A wide-field survey of milli-arcsecond software

A. Puglisi, P. Grani, F. Rossi on behalf of the AO group

First TETIS workshop, 27-29 Oct 2020, zoom
.

Waiting for AI to write its own software, you need people

SW "only":

L. Fini (retired in 2018)
A. Puglisi
P. Grani
F. Rossi
F. Tosetti (gone to the private sector)

Major contributions:

M. Xompero
L. Busoni
A. Riccardi
D. Zanotti (gone to Kentucky)

Simulation code:

G. Agapito
C. Plantet
F. Quiros (gone to L.A.)
M. Carbillet (gone to France)

Code snippets from:

R. Briguglio
S. Esposito (really!)

… and I am surely forgetting someone

And to keep people busy, you need projects

LBT:

Adaptive Secondary
Pyramid WFS
In house middleware + ICE
ARGOS (multi-laser SH WFS)
Elab_lib (more on this later)

Lab experiments:

HOT (w/ ESO), and many small things

Cophasing @ telescopes:

WHT
APE (VLT experiment)

Full-stack:

Optomechanics control, state machines, UI, data reduction...

End-to-end Simulation:

CAOS
PASSATA: end-to-end AO simulator

- with GPU
- used for everything.

ESO instruments:

ERIS
Upcoming: MAORY, MAVIS, M4, MORE...

“more” is not an instrument

Languages:

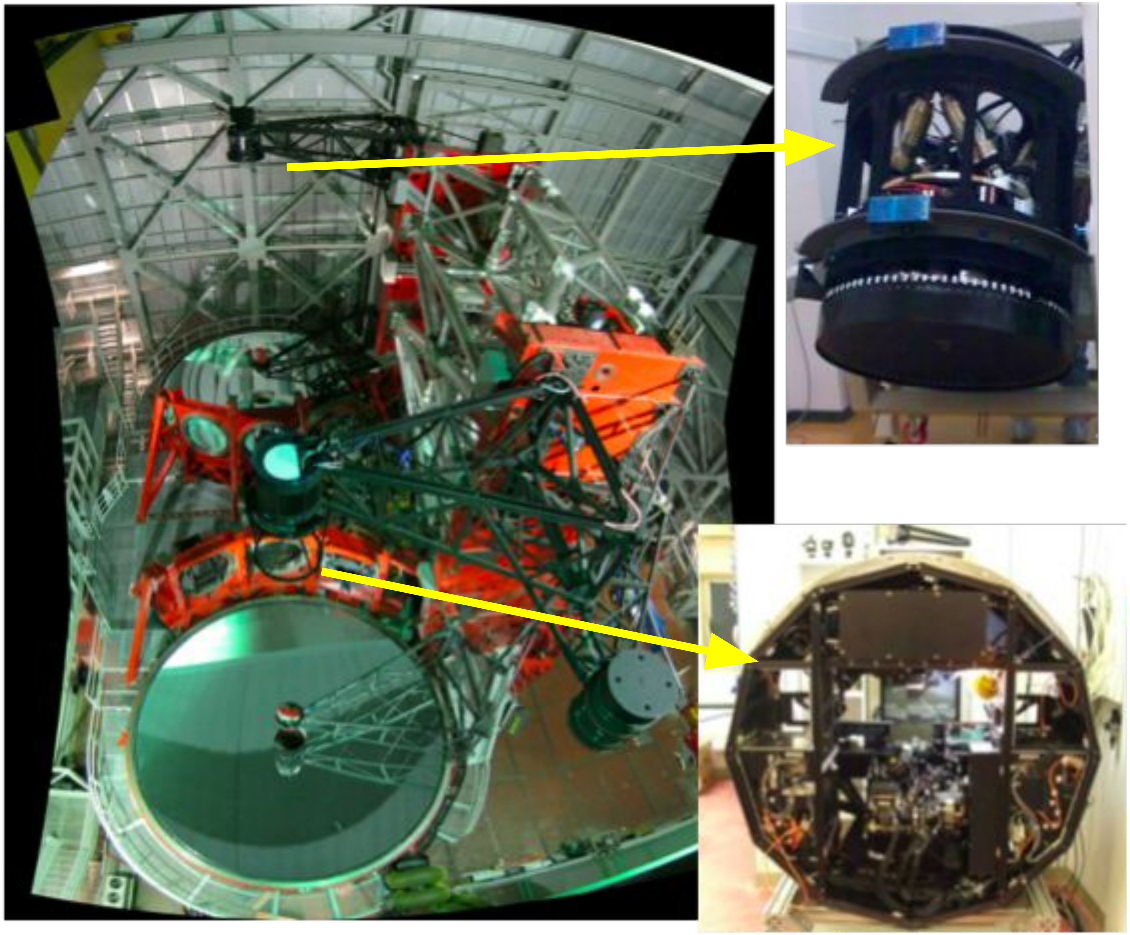
IDL - more on this later
C/C++
Python
some Perl, bash, Matlab, TCL...

One project has used more man-years than all the others

LBT-centric five-year plans

AD 2000 - beginning of time because I wasn’t there before. LBT sort of starts

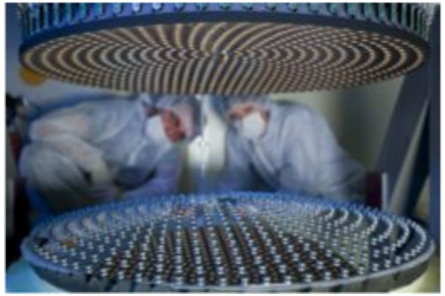
year	**LBT**	**Other things**
2000	begins!	Not many
2005	is late...	... so we do other things
2010	starts to work!	can’t really stop them now, no?
2015	commissioning!	more other things
2020	upgrade!	and now even more



Adaptive secondary

Pyramid WFS

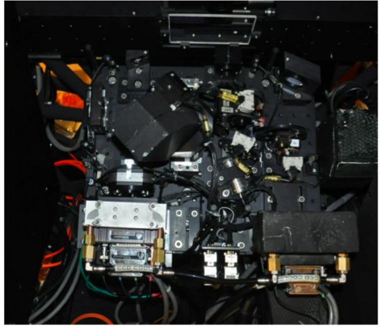
Adaptive secondary



- Custom HW and firmware by Microgate and ADS
- Low-level control SW (UDP frames in C++)
- Quasi-realtime diagnostic & safety
- High-level control SW (IDL)
- State machine (Python)
- Operator GUI (Python/C++)

Precursor to VLT UT4, M4...

Pyramid WFS



- Many different optomechanical devices
- Low-level control SW (C++ w serial lines)
- High-level control SW (Python)
- State machine (Python)
- Operator GUI (Python/C++)
- A smorgasbord of data reduction SW

Most other things are then WFSs: ARGOS, cophasing, ERIS, MAORY...

Working at LBT...

February 9th --March 17th .

How design methods have changed

2005: bottom-up

”this is the system, make it work”
(probably unfair to the adaptive secondary guys).
Lots of fun.

2015: waterfall

First, write a 200-pages document of how you think you will implement things, even if you don’t really know.

…the real difference

Experience has proved that the amount of work is similar, but:

with waterfall, you write the documentation at the beginning, so you have it at the end (even if slightly incorrect)

with bottom-up, you write it at the end, but it’s boring so you don’t.

Donald Rumsfeld’s guide to SW development:

	Known knowns	**Known unknowns**	**Unknown unknowns**	Final result
Bottom-up	Ok	We’ll do it somehow		Sort of works, no docs
Waterfall	Ok		Oh no...	Sort of works, wrong docs

It’s not only control SW - part I

The importance of simulation

Two major end-to-end AO simulators, both with improbable names: CAOS and PASSATA

And both in IDL.

These simulators have been used for **everything**.

…in a somewhat unsuitable language

PASSATA stretches IDL to the limit:

- Object oriented
- Custom GPU modules

IDL cons-and-pro:

- Large codebases are difficult
- Enabled contributions from several non-SW people who probably wouldn’t have otherwise.

Example: most of the LBT Adaptive Secondary SW is in IDL, and has been written by people whose primary qualification was not SW.

Now slooooowly moving to Python.

It’s not only control SW - part II

The importance of data reduction

LBT introduced the concept of “elaboration library”:

1. Implement a way of saving a snapshot with:
   - A burst of AO telemetry data (a few seconds worth -> about 100MB)
   - The FULL system state and configuration (more easily said than done)
   - Bonus if you can coordinate a scientific exposure from the instrument at the same time (even less easy)
2. Remember to save said snapshots every now and then
3. Archive everything and develop offline data analysis routines (the “library”) that crawl over this data and can answer many, many questions

The LBT snapshot archive is now ~20TB and is still used almost daily for diagnosis, statistics, etc.

-> Similar tool in development for ERIS (not part of the ERIS software…. yet)