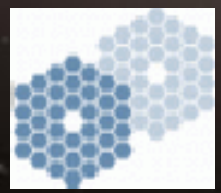
Trends in Software for large astronomy projects

G.Chiozzi, A.Wallander – ESO, Germany
K.Gillies – Gemini Observatory, La Serena, Chile
B.Goodrich, S.Wampler - National Solar Observatory, Tucson, AZ
J.Johnson, K.McCann – W.M.Keck Observatory, Kamuela, HI
G.Schumacher – National Optical Astronomy Observatories, La Serena, Chile
D.Silva – AURA/Thirty Meter Telescope, Pasadena, CA




Aspects analyzed

- Timeline
- Challenges
- Architecture
- Frameworks
- Development methodologies
- Technological implementation
  - HW platforms
  - Operating systems
  - Programming languages
  - User Interfaces.

Timeline

1990

1995

2000

2005

2010

2015

2020

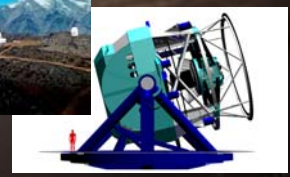
Keck

VLT/VLTI

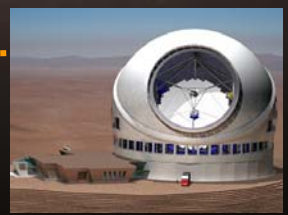
Gemini N/S

ALMA

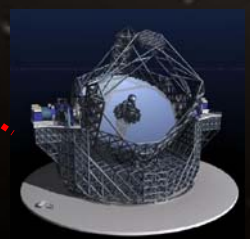
LSST



ATST



TMT



E-ELT

Challenges of new projects

- Synchronized multiple distributed control loops (wave front control)
- Multi-level off-loading schemes
- Fault detection, isolation and recovery (E-ELT M1: 1000 segments with actuators and sensors)
- Operational efficiency (TMT requirement: on target in <5 minutes).

Architecture

- All major facilities in operation: three-tier architecture
  - High-level coordination systems
  - Low-level real time control computers (LCUs)
  - Devices with limited degree of intelligence
- Fairly independent sub-systems: slow correction offloading
- Wave front control (adaptive optics and interferometry) introduces new requirements:
  - Distributed real time synchronization and feedback
  - Significant physical separation
- Systems of systems, often heterogeneous
- LCUs role is eroded on both sides.

Frameworks

- A uniform software framework has a value in simplifying development and maintenance
- Isolate application from middleware providing a layer of common services
- Separation between technical and functional architecture now formally adopted.
- Component based architectures emerged as particularly useful in distributed systems
- Sharing the technical framework would allow sharing functional components .

Frameworks adopted:

- Keck and Gemini: EPICS, RTC
- ESO Paranal and La Silla: VLT CCS
- ALMA and other projects: ACS
- ATST: ATSTCS

Common services:

- Connection
- Event
- Command
- Logging
- Persistent store
- Error handling

Development methodologies and modeling techniques

- Our constraints:
  - Multi-year observatory design periods
  - Review structure and process imposed by funding agencies is oriented to a waterfall approach
  - Floating requirements
- Methodology evolution:
  - Mid ’80s/ mid ‘90s: Structured programming
  - Mid ’90s/ beginning 2000: Object Oriented and UML (*pragmatic approach*)
  - Now: SysML, agile methodologies:
    - Requirement management and traceability
    - Integration in a coherent system model as seen from different disciplines .

Hardware platforms

- In most existing observatories:
  - High level coordination → general purpose WS
  - Real time → Local Control Units (often VME)
  - Devices attached directly to VMEs
- Many more options are available now:
  - High level coordination → Personal Computers
  - (Soft) Real time → PC with real time OS
  - Intelligent devices on ETH or industrial buses (CAN)
  - (Hard) Real time → DSPs and FPGAs
- Clusters for raw computing power
- Virtualization under evaluation. Trend for the future?

Operating systems

- The 1990s
  - Proprietary UNIX
  - Proprietary RTOS (VxWorks dominating)
- The turn of the century: open source
  - Linux
  - Real Time Linux
- And now?
  - Questioning Linux
  - Solaris re-emerging
  - Open source to stay (Solaris)
  - MsWindows (and OPC)?
  - Other players?

- OS neutrality
- Real time Java
- QNX
- LabVIEW and LabView-RT
- PLCs
- FPGAs and DSPs .

Programming Languages

- The core language(s):
  - Mid ’80s/ mid ‘90s: C domination
  - Mid ’90s/ beginning 2000: C++ takeover
  - Now: Java explosion, C++ decline, C holds

Language	Keck	VLT
C	251050	246738
C++	0	84400
Capfast	130116	0
Tcl/Tk	9408	81657
Others	118144	64136
Total	**508718**	**476931**

- The glue: from Tcl/Tk to Python and over
- LabVIEW’s role growing
- We have to cope with:
  - Different languages for different purposes
  - Highly distributed systems .

User Interface

- A challenging area. Growing complexity.
- We are comfortable with Engineering UI development
- We do not have skills for good Operator UIs
- Java and Tcl/Tk the most used.
- GUI builders are not adequate
- Rapid prototyping: necessary, but with a dark side
- We cannot afford specialized UI development teams .

Conclusion

New facilities are NOT scaled up versions of existing ones.
Paradigm changes may be required

- Analysis of control system evolution in observatories is on-going
- We have identified clear common trends
- We aim at:
  - Sharing lessons learned
  - Identifying areas for cooperation
  - Sharing architectural elements and infrastructure
- Cooperation is made easier by international collaborations and the open source movement .

Questions?

The authors represent just a subset of the projects in astronomy. Many more colleagues in the astronomical observatory community have given their ideas and time as we have developed this paper.

Web Links

ESO	www.eso.org – Email: gchiozzi@eso.org
W.M.Keck Observatory	http://www.keckobservatory.org
Gemini Observatory	http://www.gemini.edu
ALMA	http://www.alma.cl
ATST	http://atst.nso.edu
LSST	http://www.lsst.org
Thirty Meter Telescope	http://www.tmt.org