Designing and Implementing LabVIEW Solutions for Re-Use

Presentation to

14th International Conference on Accelerator & Large Experimental Physics Control Systems (ICALEPCS)

October 6-11, 2013

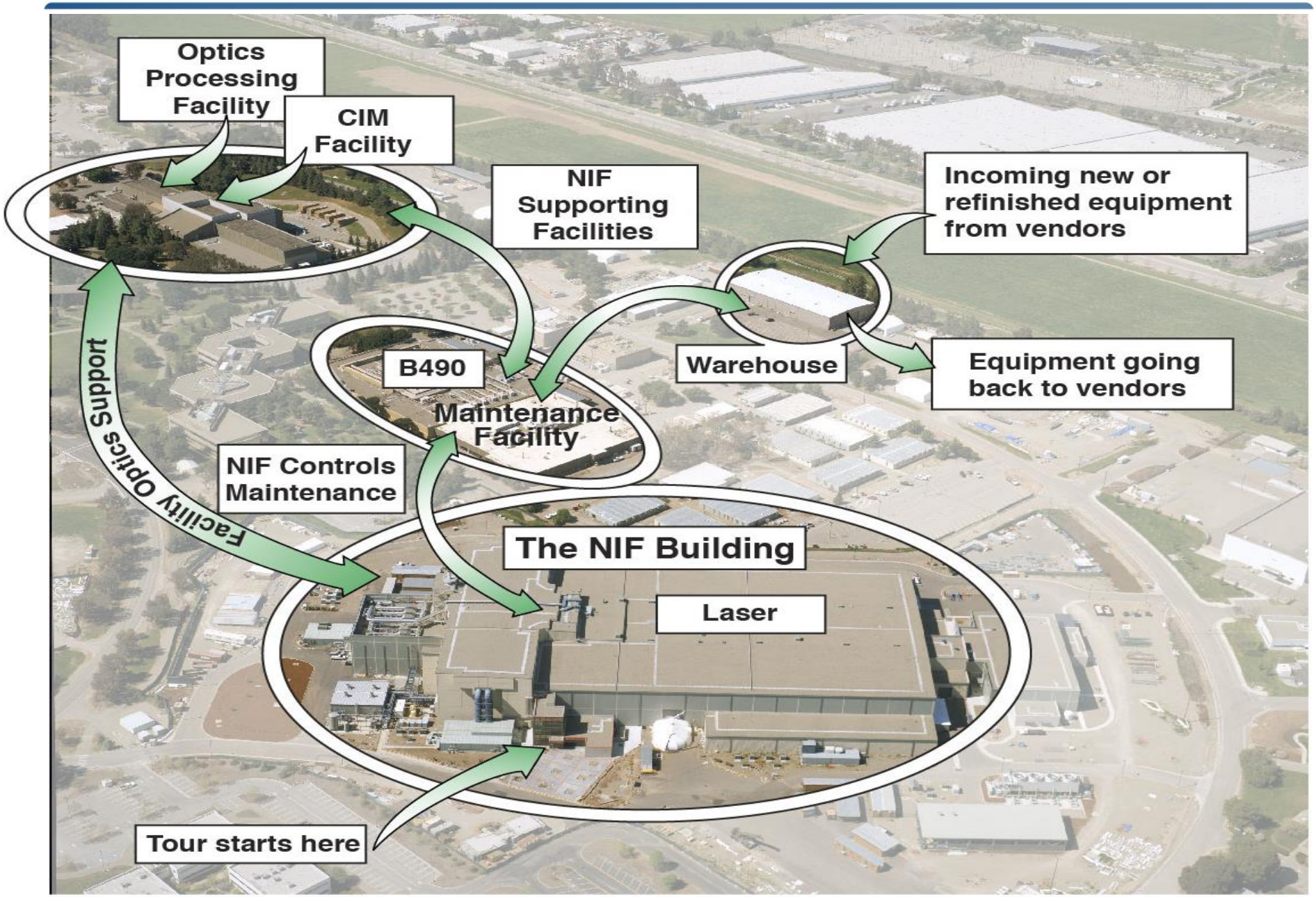
M. Flegel, G. Larkin, L. Lagin, B. Demaret

Lawrence Livermore National Laboratory • National Ignition Facility & Photon Science

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344

LLNL-PRES-644307

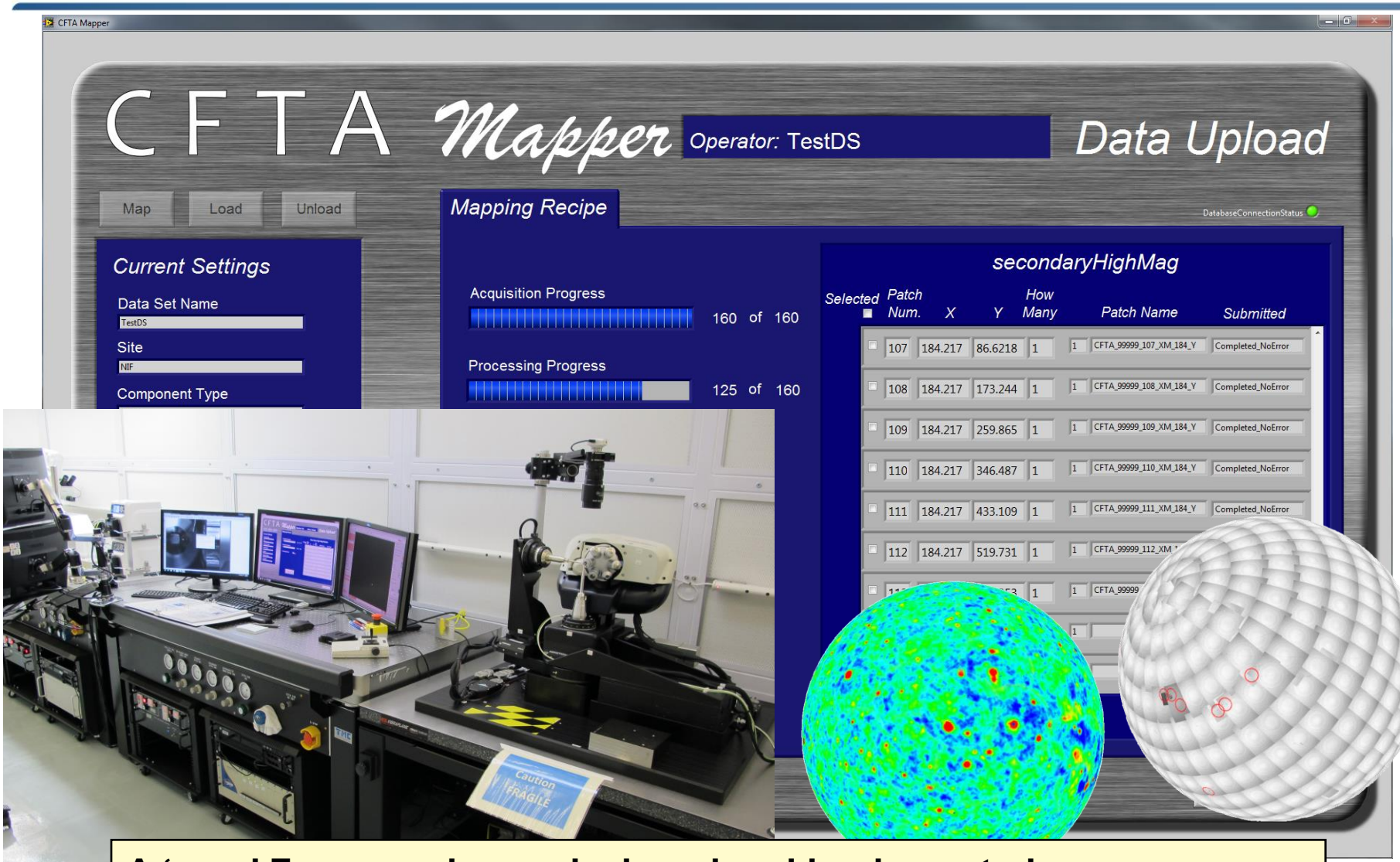
To operate the NIF requires support from many auxiliary support facilities.



The Optics Mitigation Facility, completed in 2010, helps meet NIF's requirement for near perfect optics.

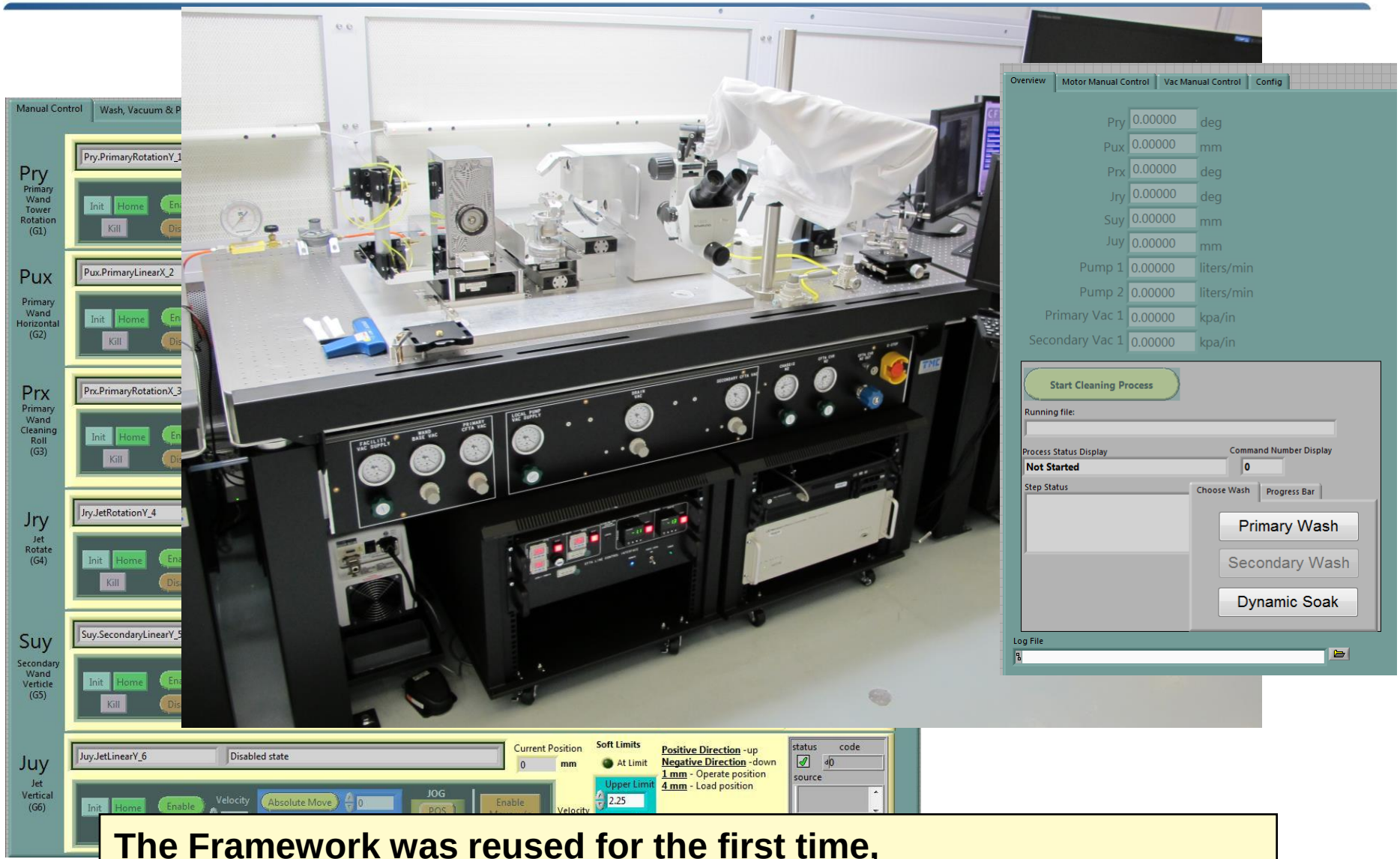
From its success, the Lab Systems team was formed to develop controls for machines using Software Engineering best practices.

The CFTA Mapping system characterizes capsule surface features using confocal microscopy.



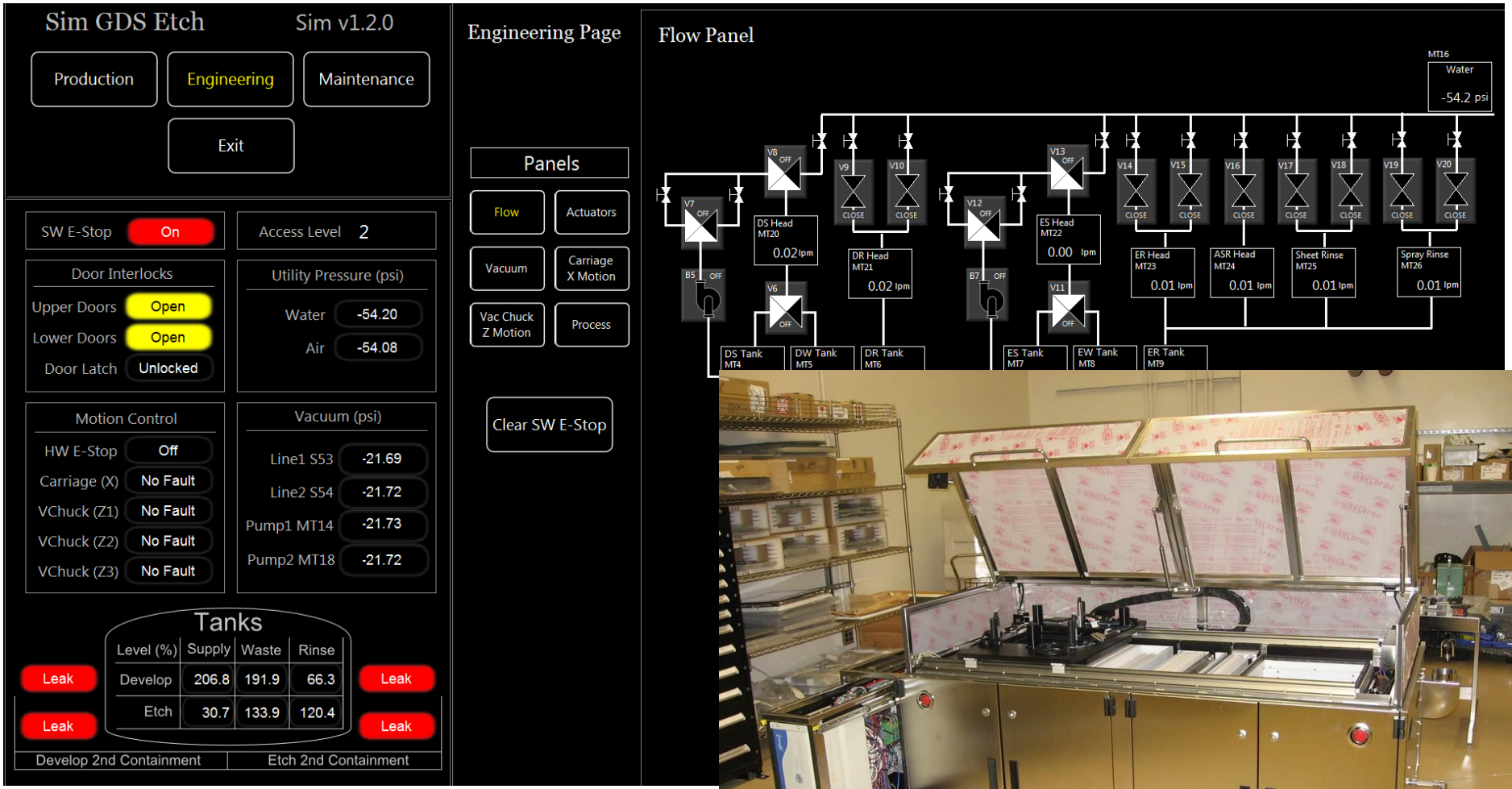
A ‘core’ Framework was designed and implemented.
The team was trained in OO and introduced to best practices.

The CFTA Cleaning station cleans capsules to improve performance.



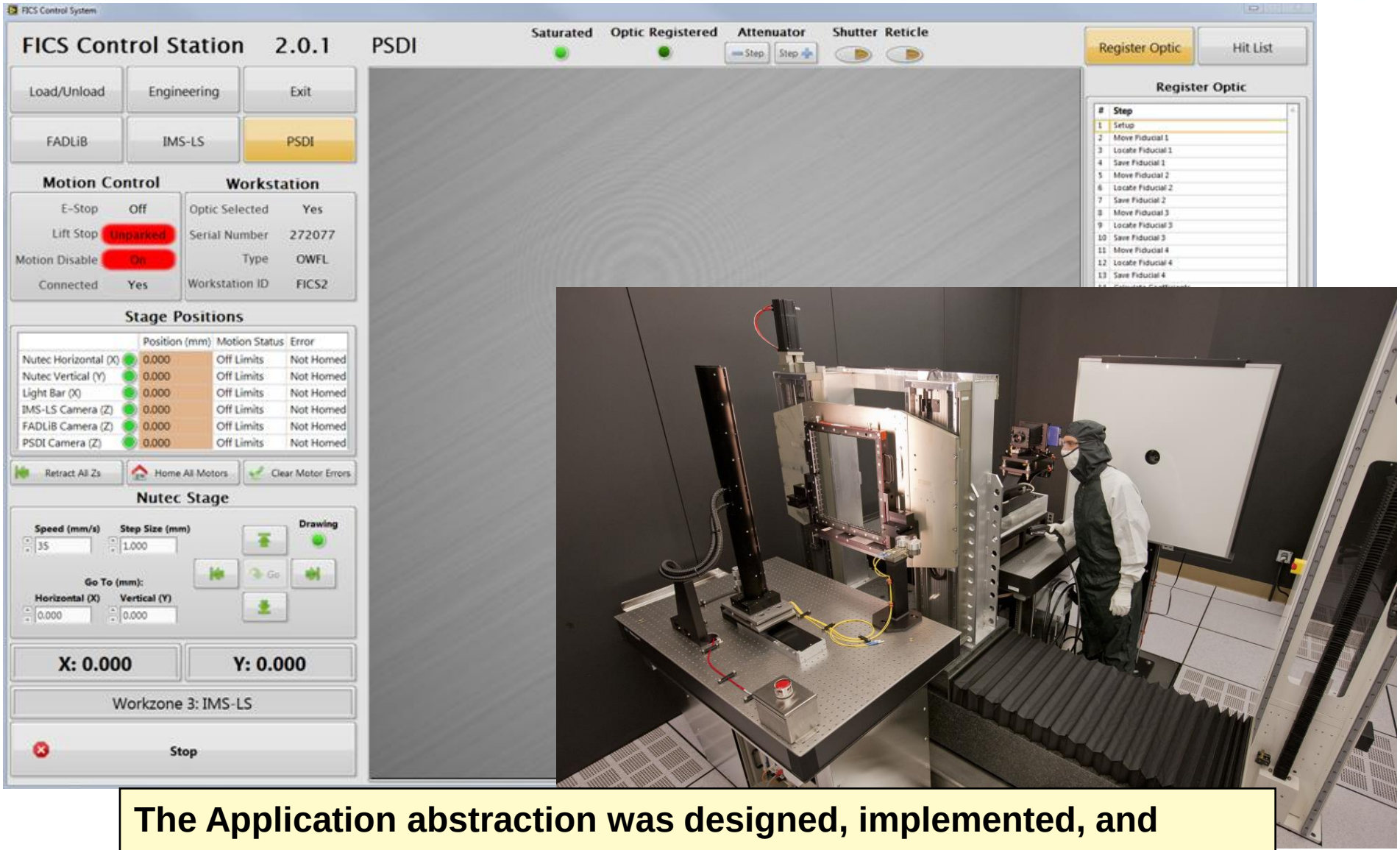
The Framework was reused for the first time,
and evolved to meet new requirements and lessons learned.

The Etching station develops and etches Grating Debris Shield optics.



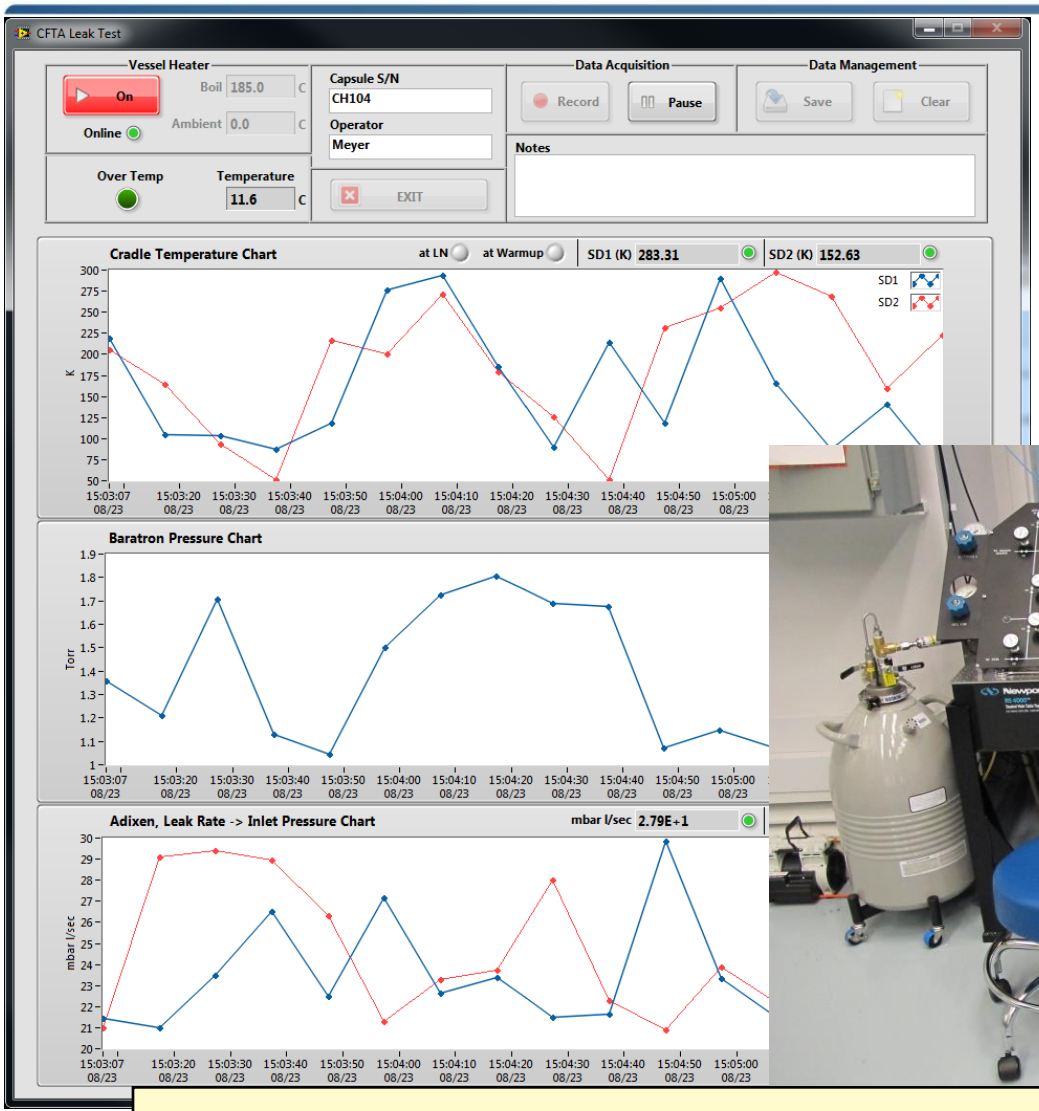
The GUI and Recipe abstractions were designed, implemented, and added to the Framework.

The Flaw Identification and Characterizations Station (FICS) characterizes optic flaws.



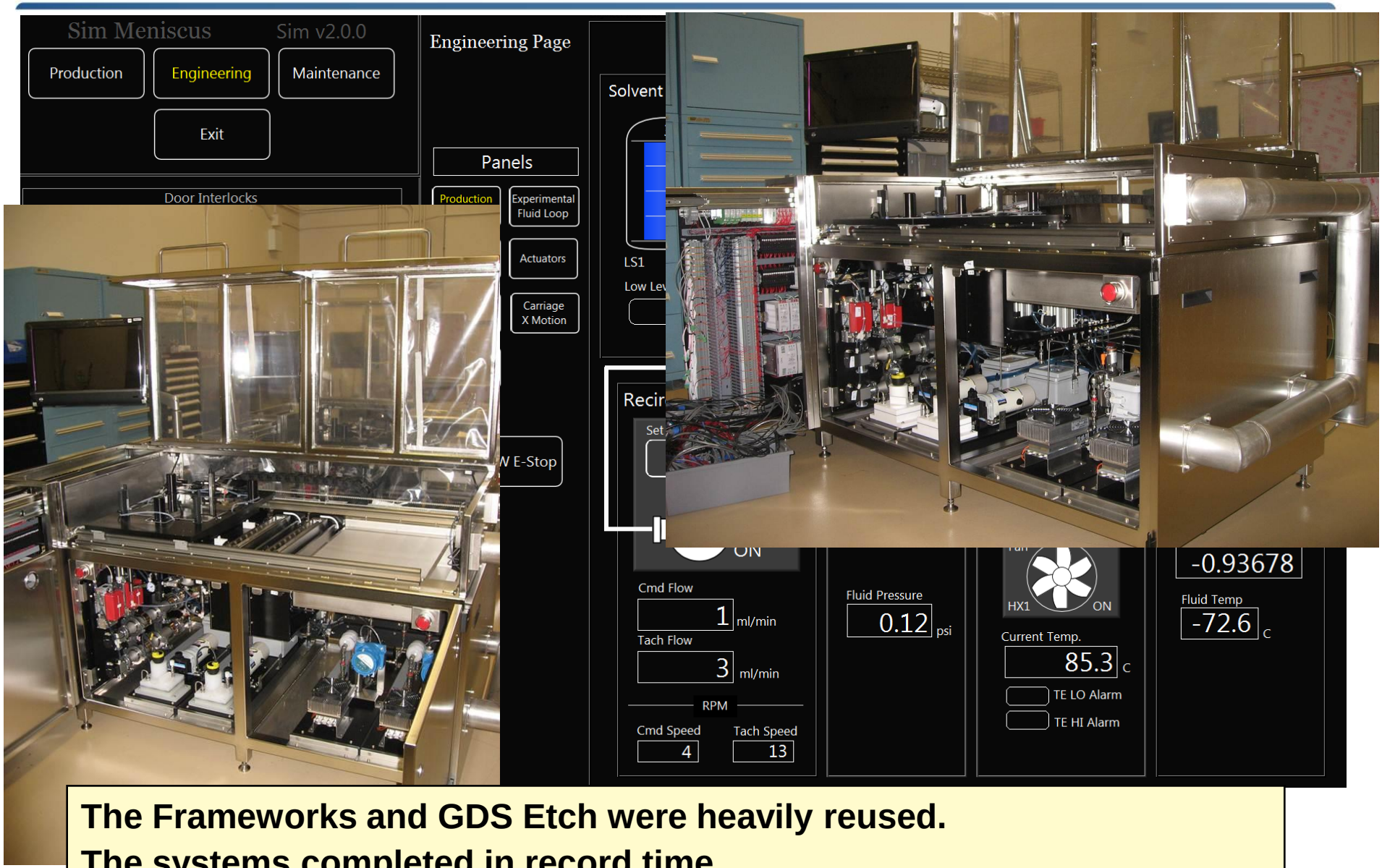
The Application abstraction was designed, implemented, and added to the Framework, the GUI abstractions were refined.

CFTA Leak Test station quantifies capsule integrity.



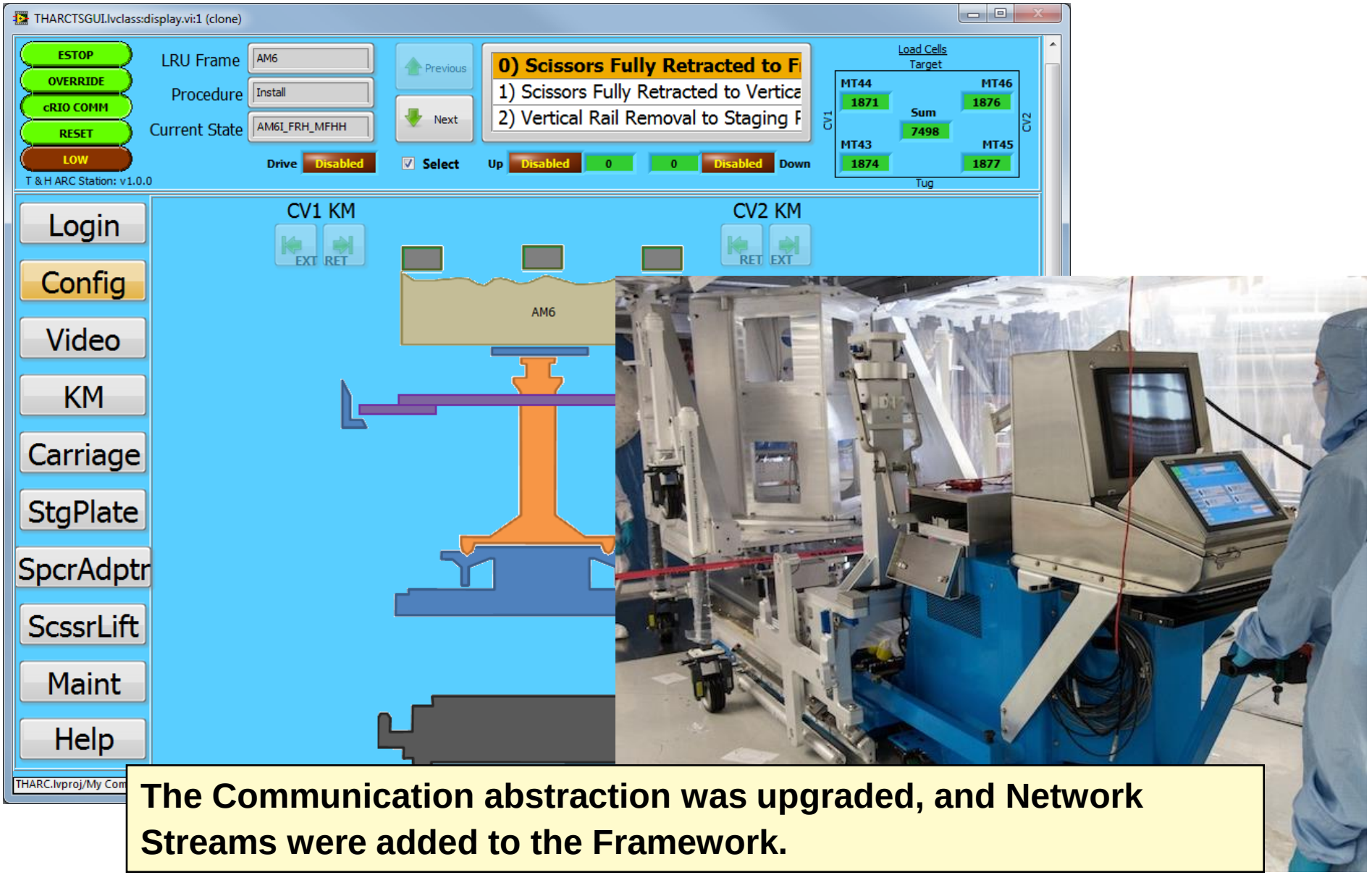
RS232 Communication and Protocol abstractions were added to the Framework.

Meniscus coaters apply PhotoResist or SolGel to optics.



The Frameworks and GDS Etch were heavily reused.
The systems completed in record time.

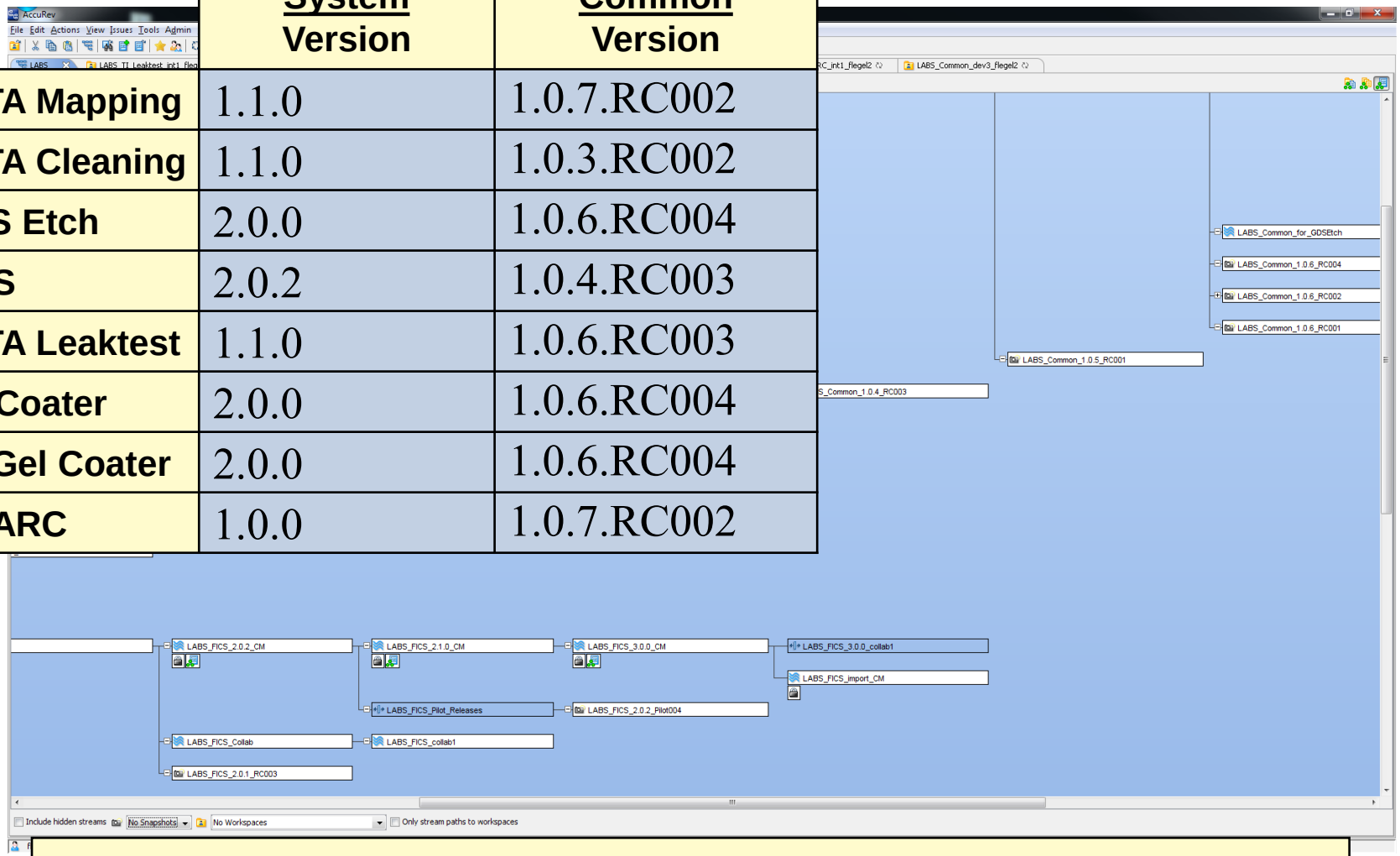
The ARC Transporter installs/removes AM6, AM7, and AM8 LRUs in the target bay's parabola vessel.



The Communication abstraction was upgraded, and Network Streams were added to the Framework.

These systems are under configuration management and have something in *Common* …

	System Version	Common Version
CFTA Mapping	1.1.0	1.0.7.RC002
CFTA Cleaning	1.1.0	1.0.3.RC002
GDS Etch	2.0.0	1.0.6.RC004
FICS	2.0.2	1.0.4.RC003
CFTA Leaktest	1.1.0	1.0.6.RC003
PR Coater	2.0.0	1.0.6.RC004
SolGel Coater	2.0.0	1.0.6.RC004
TH ARC	1.0.0	1.0.7.RC002



These systems are built from various releases of the same *Common* Framework.

Each system uses the *Common* Framework.

	Total		Reuse	
	Classes	**Methods**	**Classes**	**Methods**
CFTA Mapping	131	1098	66%	59%
CFTA Cleaning	83	611	80%	83%
GDS Etch	143	1331	58%	53%
FICS	173	1166	60%	68%
CFTA Leaktest	173	1166	60%	68%
PR Coater	83	652	70%	67%
SolGel Coater	161	1110	65%	71%
TH ARC	116	883	65%	75%
Average	**132.9**	**1002.1**	**64%**	**66%**

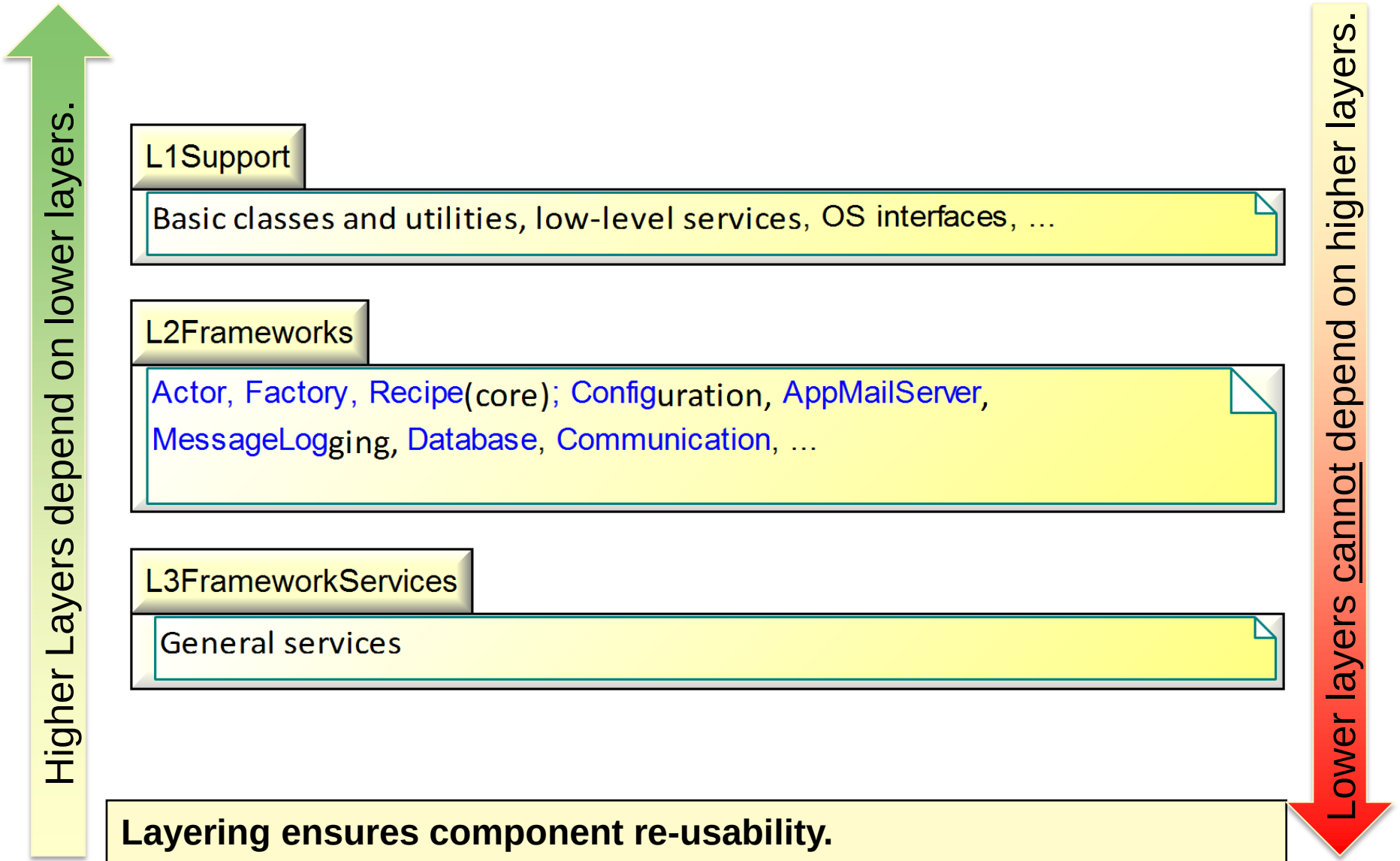
On average, 85 classes (including 665 methods) are reused.

Why did we do this?

- OMF was a successful project:
  - Completed in 15 months, 1/3rd the Java/C++ estimate
  - Applied software engineering best practices
  - Relied on LabVIEW's built-in GUI and hardware support
  - Focus of a highly respected NI case study
- All systems have something in common, they:
  - Control devices (drive motors, toggle switches)
  - Collect data (take pictures, generate signals)
  - Interact with the User / Operator
- So we created this Common, reuseable Framework
  - Used by all systems
  - Implemented with Best Practices
    - Designed, Coded, Tested
    - Configuration Managed

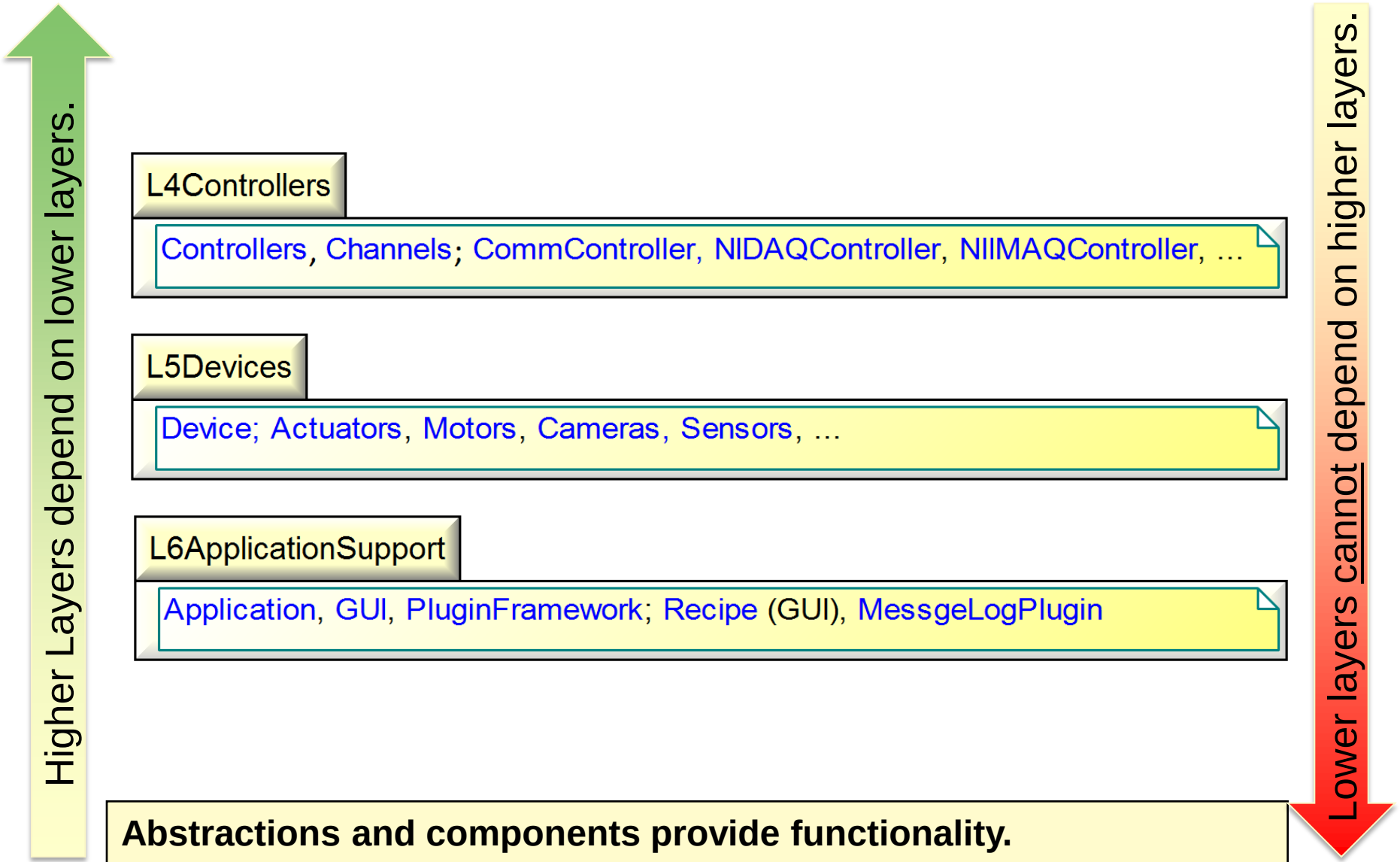
Reused code is 'free' – already developed, already tested.
Reused code is 'consistent' – architecture, look & feel.

What is the Framework?
… Code layering, and …



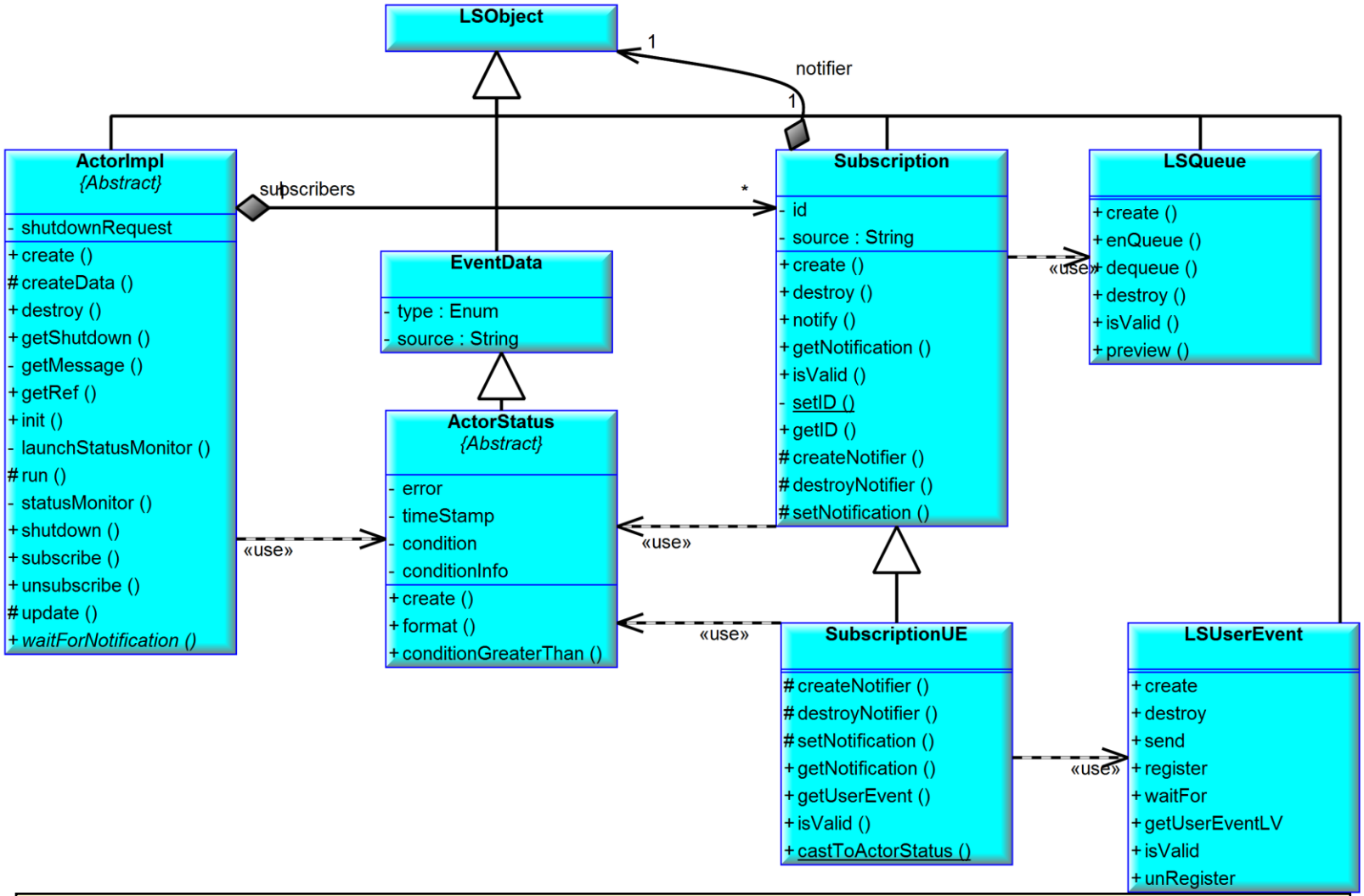
Layering ensures component re-usability.
Layers 1-3 are domain independent – designed for any system.

… abstractions and components.



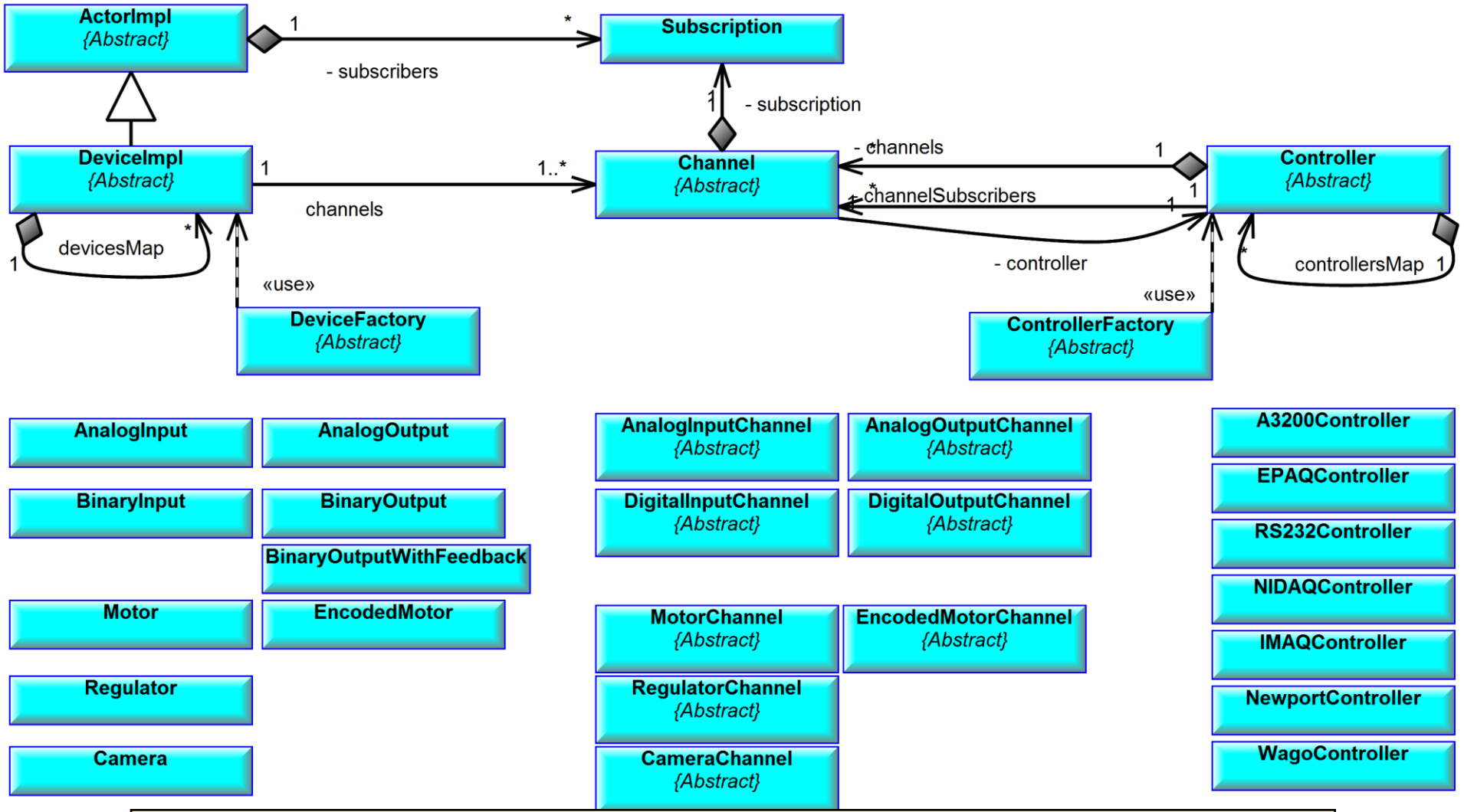
Abstractions and components provide functionality.
Layers 4-6 are domain dependent - designed for control systems.

Designed and implemented in Object Oriented LabVIEW.



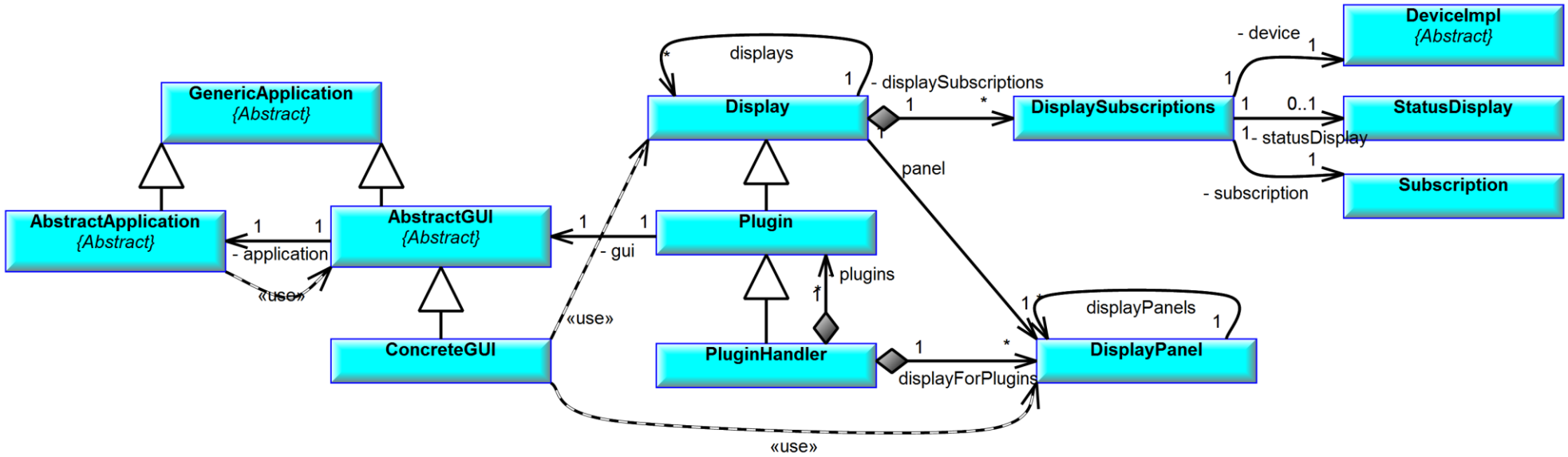
The Actor is the core of the Supervisory control and Hardware abstraction.

The Device / Channel / Controller classes are the core of Common hardware abstraction.



The hardware interface implementation is hidden from the device modeling.

The Application and GUI classes are the core of Common user interface capabilities.



Common displays can be created, shared between applications, and provide consistent look & feel.

What perceptions and concerns were encountered along the way?

- LabVIEW applications are ‘sub-standard’ and unstable for production.
  - ⇒ It’s how LabVIEW is applied, not LabVIEW itself.
- Why is it taking sooooo long?
  - ⇒ Early systems absorbed the cost for creating the Framework
  - ⇒ We evolved to a more agile development process.
  - ⇒ Deliver manual control of the machine, … then add features.
- You implemented what I asked for, but that’s not what I want!
  - ⇒ Requirements analysis includes GUI prototyping.
  - ⇒ Deliver manual control of the machine, … then add features.
- Individuals had their own software ‘toolbox’.
  - ⇒ We have a shared toolbox the whole team understands.

Software was audited and meets ‘DOE Order 414.1D’ for Risk Level 3.
LabVIEW can be used to develop robust, re-usable software.

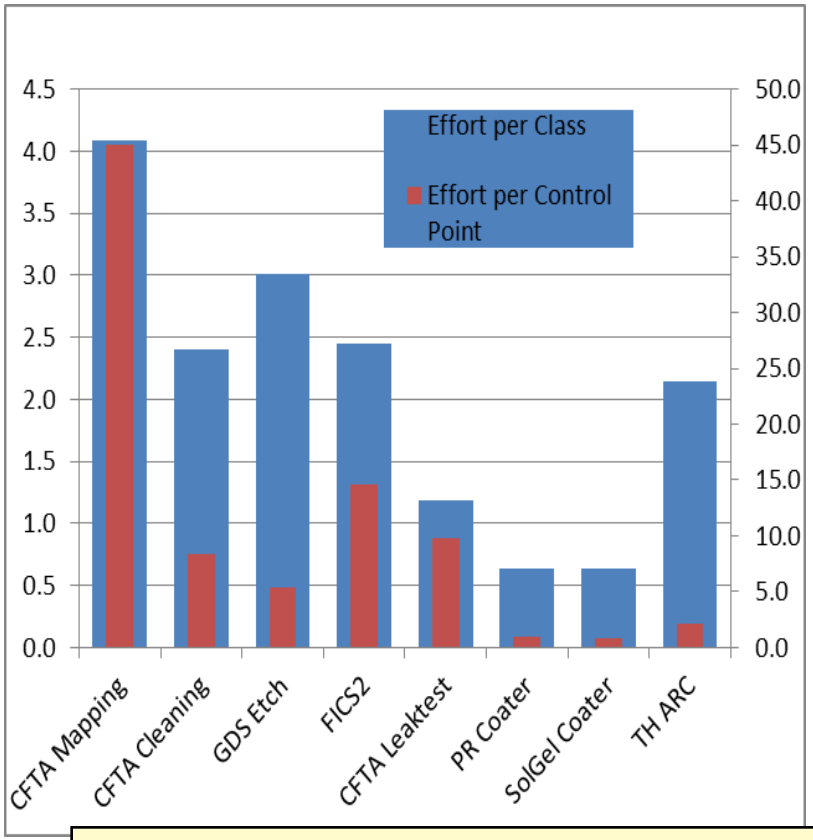
How did we do this?

- Formed and Trained the team:
  - Object Oriented Design & Programming
  - Configuration Management
    - Change management (Jira), Source Code Control (AccuRev)
    - TUCOBAB03: "Utilizing Atlassian JIRA for Large-Scale Software Development Management"
- Performed Software Engineering
  - Software project planning
    - tasks, estimates, schedules, communication, requirements management
  - Requirements Analysis
  - Code Reviews
  - Independent Testing
- Designed for reuse
  - Focus on system design, with reuse in mind
  - Abstractions and Components refactored into Common when needed and/or mature for reuse
- Implemented in LabVIEW

The team is performing and we are reaping rewards.

How well are we doing?

- Each system builds on improvements from earlier systems.
- The cost to build each system is trending downward.



		Effort (d) Total	Effort (d) per Class	Effort (d) per Control Point	Control Points (Devices & Controllers)
1	CFTA Mapping	585	4.1	45.0	13
2	CFTA Cleaning	199	2.4	8.3	24
3	GDS Etch	486	3.0	5.3	91
4	FICS2	321	2.5	14.6	22
5	CFTA Leaktest	98	1.2	9.8	10
6	PR Coater	109	0.6	0.9	119
7	SolGel Coater	109	0.6	0.9	123
8	TH ARC	249	2.1	2.1	120

‘These are some of the most stable systems we [customers] have seen.’ National Instruments is taking a keen interest in what we are doing.

What next?

- Continually improve
  - Agile development
  - Framework packaging
  - Encourage developers to enhance their skills
    - Training & Certification
- Rapid Prototyping
  - Some customers need applications running ‘today’
  - Support fast prototyping
- Communication
  - Advertise and interact with the community


NIF