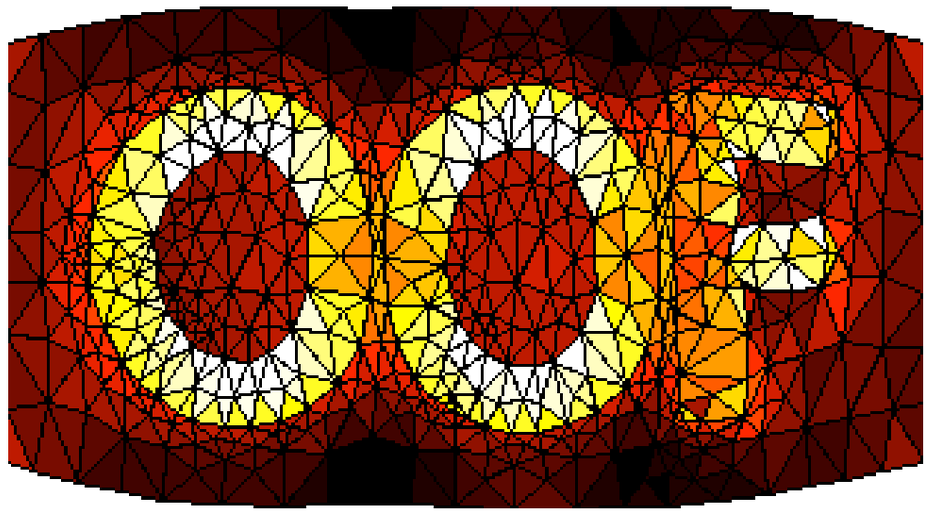


# Object Oriented Finite Element Analysis for Materials Scientists


| | |
|---|---|
| Stephen Langer | NIST ITL |
| Andrew Reid | Drexel (NIST ITL)* |
| Seung-Ill Haan | UMBC (NIST MSEL)* |
| Edwin Garcia | Penn State (NSF-ITR)* |
| Edwin Fuller | NIST MSEL |
| W. Craig Carter | MIT |
| Panos Charalambides | UMBC |
| Zi-Kui Liu | Penn State |

*NIST Center for Theoretical and Computational Materials Science

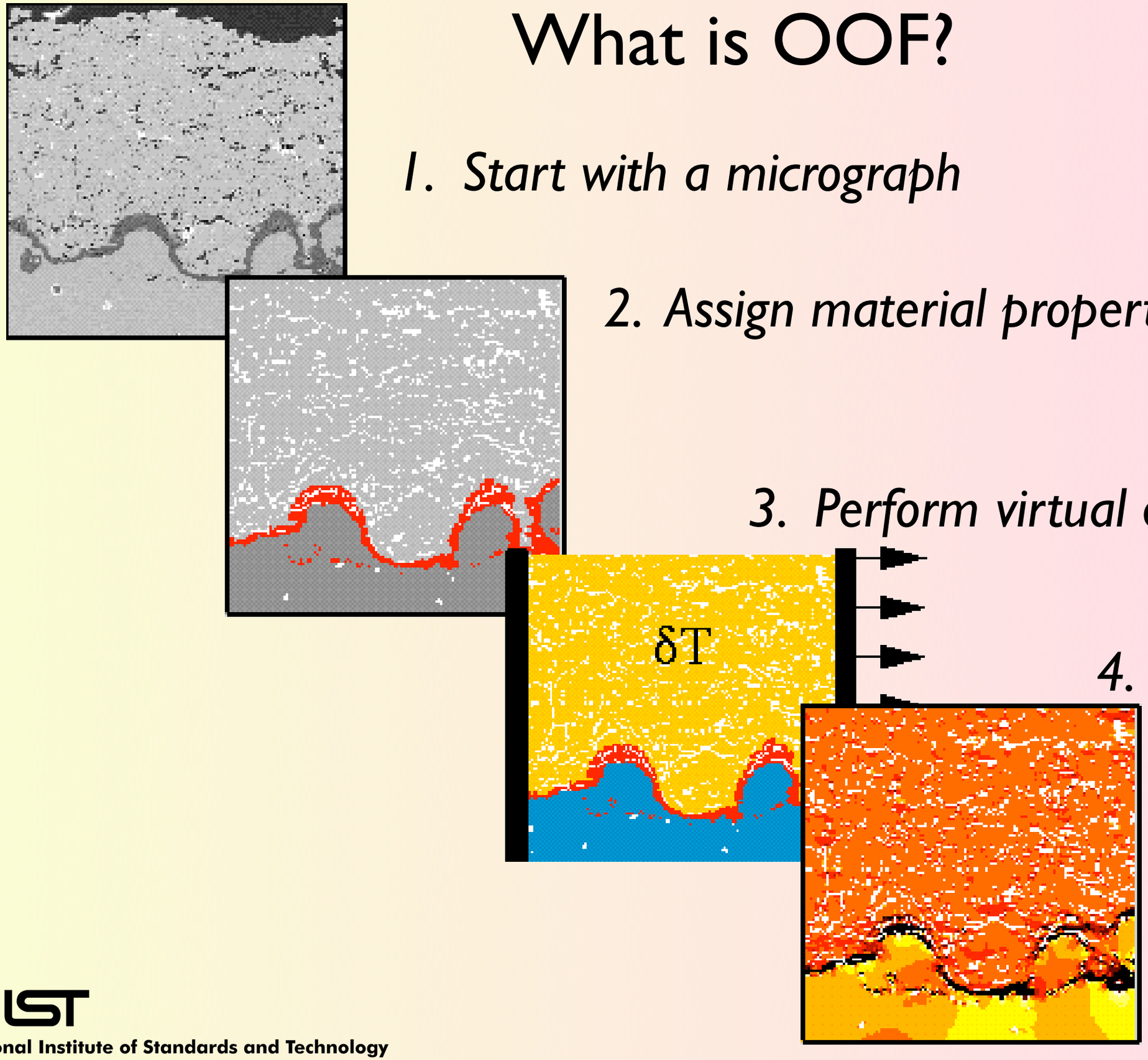


NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

# Outline of the Talk

- What?
- Examples:
  - Thermal Barrier Coatings
  - MatCASE
- OOF2
  - Why?
  - How?
    - Design Goals
    - Ingredients

# What is OOF?




*1. Start with a micrograph*

*2. Assign material properties*

*3. Perform virtual experiments*

*4. Visualize and quantify*

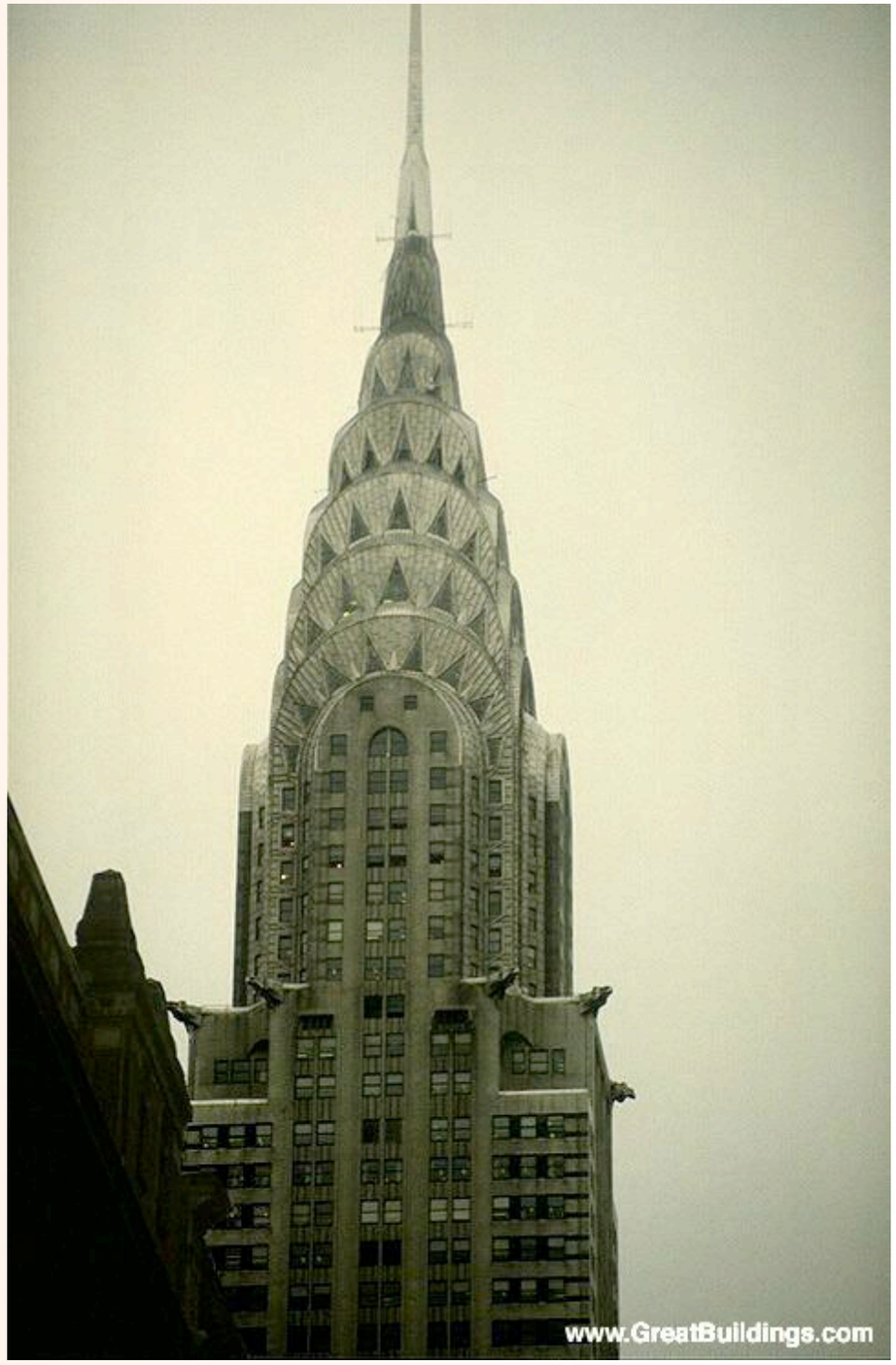
# Why OOF?

Goals:

- Elucidate the role of *heterogeneous*, *stochastic* microstructures on the bulk physical properties of materials.
- Correlate properties to microstructure
  - to shorten the materials development cycle.
  - to improve materials and processing.
  - to enable more reliable design.
- Provide an *extensible* platform for computing *general* physics of microstructures.
- A "Research Problem Solving Environment" for materials scientists.

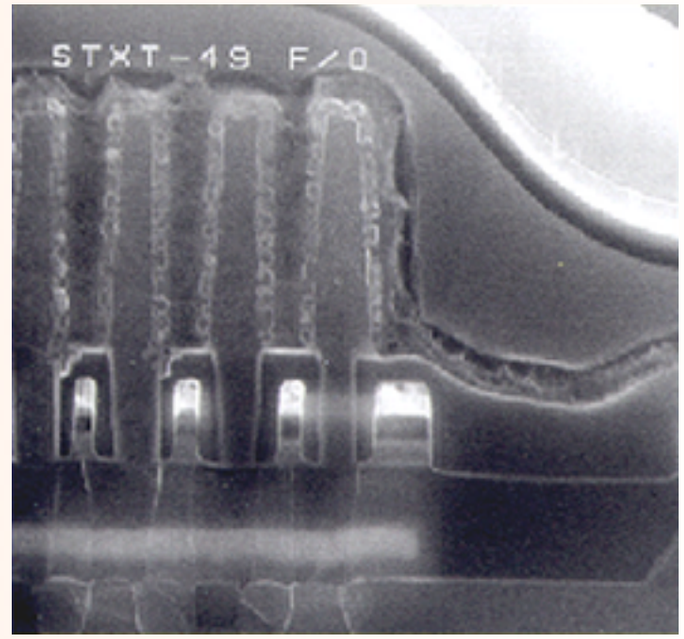
# Why OOF?

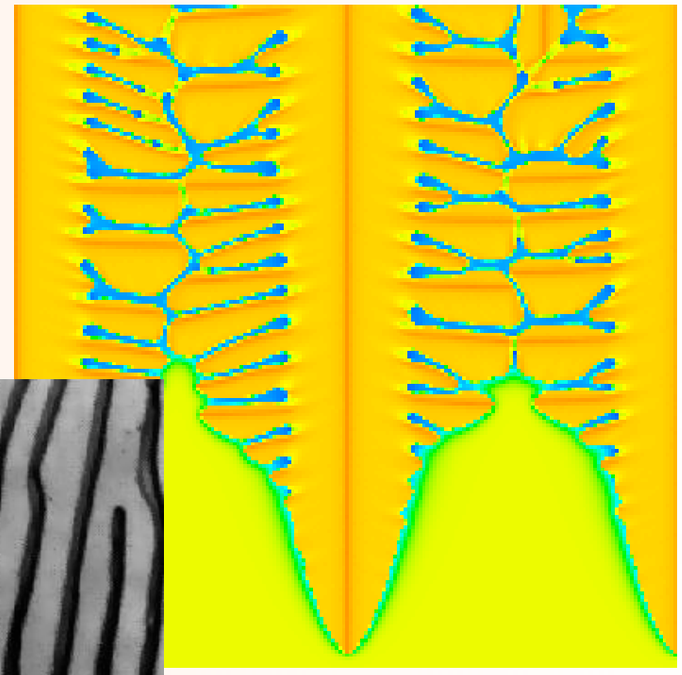
- Commercial finite element packages work best with large scale systems with regularly shaped components.

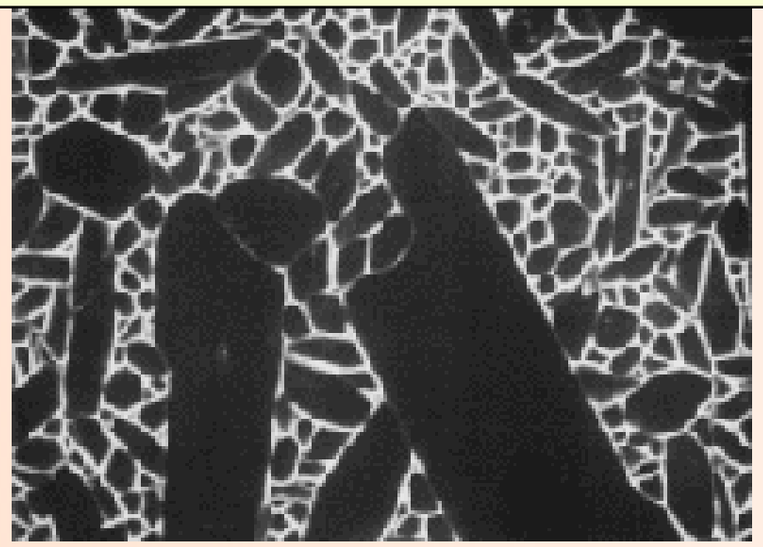


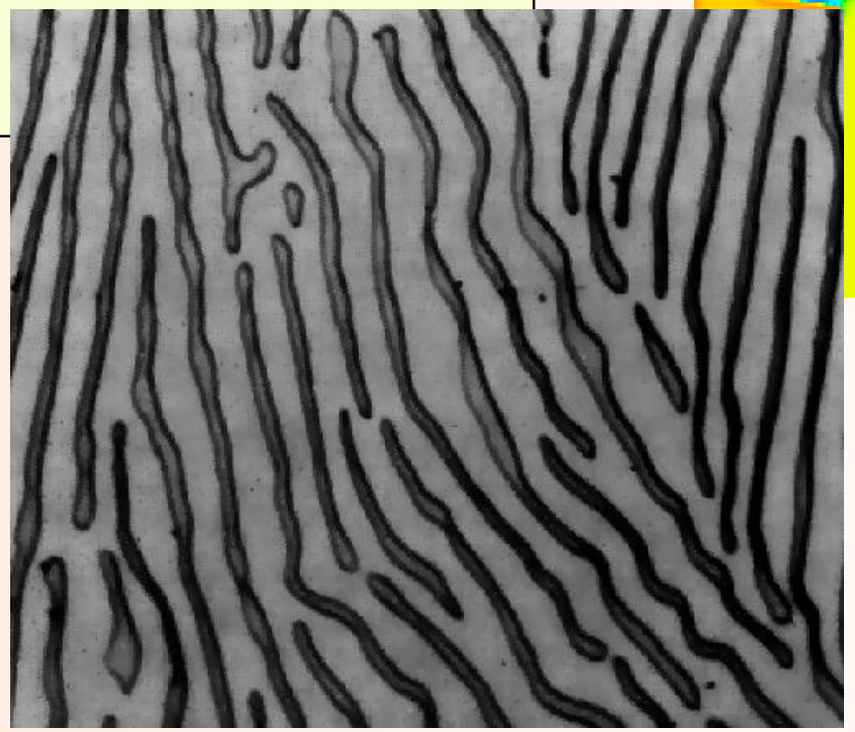
# Why OOF?

- Commercial finite element packages work best with large scale systems with regularly shaped components.
- Materials systems are small scale and disordered.



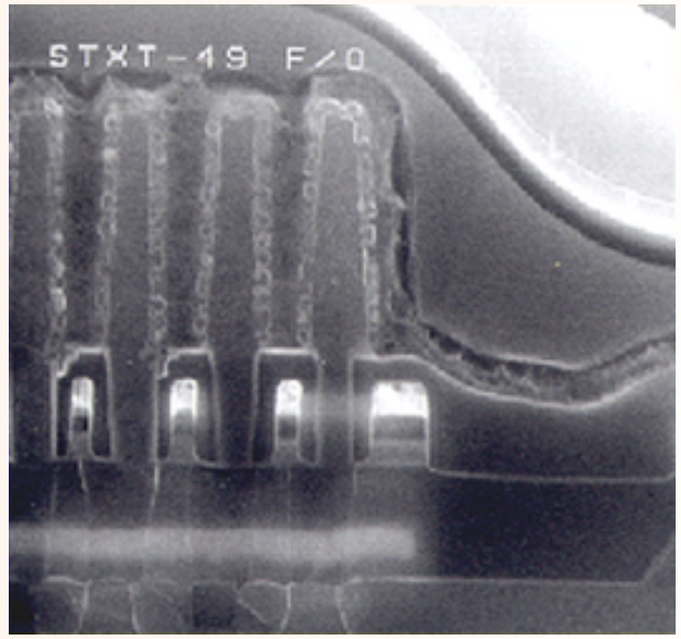



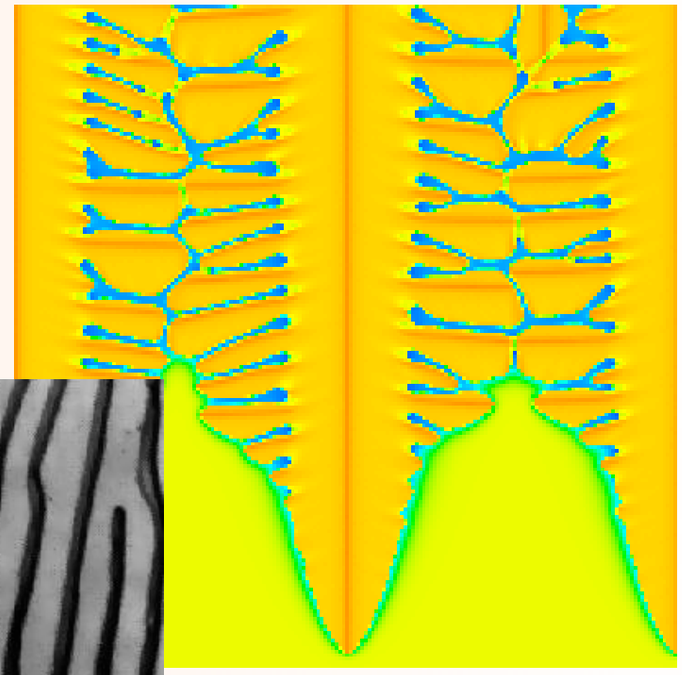


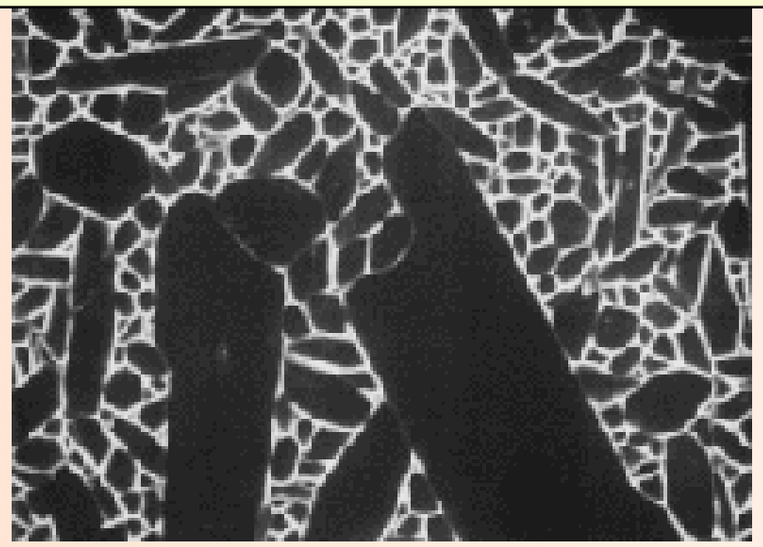


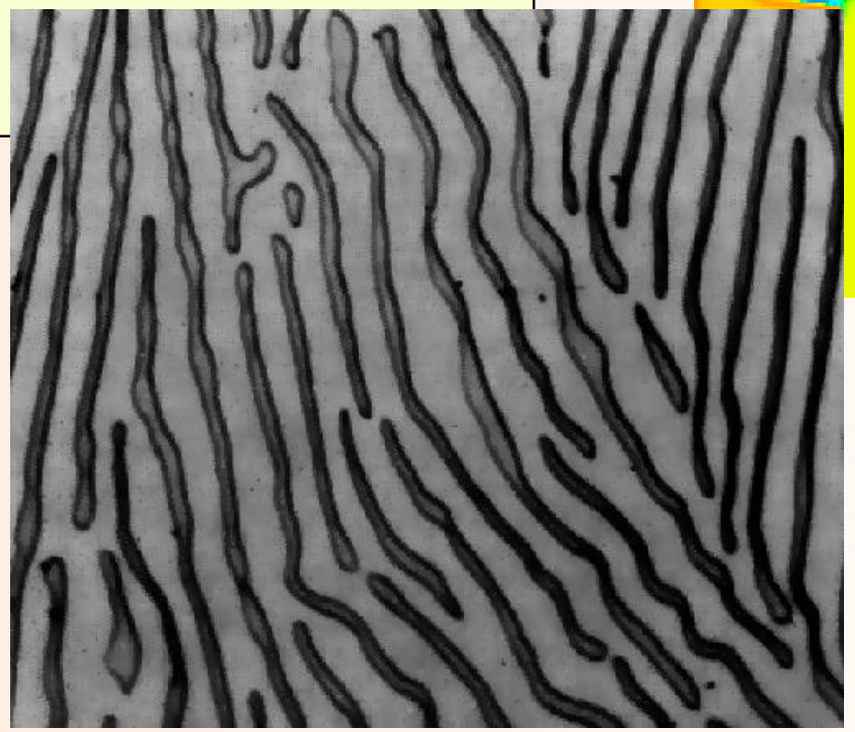
# Why OOF?

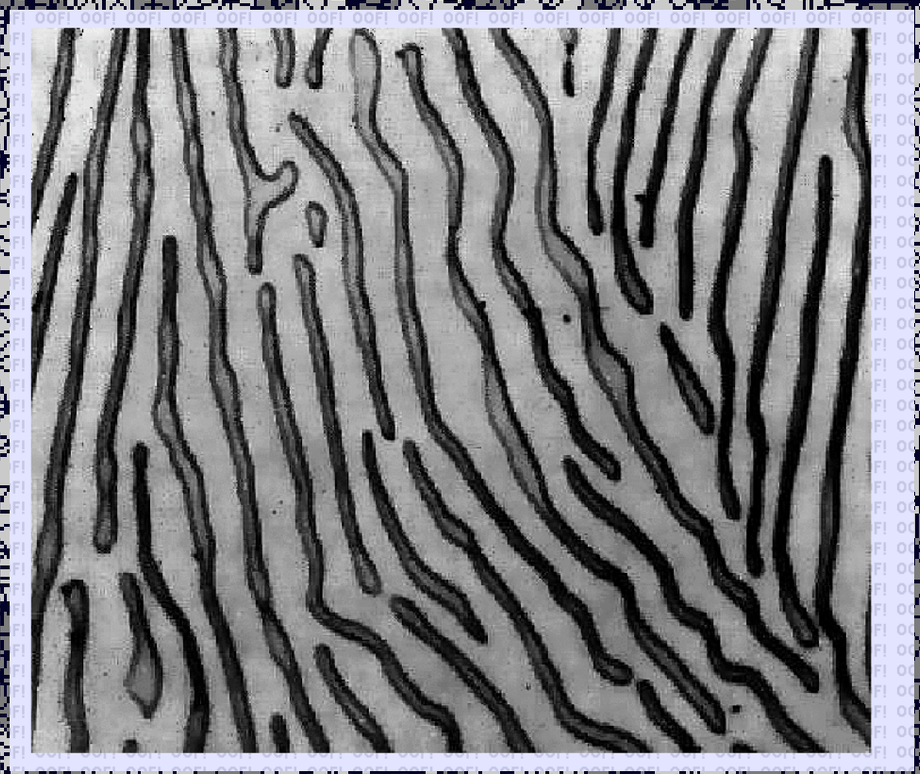
- Commercial finite element packages work best with large scale systems with regularly shaped components.
- Materials systems are small scale and disordered.
- OOF is designed to answer the questions that materials scientists want to ask.
- OOF is easy to use.







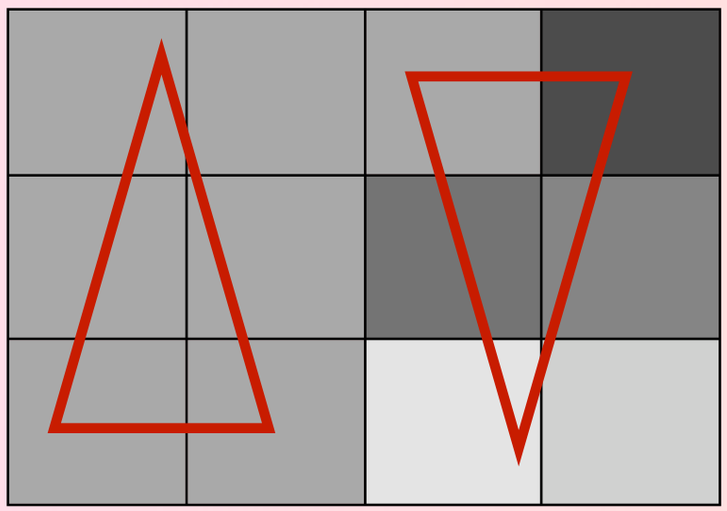


# Fitting a Mesh to Pixel Boundaries

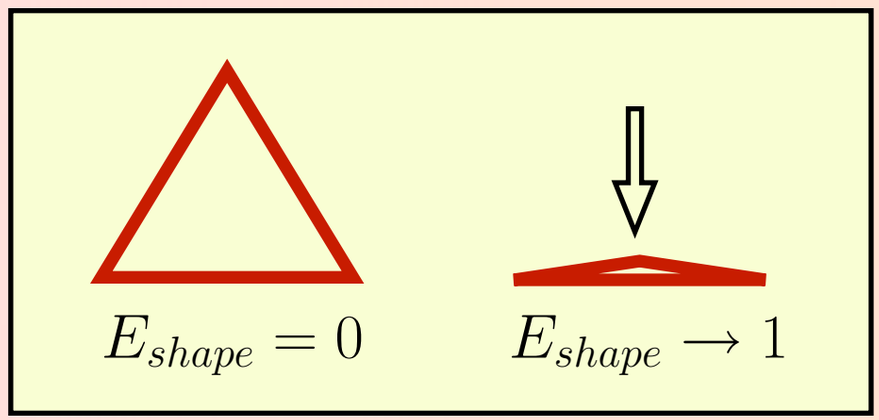
- Mesh operations work to minimize an "energy" functional $E$ of the mesh.

$$E = (1-\alpha)E_{shape} + \alpha E_{homogeneity}$$

$$E_{shape} = 1 - \frac{36}{\sqrt{3}}\frac{A}{L^2}$$



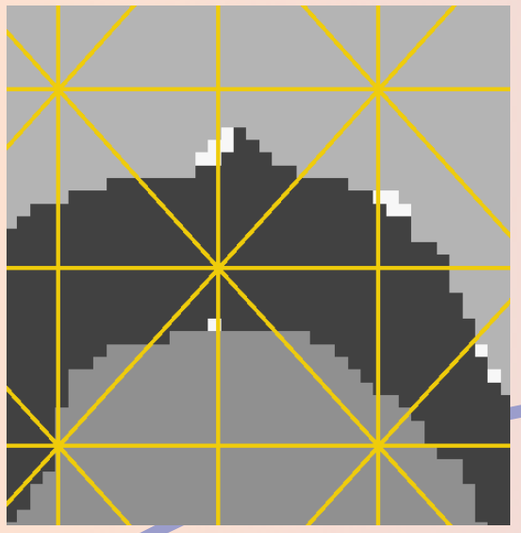
$E_{homog.} = 0$ $\quad$ $E_{homog.} \to 1$



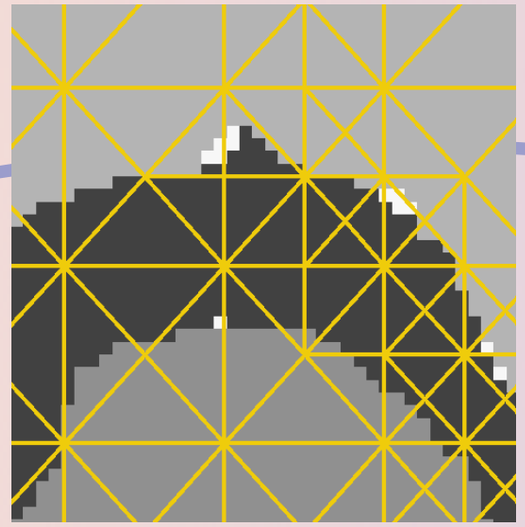

$$E_{homog.} = \prod_{i=1}^{N} [(1-a_i)/(1-1/N)]$$

$a_i$= fractional area of pixel type $i$
$N$ = total number of pixel types

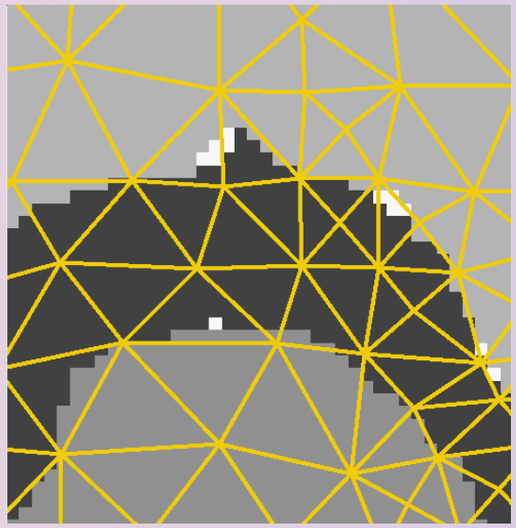
# Fitting a Mesh to Pixel Boundaries



Initial Coarse Mesh

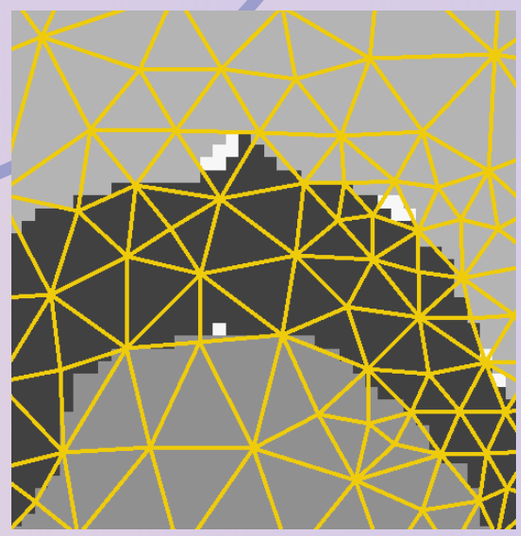


Refining Elements



Monte Carlo Node Motion
and Edge Swapping

Operations to
reduce $E$



Final Mesh

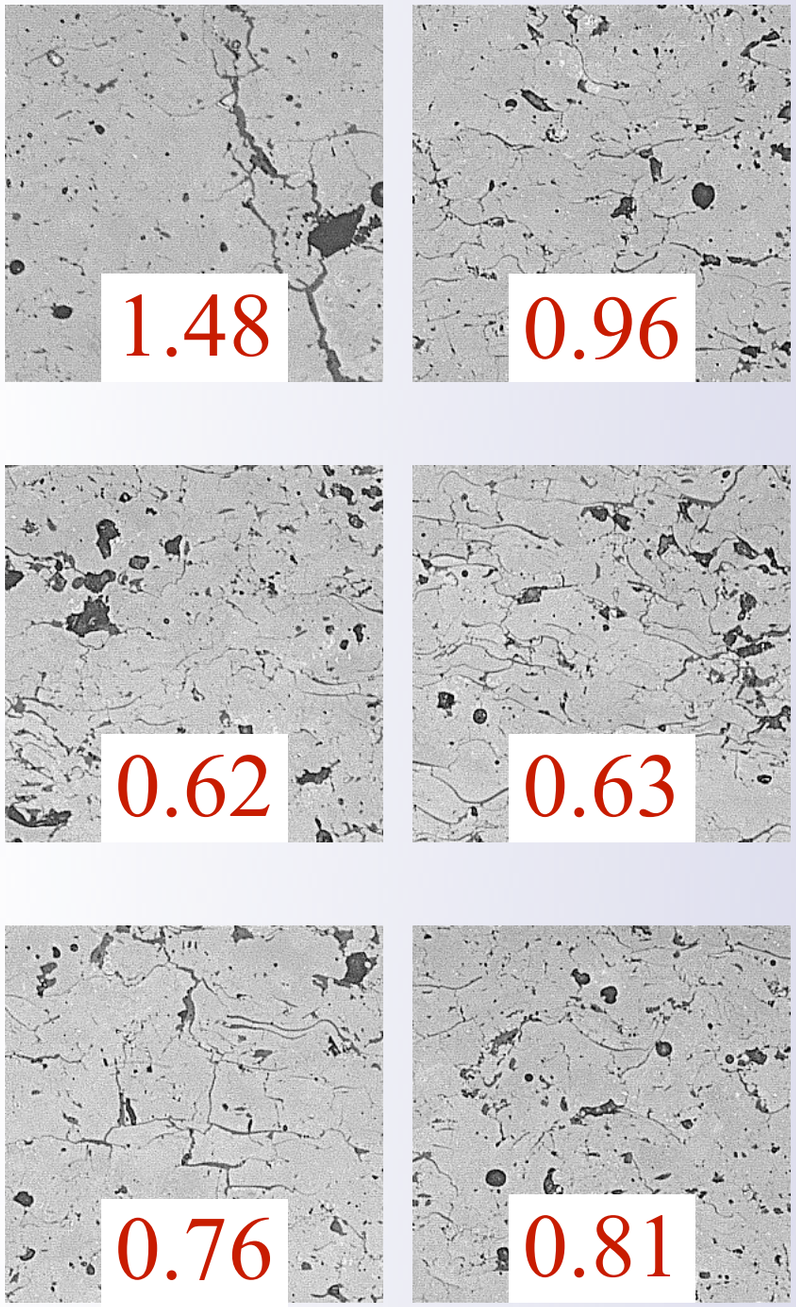
- Examples:
  - Thermal Barrier Coatings
  - MatCASE:
    Materials Computation and Simulation Environment

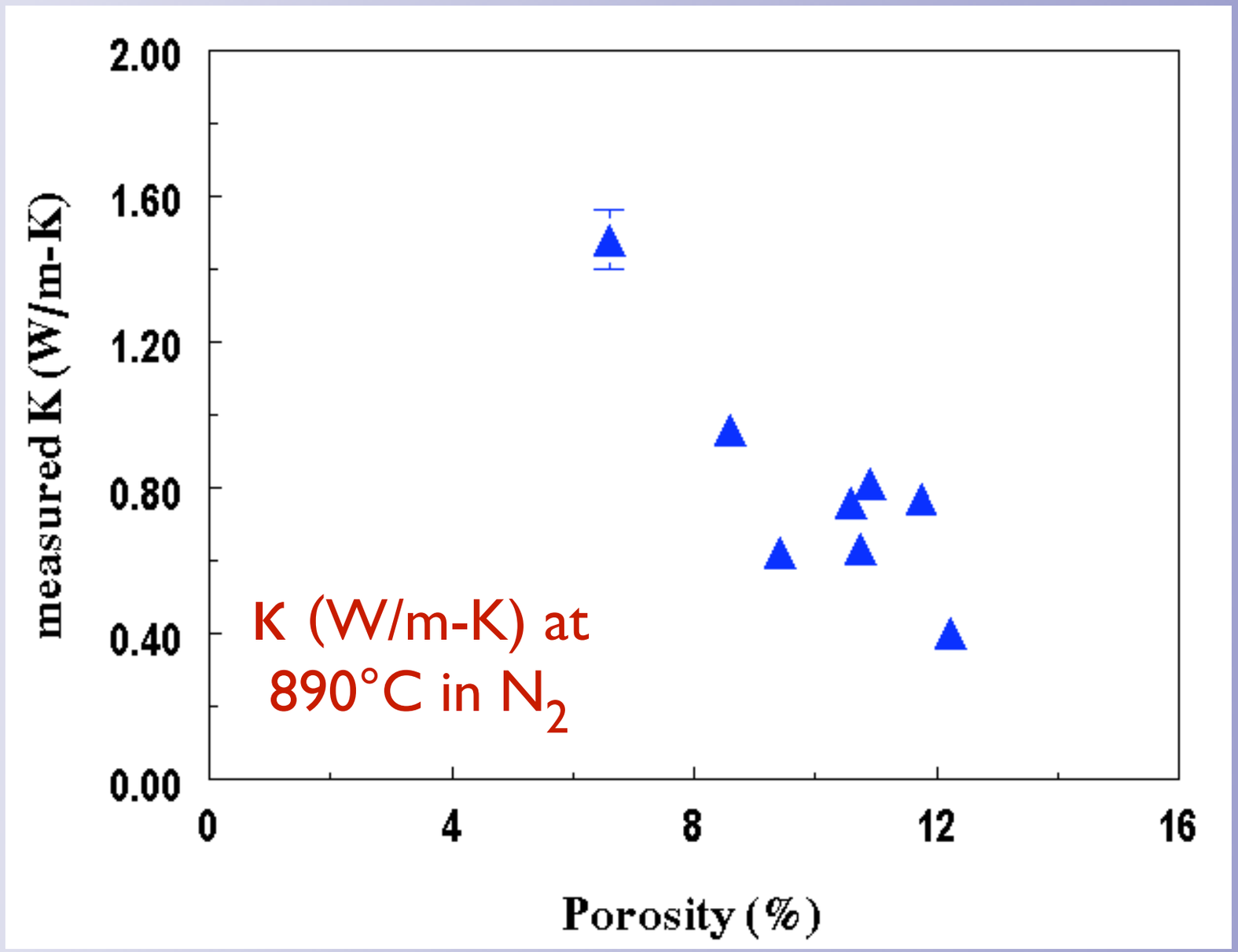
# Predict Thermal Conductivity κ of Ceramic Thermal Barrier Coatings for Turbine Blades

with James Ruud, NS Hari, James Grande, and Antonio Mogro-Campero,
GE Corporate R&D
Funded in part by DOE Advanced Turbine Systems Program

- TBC's allow jet engine blades to operate at higher temperatures.
- Physical measurements of κ are difficult, time consuming and expensive. Hardly ever done during quality control.
- OOF could replace measurements during research, development, design, and production.

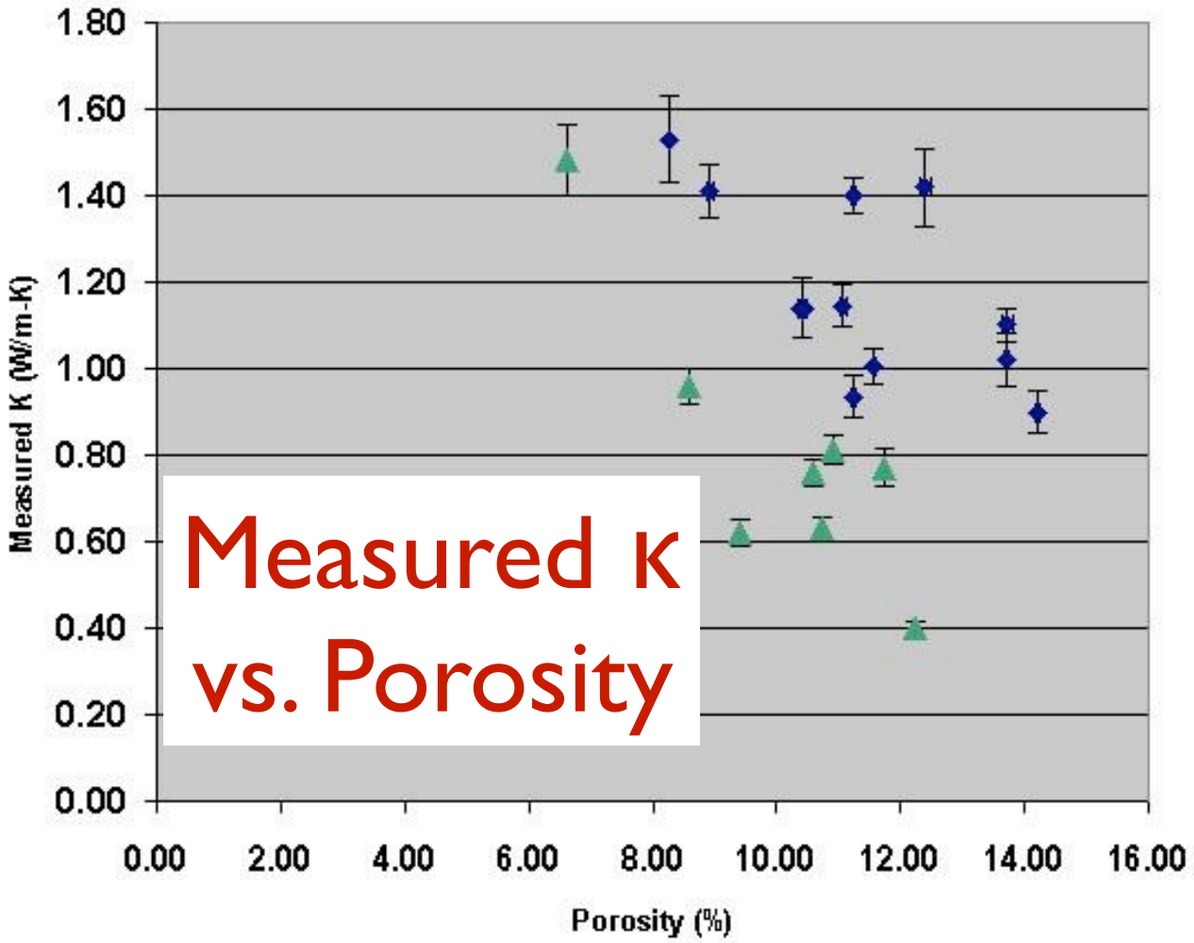
# Thermal Conductivity Measurements

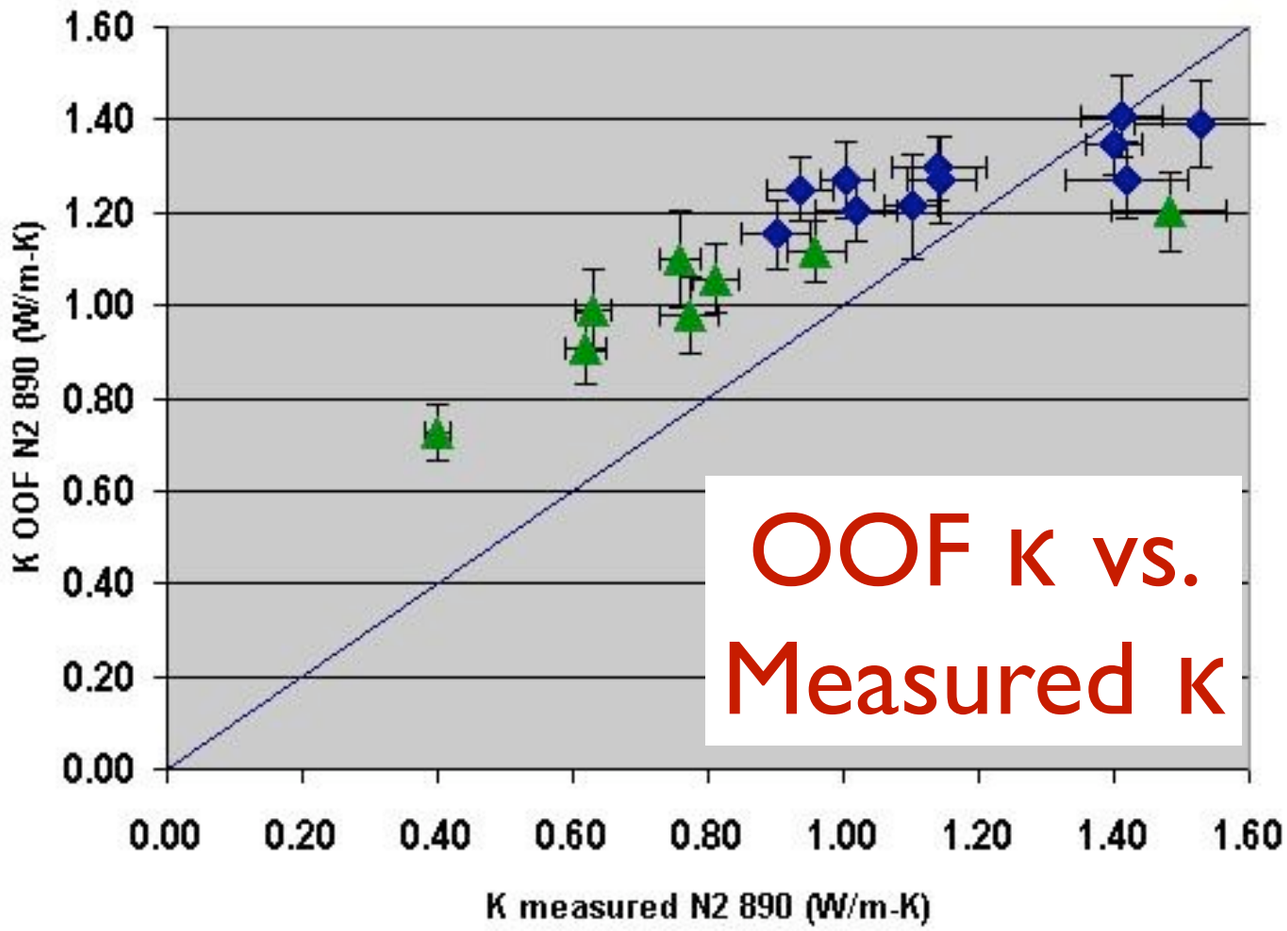



Porosity is not a good predictor of thermal conductivity.

# Comparison of Two Specimen Sets



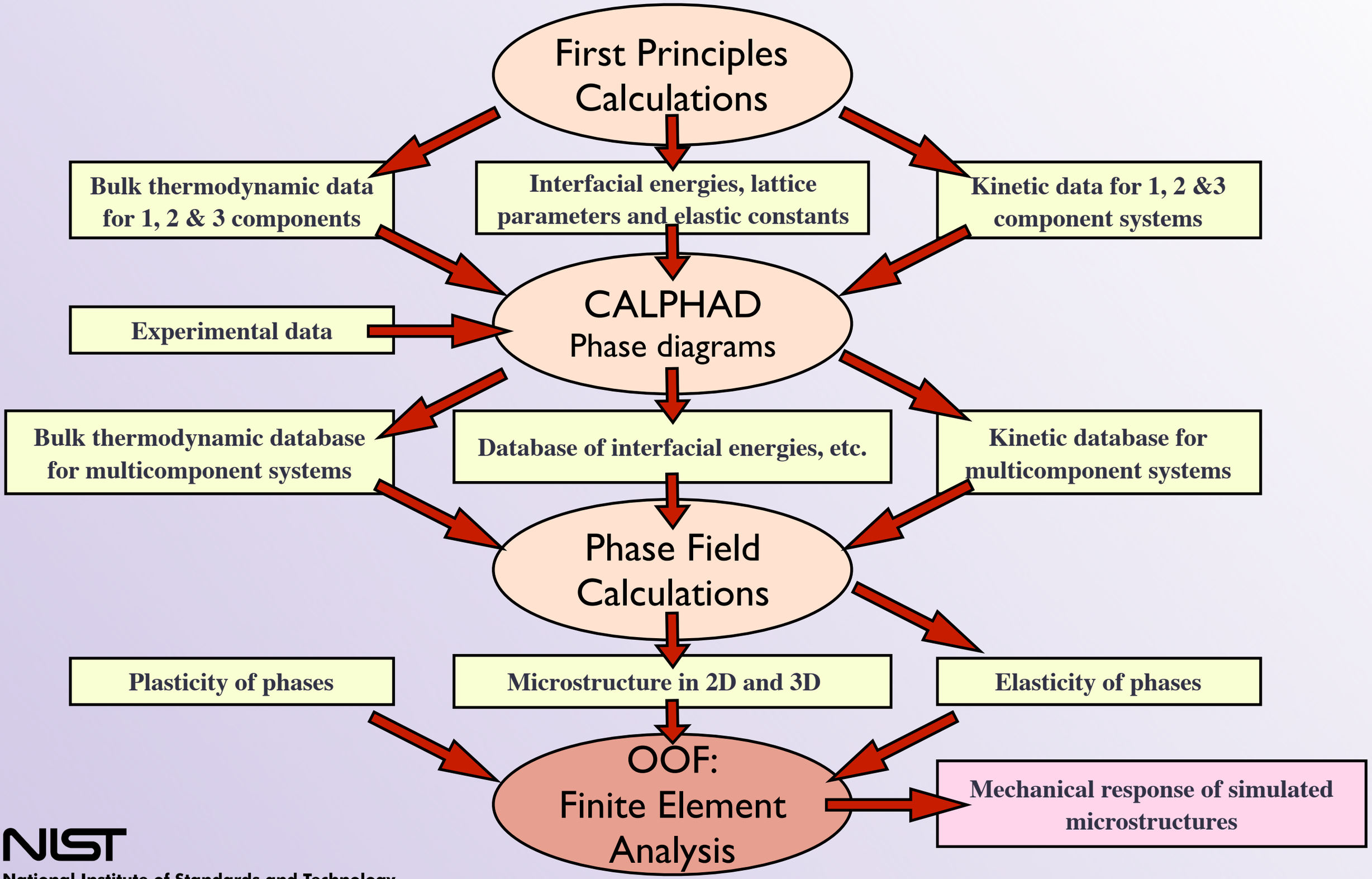




- Consistent correlation for a wide range of microstructures.
- OOF is a good predictor of thermal conductivity.

# The MatCASE Project

- Materials Computation and Simulation Environment
- Funded by NSF-ITR.
- Includes Materials Scientists, Physicists, Mathematicians and Computer Scientists.
  - Participants are from Penn State, Ford, and NIST.
  - Zi-Kui Liu, Long-Qing Chen, Padma Raghavan, Jorge Sofo, Chris Wolverton, Qiang Du, SAL.
- Goal: To develop integrated computational tools for multiscale materials design.
- OOF2 will be one part of a parallel, grid-enabled, computing environment.

# Integration of Four Computational Methodologies



# Why OOF2?

OOF2 reflects lessons learned from OOF1.

- ☐ More expandable.
- ☐ More flexible.

Emphases:

- ☑ *Extensibility* and *maintainability* through proper object-oriented design reflecting the underlying mathematics.
- ☐ *Generality* by making few assumptions about the problems being solved.
- ☐ *Usability* with a clear user interface.
- ☑ *Sanity* with a flexible infrastructure.

# OOF2

- Easily extendible to a wide variety of problems
  - elasticity, plasticity, thermal conductivity, mass diffusion, electrical polarization, piezoelectricity, ferroelectricity, Darcy's Law fluid flow, …

$$\sigma = \sum_i k_i \nabla \phi_i \qquad -\nabla \cdot \sigma = f$$

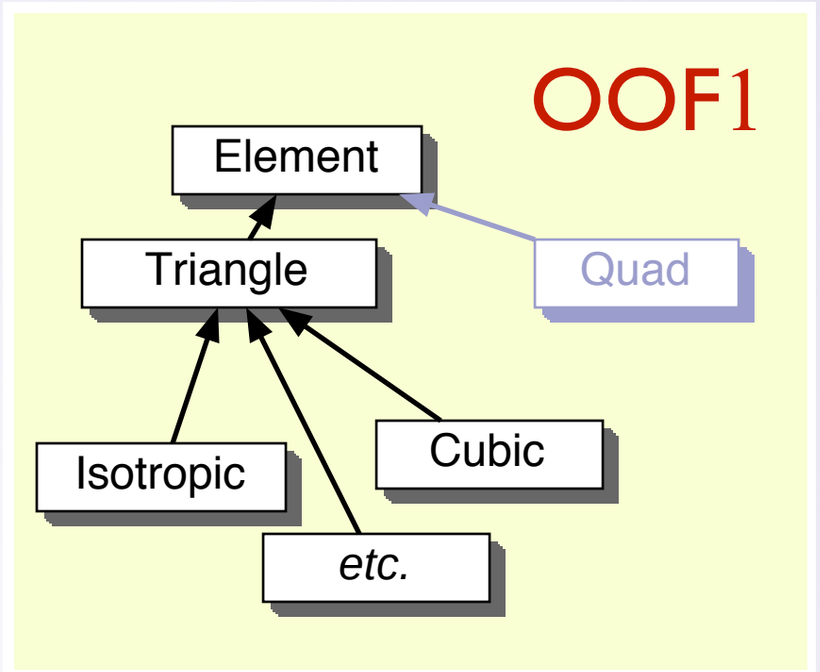
SCHEMATIC

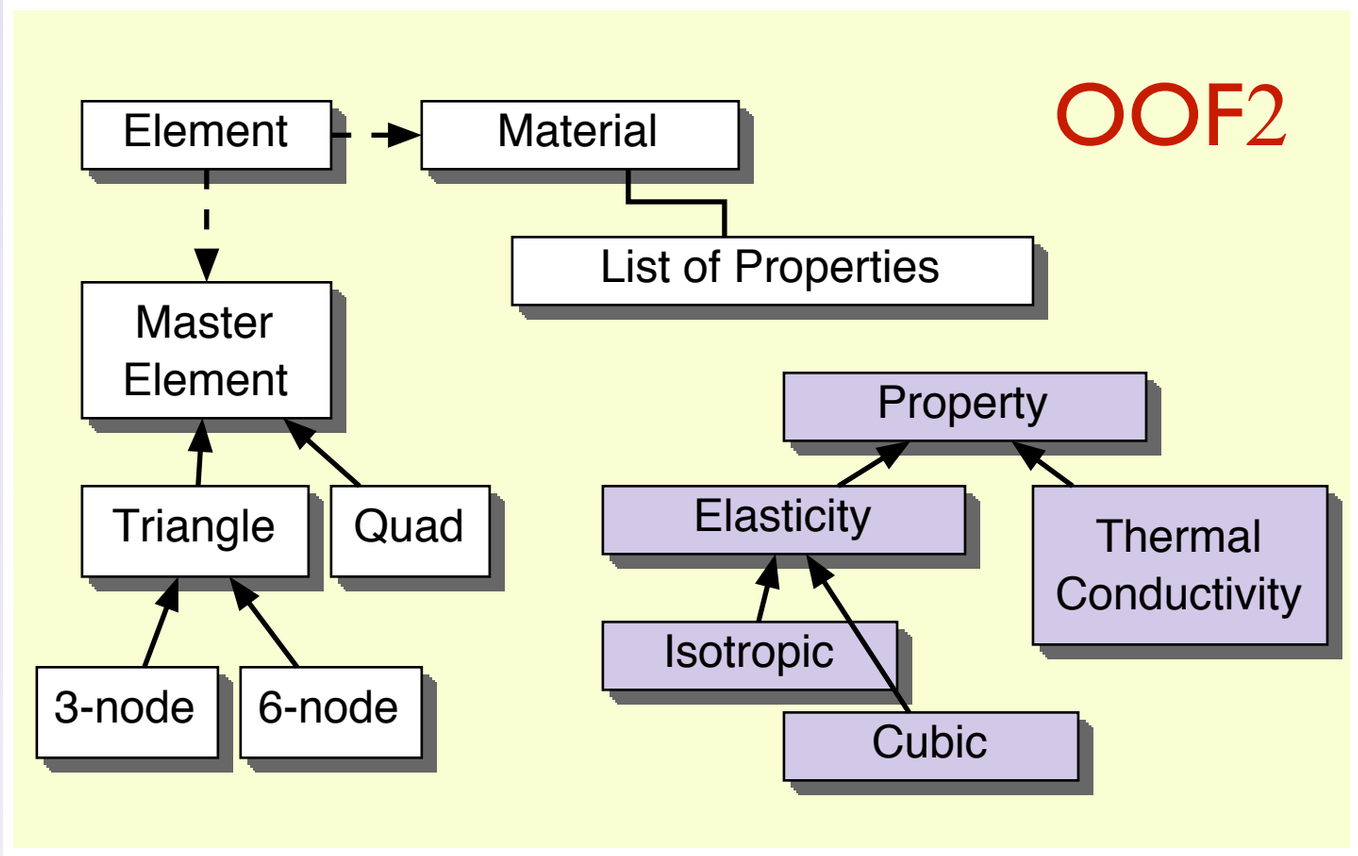
| | Elasticity | Thermal Cond. | ? |
|---|---|---|---|
| Field Φ | displacement | temperature | ? |
| Flux σ | force | heat flow | ? |
| Modulus *k* | $C_{ijkl}$ | $\kappa_{ij}$ | ? |
| Force *f* | force | heat source, sink | ? |

- Designed for simple addition of new fields, fluxes, and equations.

# Why OOF2?

- For example (proper design):
  - Physics and Finite Element class structure more closely tied to the underlying mathematics.
  - Allows more physics *and* more types of finite elements.



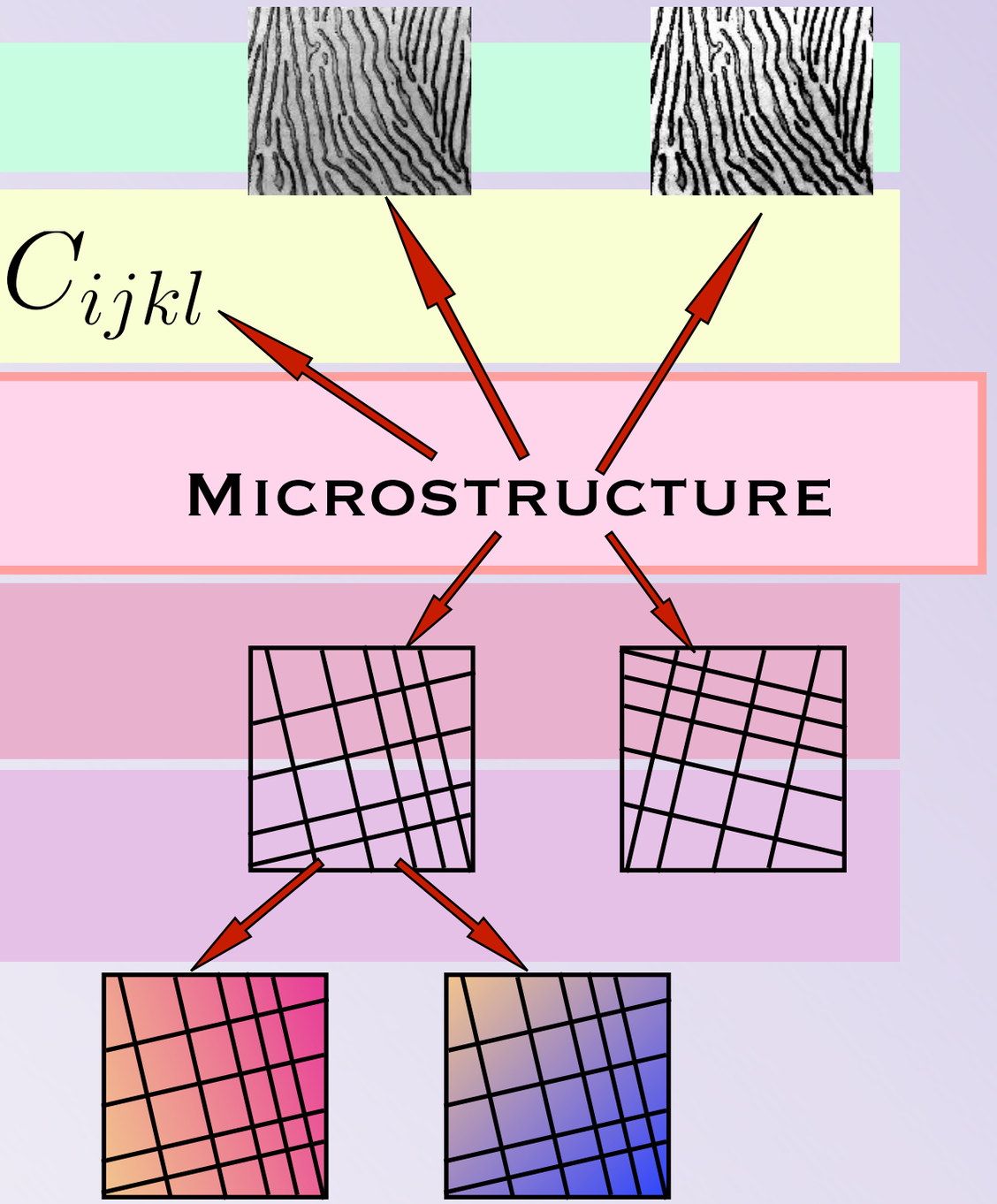

Properties can be coded completely independently from the element classes.

# OOF2 Code Ingredients

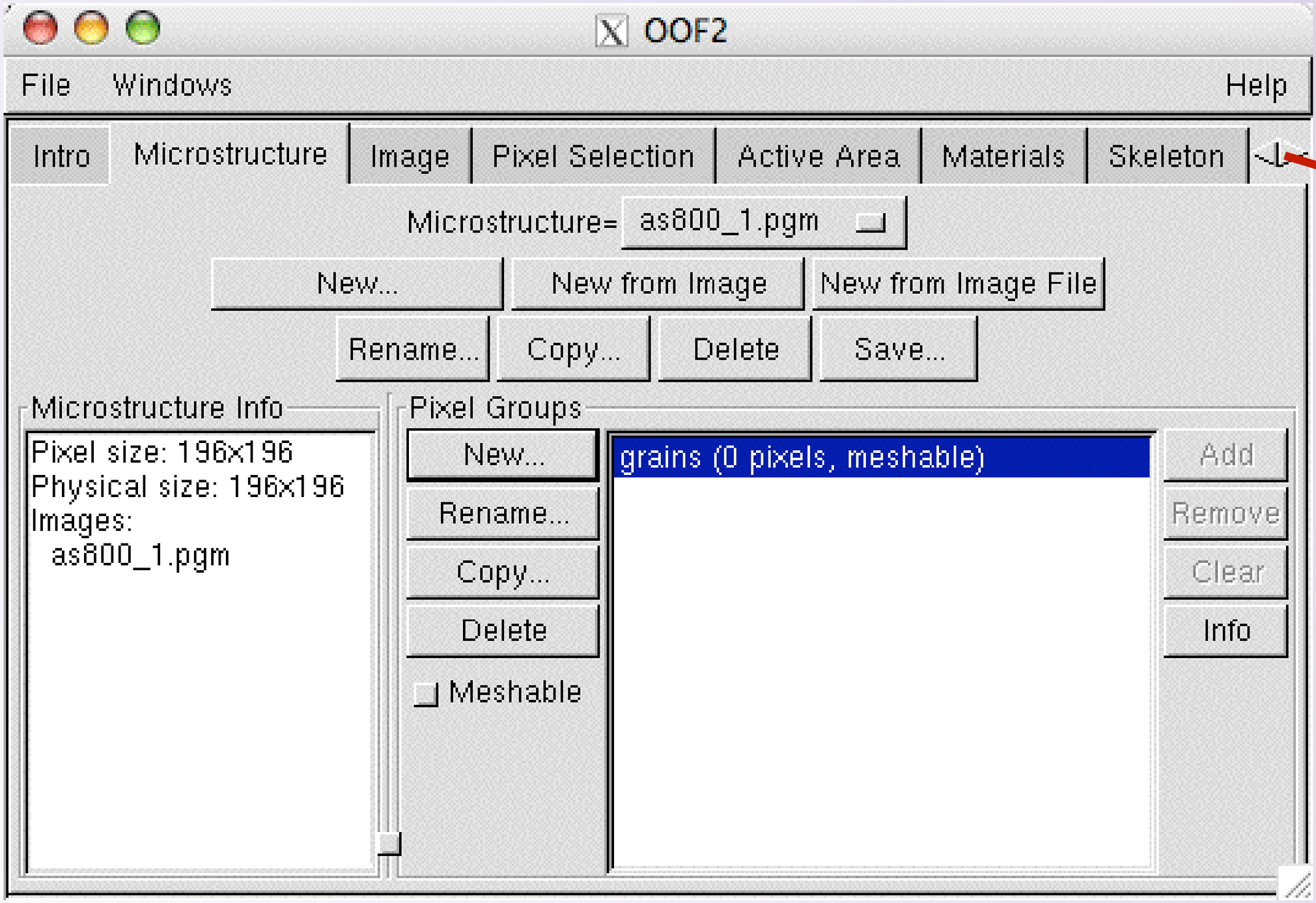
- C++ (core) and Python (interface).
- C++/Python glue code generated by SWIG.
- Libraries:
  - GTK+ graphics.
  - PETSc, MPI parallel solvers.
  - ImageMagick image manipulation.
  - IML++, MV++, SparseLib++ linear algebra.

# OOF2 Conceptual Ingredients

- image
- materials
  - assembled from lists of properties
- microstructure
  - materials assigned to groups of pixels
- skeleton
  - only the geometry of the finite element mesh
- mesh
  - skeleton + mathematics + physics
- solution

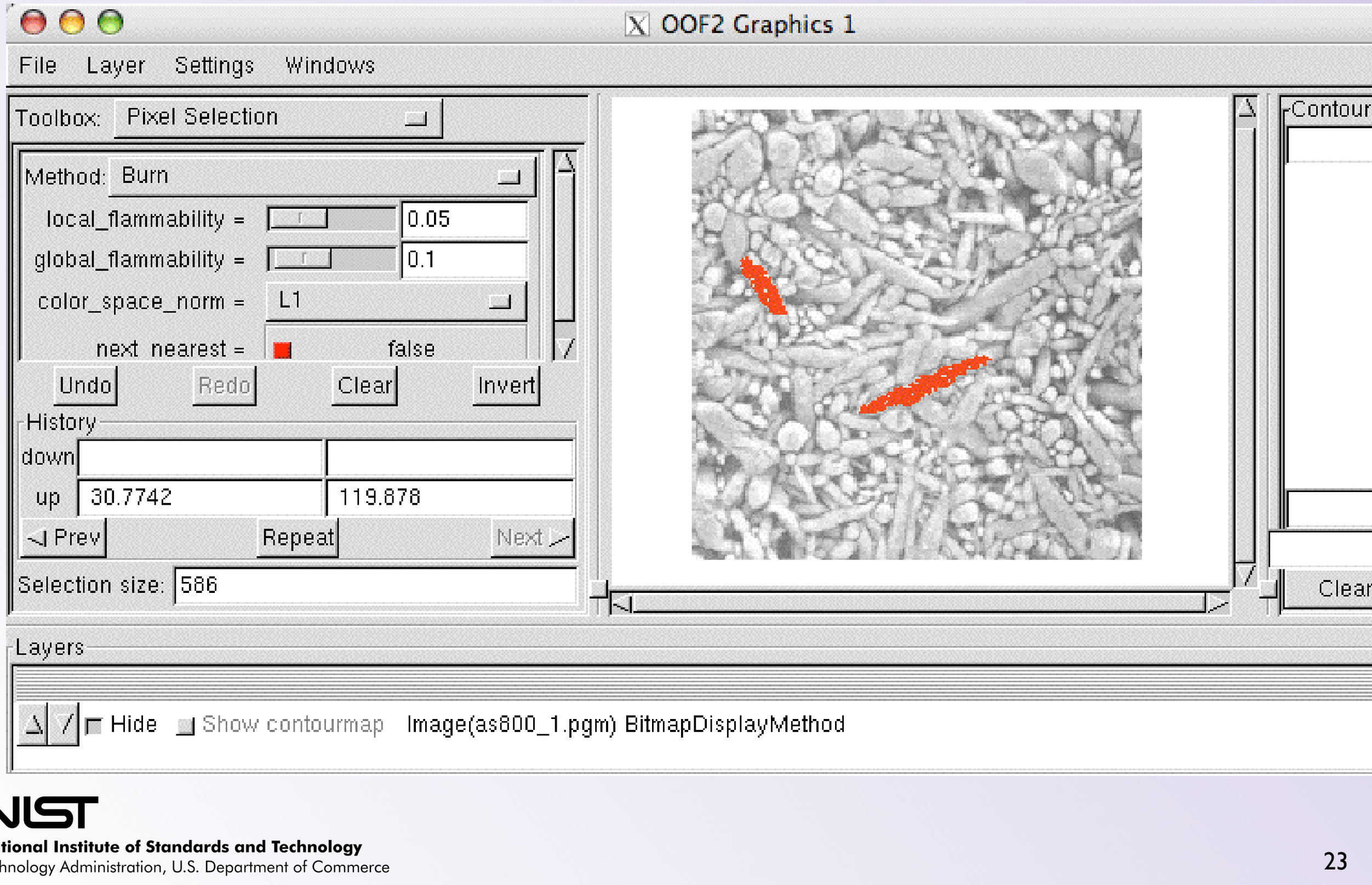


# Interface leads users through the tasks

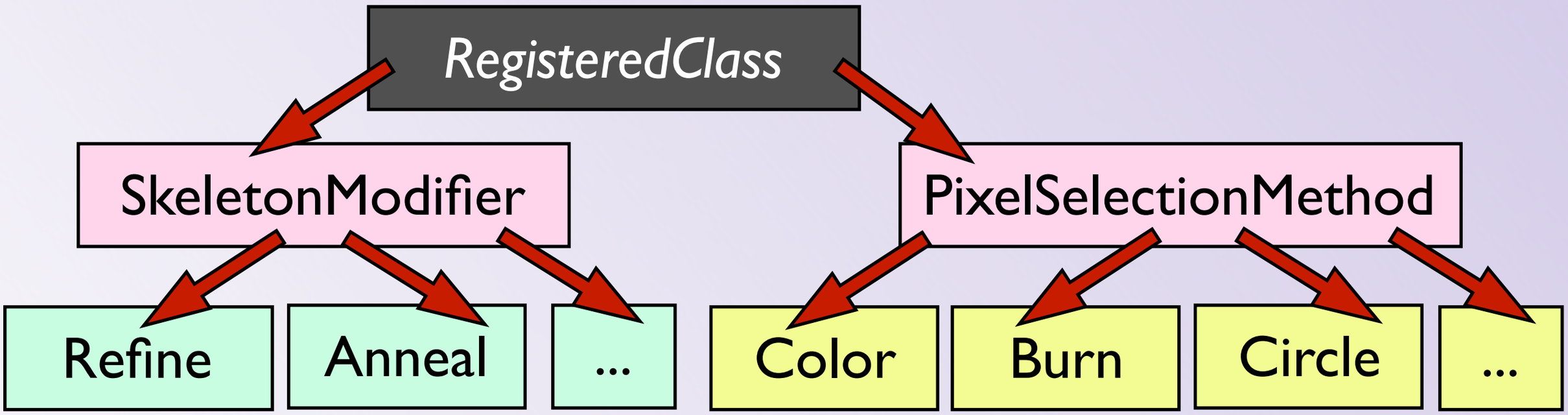



Skeleton Boundaries
FE Mesh
Fields
Equations
Boundary Conditions
Solver
Analysis
Boundary Analysis

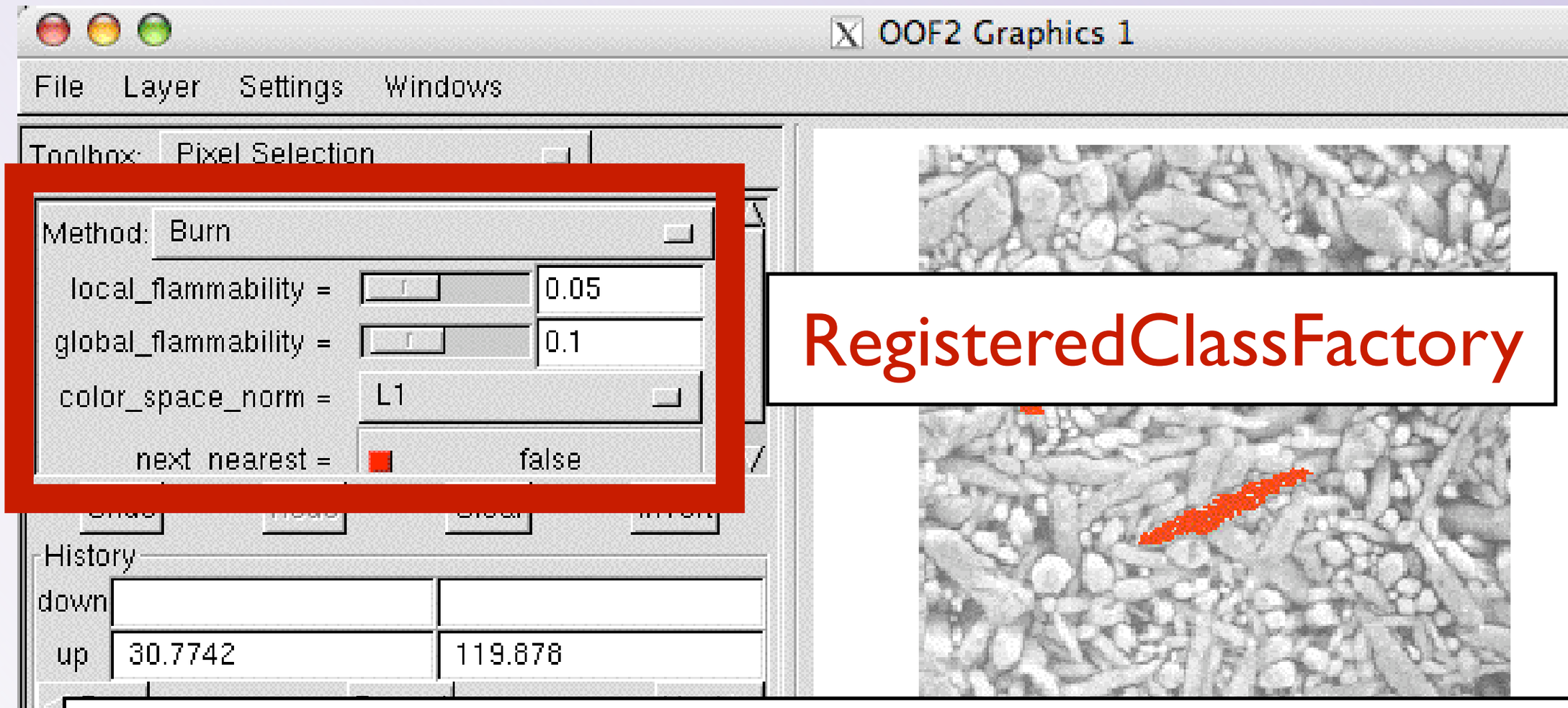
# Graphics Window



# Extensibility via Class Hierarchy




- Registered classes represent:
  - Operations on images, meshes, etc.
  - Material properties.
  - Parameters for the above.
- *Registrations* describe how to create objects in the classes
- Menus and GUI components are created *automatically* from Registrations.



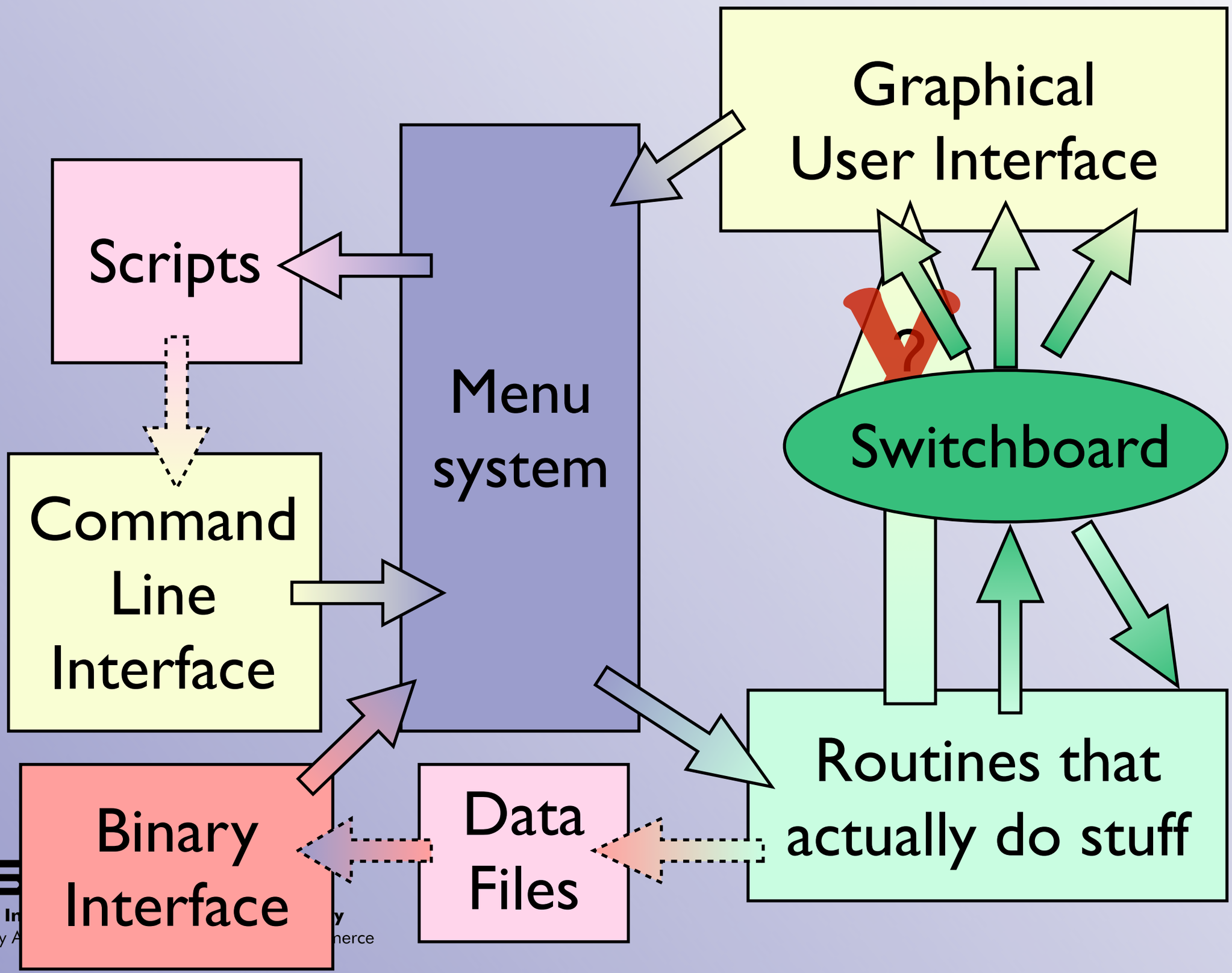

# RegisteredClassFactory

```
Registration('Burn', Burn, PixelSelectionMethod,
             params=[
                FloatRangeParameter('local_flammability',
                                    range=(0,1,0.01), value=0.1,
                                    tip="don't take any wooden nick
                FloatRangeParameter('global_flammability'...),
                EnumParameter('color_space_norm', ColorNorm...),
                BooleanParameter('next_nearest', ...)],
             tip="Select a contiguous set of pixels..."
             )
```

# More Infrastructure

- Underlying menu driven structure (in Python):
  - Specify name, callback function, menu, argument parameters.
  - Menu items created explicitly, or implicitly from Registrations.
- Communication between different code components is by means of a “switchboard”
  - Objects send messages to switchboard.
  - Other objects subscribe to messages.
  - Sending object doesn’t have to know who (if anybody) is listening.
  - Allows modular development and use.

# OOF2 Control Structure



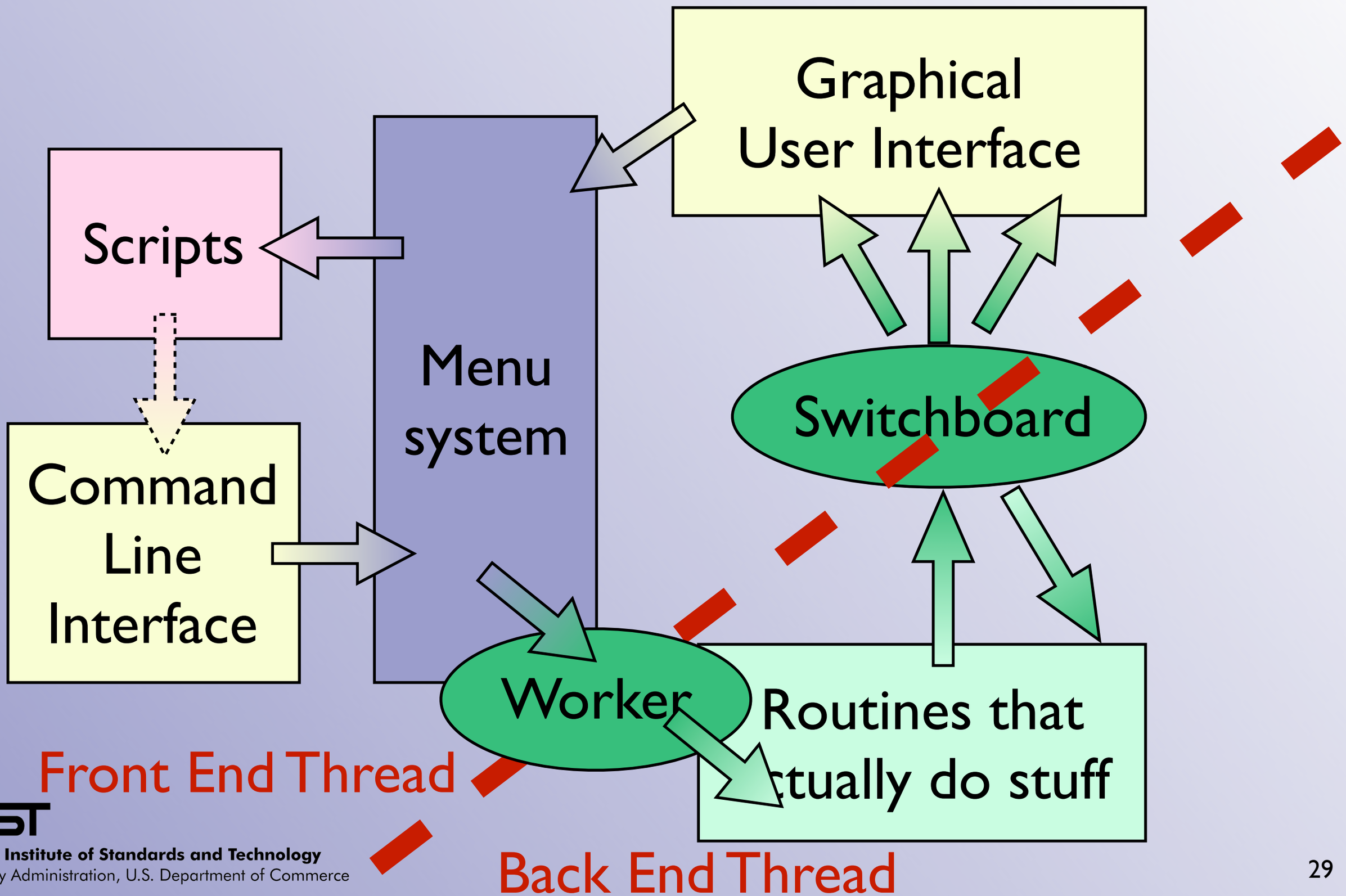
# GUI, Threading & Parallel Processing

- OOF is meant to be an interactive system in which users can experiment with different scenarios in real time.
  - Need a responsive multithreaded interface.
  - Parallel back-end for quick turnaround.

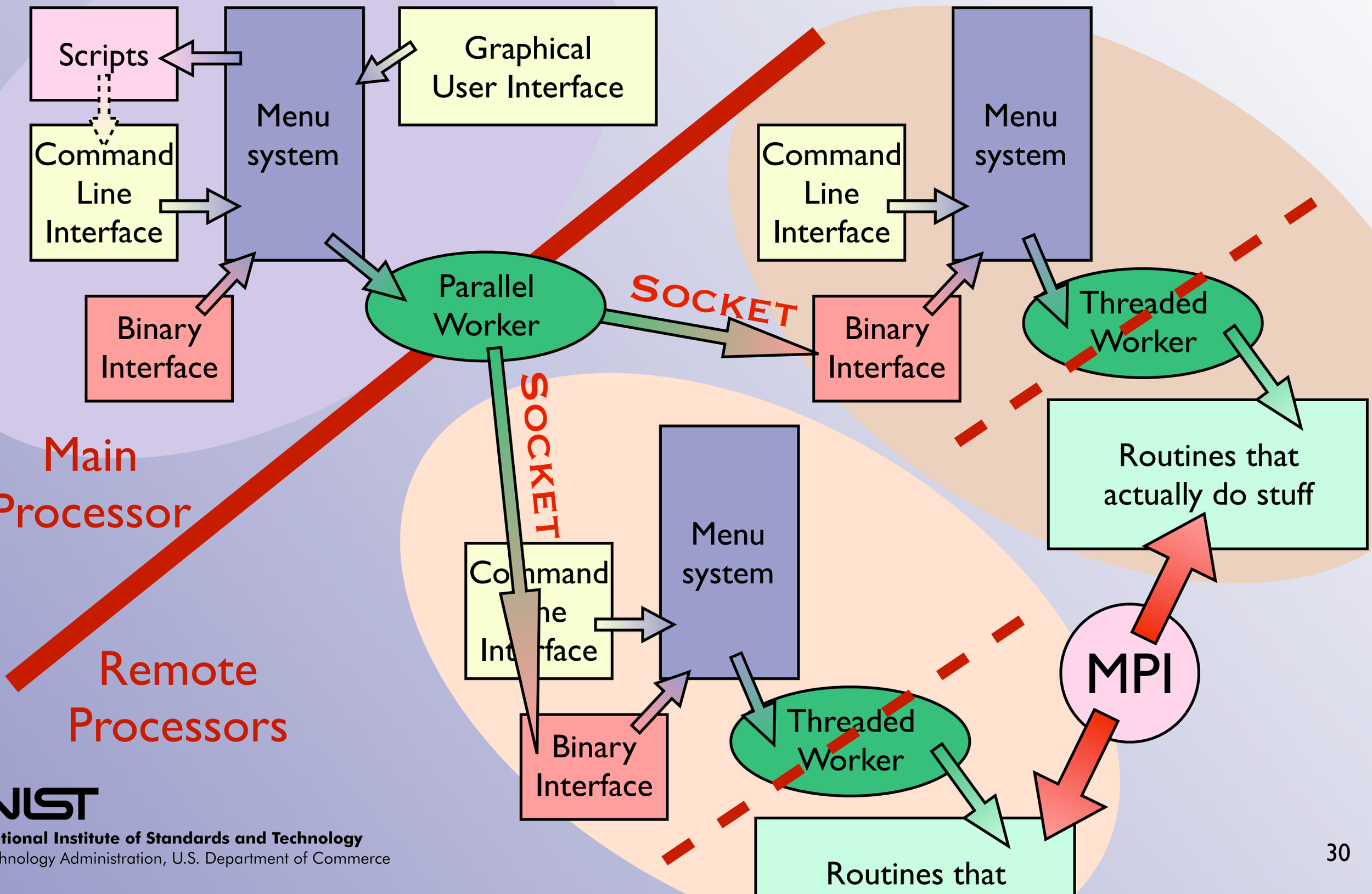
- Still, lengthy computations need to be performed in batch mode, without a GUI.

- "Worker" classes added to menu system to handle different modes of operation.
  - TextWorker, GUIWorker, ThreadedWorker, etc.

# OOF2 Control Structure



# OOF2 Control Structure



# What's Next?

- OOF2.x will add
  - non linear solvers
  - plasticity models
  - time dependent & eigenvalue problems
- OOF3D

- OOF1 is available now at
  **http://www.ctcms.nist.gov/oof/**
  - source code for Unix computers
  - precompiled binaries (Linux, SGI, Mac OSX)
  - manuals & tutorials
  - mailing list

- OOF2 will be available real soon now (within a few months, we hope).