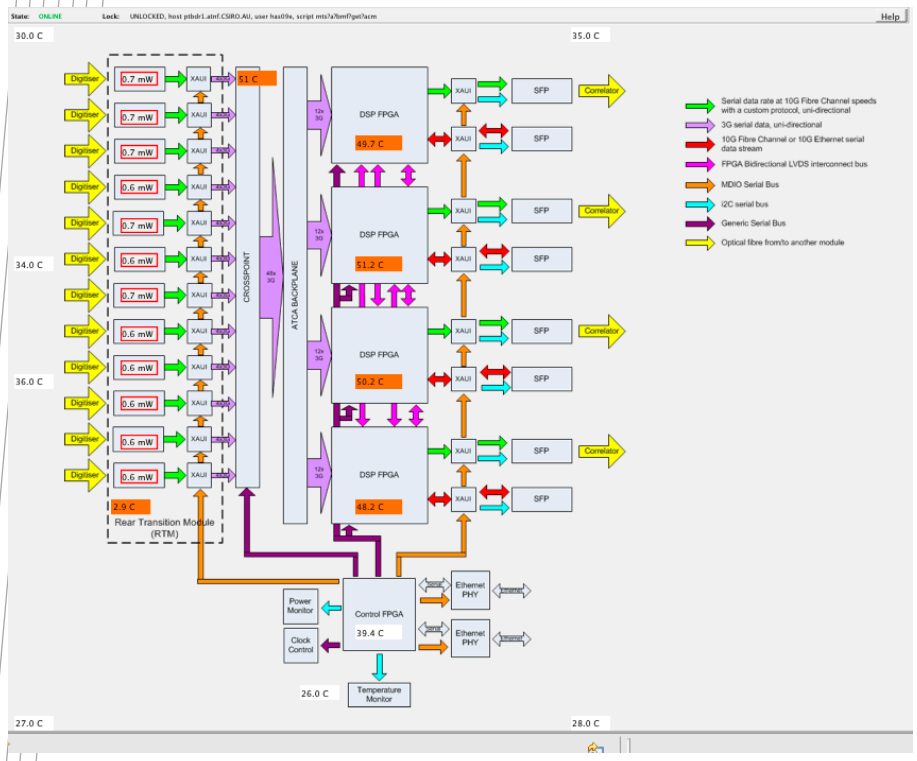
- Adopted in 2009
- Very good core functionality
  - Interface Definition Language (SLICE)
  - RPC communication paradigm
  - Locator service
  - Easy client/server code development
- Publish/Subscribe Issues (version 3.4.1)
  - Inability for subscriber to identify IceStorm server has died (no auto-connect)
  - No message queues or cache
- Looking into ways to improve

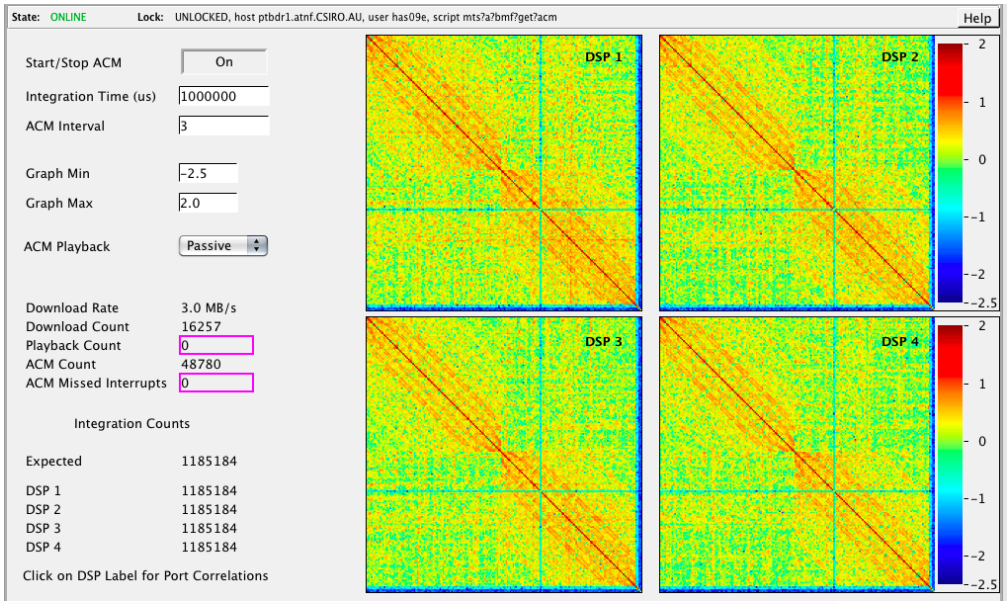
Key Technologies
EPICS

- Adopted in 2009
  - 6 months operations on Parkes Testbed Facility (PTF) System
  - BETA (6 Antennas): 300,000 I/O points, 40,000 points to be archived
  - ASKAP (36 Antennas): 1M I/O points, 240,000 points to be archived
- Solid and robust communication protocol optimised for control systems named Channel Access
- Flexible, modular, multiple platform, lots of drivers available and tools
- Big, friendly and growing community
- EPICS Tools
  - “Off-the-shelf” drivers: SNMP, Modbus-TCP, TCP/ASCII(StreamDevice)
  - ASYN a C/C++ EPICS Driver framework
  - SNL a language to implement state models and sequences in IOC
  - PyEpics3, Python CA Client Library
  - EDM and CSS BOY for Rapid GUI development

Key Technologies
EPICS – Control System Studio

- Best OPI Yet (BOY)





CSIRO

Software Development Process

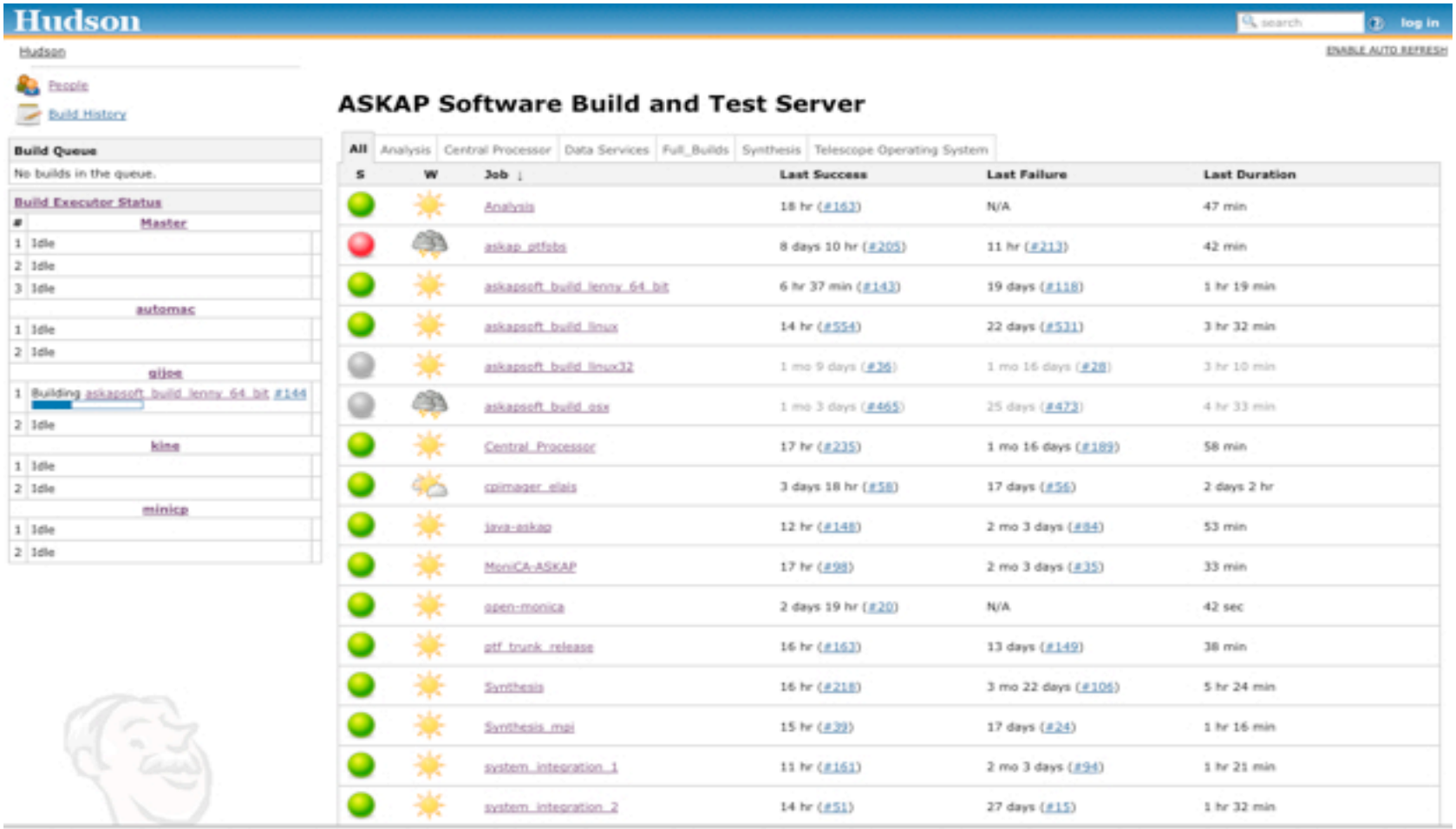
- Iterative light-weight approach
- Integrate early, integrate often
  - Monthly software releases
  - Fixed date, variable features and bug fixes
- Milestone = Delivery of Software
  - Not using Gantt Charts
  - Highly adaptable to change of requirements
- “Document as you code”
  - …but focus on code not heavy (useless) documentation
  - Automated tools to generate API documentation
- “Test as you code”
  - Unit and functional testing as part of each module or component

Software Development Process
Essential Tools

- Version Control System: Subversion
- Feature-rich build system
  - Recursive builds
  - Integrates many Third-party packages
  - One command: Update ➔ Compile ➔ Docs ➔Test ➔ Release
  - Wraps constructions tools: scons, make, autoconf, setuptools and ant
  - Written in Python
  - Used by all software developers (TOS, CP and Science Archiving)
- Redmine http://www.redmine.org
  - Project Management Tool
  - Issue (bug) Tracking with Milestone grouping
  - Sub-projects
  - Wiki
- Hudson http://hudson-ci.org
  - Continuous Integration Tool
  - Nightly builds, unit and functional testing

Software Development Process
Essential Tools

- Hudson dashboard shows status of all build executors and jobs



CSIRO

ASKAP TOS Status

CSIRO

ASKAP TOS Status
Highlights

- Most of the high-level components and EPICS IOCs (local control) implemented
  - Except for FCM, OMP, Correlator
- TOS-0.9 current release
  - In Production in Parkes Test Facility (PTF) System
- First TOS Testing at MRO in Sep 2011

CSIRO

ASKAP TOS Status
Next Steps

- TOS-0.10 (18 Nov 2011)
  - Support baseband data recording (after beamforming)
  - Software correlator for debugging with up to three antennas
  - Specialised SB to undertake observation (tracking 8 – 10 hrs)
  - Python Script Library for maintenance operations
- Getting results with BETA (6 Antenna with PAF)
  - Integrate Correlator Hardware/Firmware
  - Installation/verification of the TOS @ MRO
- Control System Studio
  - Integrate it into our build system
  - BEAST as a replacement of Motif-based Alarm Handler (ALH)
  - BOY as a replacement of EDM
- Facility Configuration Management
- Observation Management Portal
  - Evaluate different web technologies, in particular Google Web Toolkit (GWT)

Lessons Learned

CSIRO

Lessons Learned
Things that worked well

✓Did not separately managed control and other software

- Computing IPT responsible to deliver all ASKAP software
- Minimised risk of arbitrary interfaces and duplicated software development effort
- Cornwell and Glendenning, “Software Development for SKA”, Experimental Astronomy (2004) 17: 309–315

✓Designed the overall (observatory) Software Architecture early

- Helped identify users and subsystems (components):
  - Define responsibilities, derived requirements and specifications
- Realised a software design early that fulfill key project requirements
- Defined common data entities (SB, configuration parameters)
- Get “key” software architects/developers from control, science processing and science archive domains involved
- Software Architecture is not static and most likely needs to adapt to changes in requirements

Lessons Learned
Things that worked well (cont.)

✓Did not reinvent the M&C wheel

- Code re-use is our default option not the exception
- Looked at large experimental physics facilities: VLT, Gemini, ALMA, LOFAR, LHC, ASP and others
- Looked at conference papers: ICALEPCS, SPIE and ADASS
- Produced an evaluation report

✓Integrate often and integrate early

- Use an automated tool not a manual procedure, i.e. Hudson
- Build and test the end-to-end system
- Build and use hardware emulators (where practical)

✓Invested in software development infrastructure early on

- Version control: subversion, git, mercurial
- Build system: update, compile (including third-party software), run unit/ functional tests, release

Lessons Learned
Things that worked well (cont.)

✓Iterative Software Development Process

- Milestone driven by software (concrete) releases, with built-in documentation – 4 weeks release cycle
- Fix release date, variable set of features
- Best fit for adaptability (changes in requirements)

✓Used a descriptive language for software (component) interfaces

- SLICE (ICE's inteface description language) has served us very well

✓Get domain experts involved during system verification

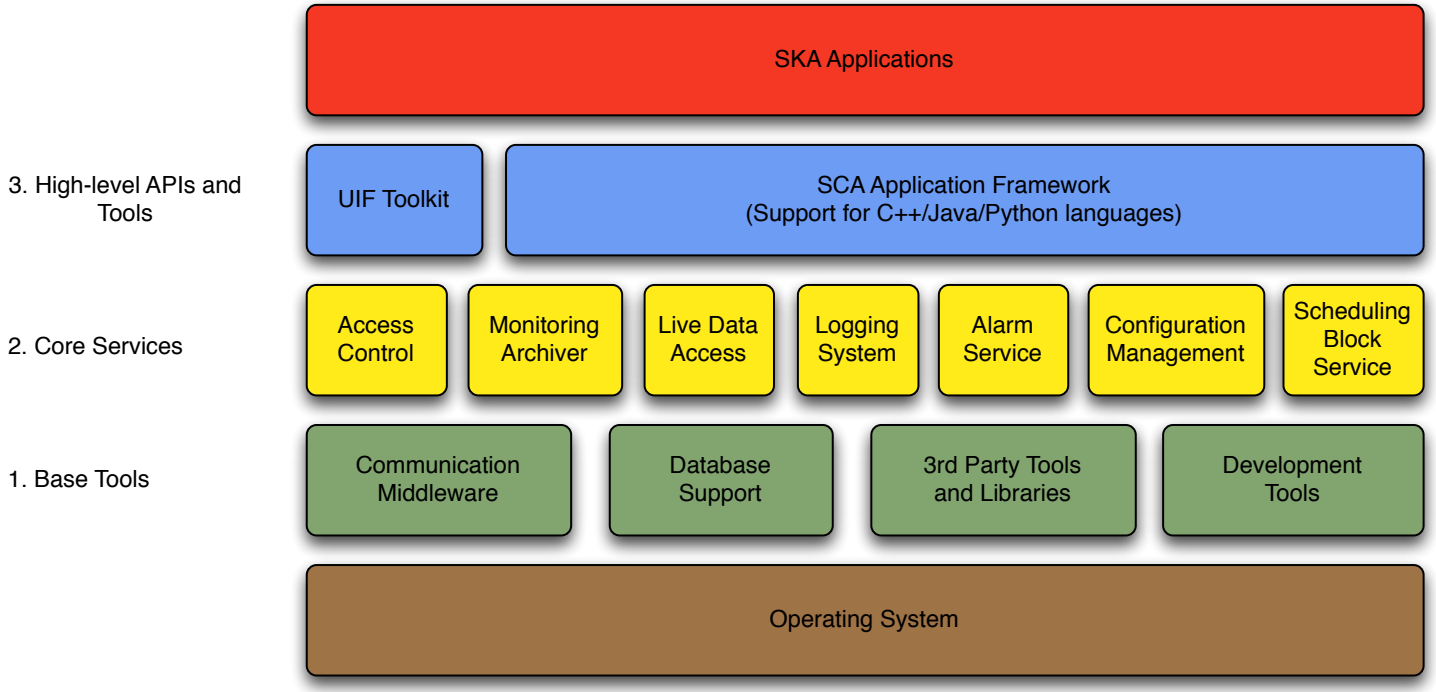
- We call them "instrument specialists"
- Experts in radio interferometry with programming skills (Python)
- Write complex system verification scripts
- Test usability of the software system

Lessons Learned
Things that we are working on it

- Get (demand) Interface Control Documents (ICDs) between hardware and software early
  - Essential to build (hardware) emulators
  - Collaborative effort, even if sub-contracted externally
  - Adapt to hardware developers “work culture”
  - System Engineers with some software expertise should ease the pain of managing the process of ICD production
- Avoid too early optimisation in hardware and software
- Get Telescope Operation Model as early as possible
  - Essential for determining software requirements
  - Get people with radio telescope operations experience on board

Lessons Learned
Things that we are working on it

- Define a Common Software Framework
  - Framework to build and run Telescope Applications
  - Separates Technical vs Functional (Applications) concerns
  - Common Data Model definition and implementation for all type of applications
  - Unifies API look-and-feel across multiple programming languages



CSIRO

Australia Telescope National Facility
CSIRO Astronomy and Space Science
Juan Carlos Guzman
ASKAP Computing IPT

Phone: +61 2 9372 4457
Email: Juan.Guzman@csiro.au
Web: http://www.atnf.csiro.au/projects/askap/

www.csiro.au

Thank you

Contact Us
Phone: 1300 363 400 or +61 3 9545 2176
Email: enquiries@csiro.au Web: www.csiro.au

CSIRO