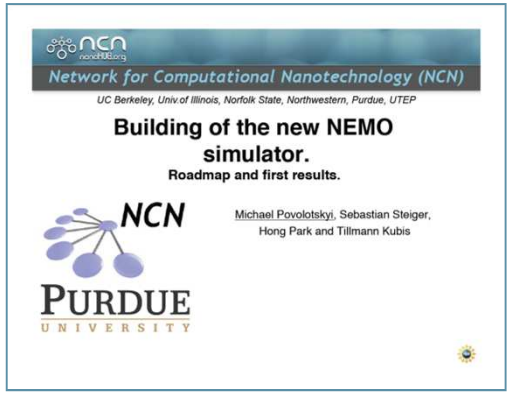
NEMO5 Server

PURDUE UNIVERSITY

Daniel Mejia, Santiago Perez

Nemo 5 Original Roadmap



Building of the new NEMO Simulator (2010). Michael Povolotsky, Sebastian Steiger, Hong Park, Tillmann Kubis

Objectives for new NEMO code:

- should include the functionality of the existing codes, e.g. *NEMO1D, NEMO3D, OMEN, OMEN3d_par,…*
- should replace nanohub tools, e.g. *1D Hetero, OMEN wire*

  - Modular, clearly written and easily maintainable, documented
  - Single code supported by many people

- Allows for expansions: new physics must be added easily

  - Must be user friendly: clear script/GUI, easily visualized output

PURDUE UNIVERSITY Michael Povolotskyi

Misconseptions about UI

Creating a user interface is a the last step

Other Users are not my problem

A "good" documentation is not enough

Dilbert by Scott Adams / www.dilbert.com

Simulation Design Process

1. Designer has a wonderful idea
2. Designer chooses a simulation tool/engine
3. Designer present results

Simulation Design process

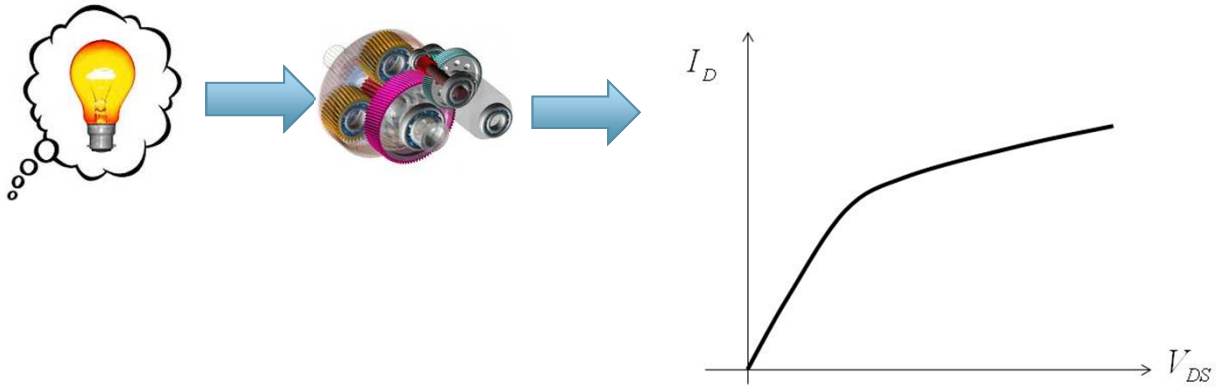
Designer chooses a simulation tool/engine

NEMO5

PURDUE UNIVERSITY

Simulation Design process

Designer presents results



ParaView
Parallel Visualization Application

MATLAB
The Language of Technical Computing

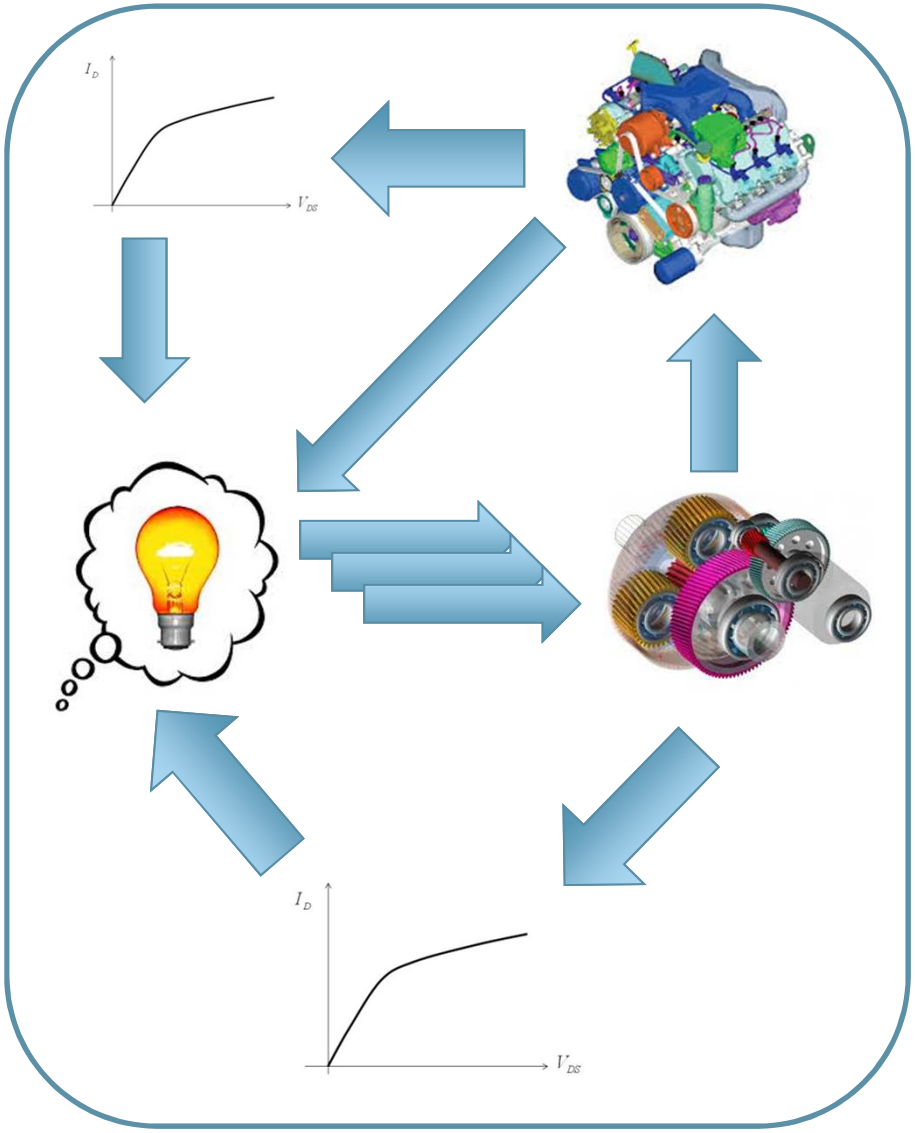
python

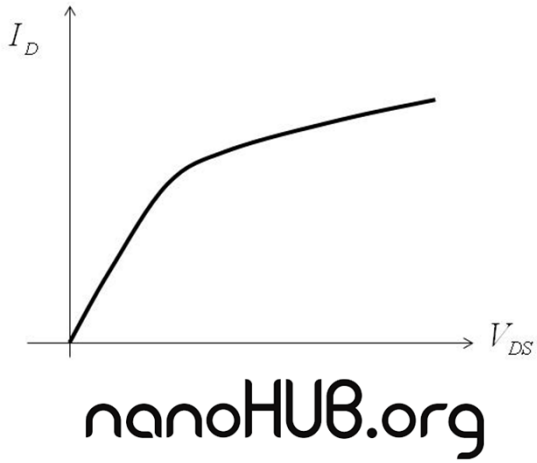
VISIT

nanoHUB.org

R

Simulation Design process



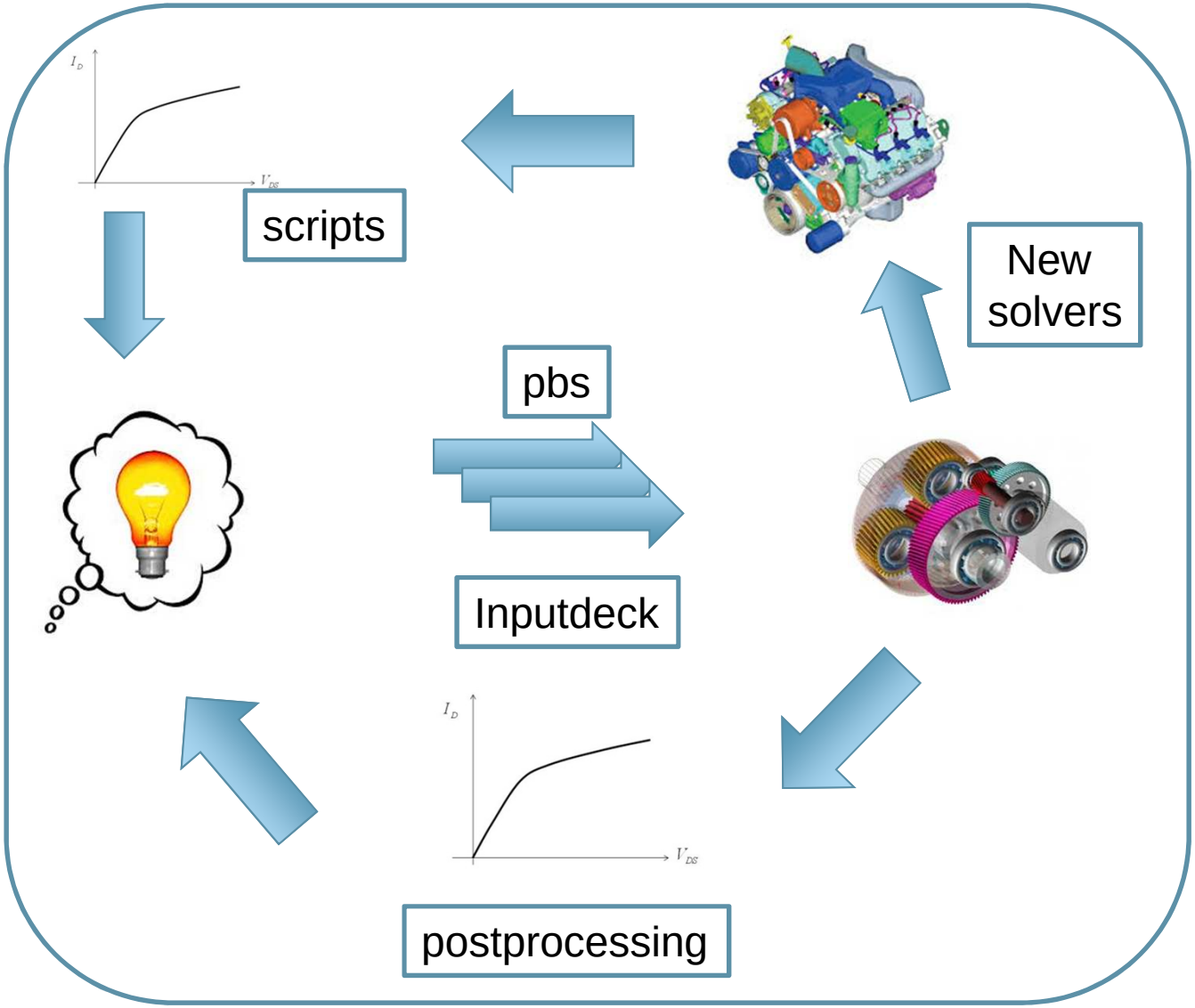


Nemo

Rappture

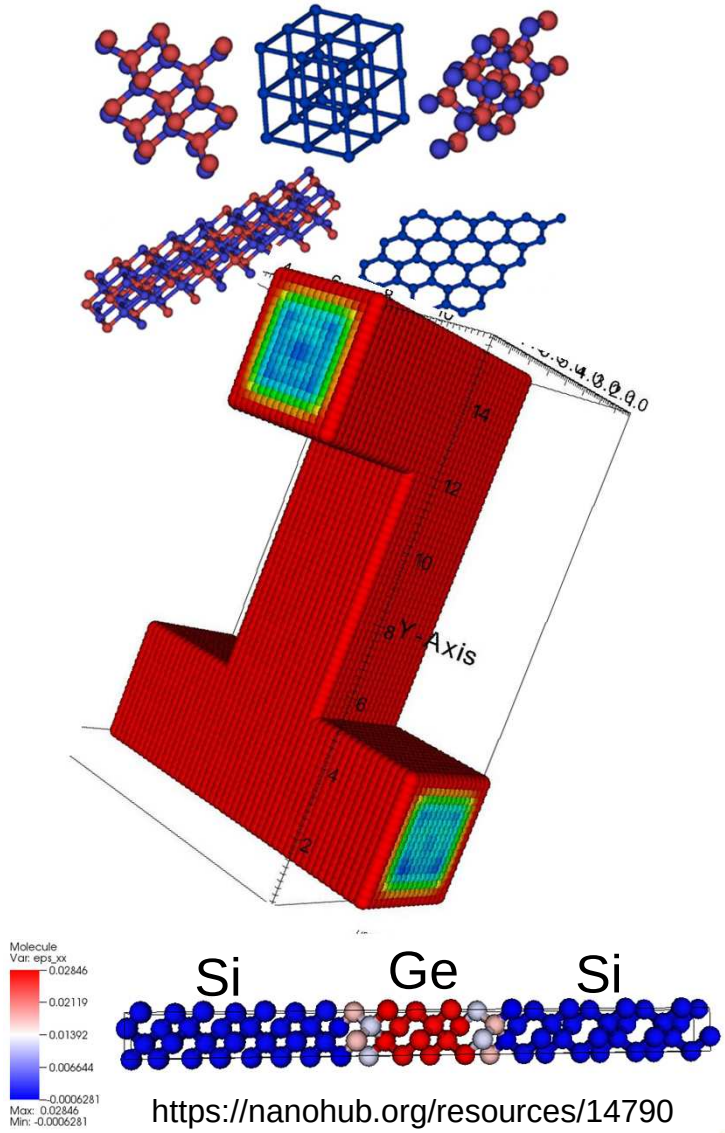
PURDUE UNIVERSITY

Simulation Design process



Nemo5 User Challenges

- Atomistic representation
  - » Crystallography: 2D/3D space
  - » material-dependent parameters
  - » Periodicity
- Arbitrary geometries
  - » Different geometric shapes
  - » Non-standard devices
- Dependant parameters
  - » Alloys
  - » VCA



https://nanohub.org/resources/14790

Nemo5 User Challenges

- Hardware resources.
  - » MPI Ranks, Cores - Nodes
  - » GPU/Intel Phi
- Algorithm decision.
  - » QTBM? RGF?
  - » Multi-Physics
  - » Non-linear Poisson / Semiclassical?
- Algorithms parametrization.
  - » Iterative Methods, Spacial Parallelization
  - » Preconditioners
  - » Number of iterations

http://www.olcf.ornl.gov/titan/

Nonlinear Systems		
Line Search	Trust Region	Other

Time Steppers			
Euler	Backward Euler	Pseudo Time Stepping	Other

Krylov Subspace Methods							
GMRES	CG	CGS	Bi-CGStab	TFQMR	Richardson	Chebychev	Other

Additive Schwarz	E J

Compressed Sparse Row (AIJ)

Vectors

SVD Solvers			
Cross Product	Cyclic Matrix	Lanczos	Thick R. Lanczos

Quadratic	
Linear-ization	Q-Arnoldi

Eigensolvers					
Krylov-Schur	Arnoldi	Lanczos	GD	JD	Other

Spectral Transformation				
Shift	Shift-and-invert	Cayley	Fold	Preconditioner

https://engineering.purdue.edu/gekcogrp/software-projects/nemo5/numerics.php

PURDUE UNIVERSITY

Objectives for new NEMO code:

- should include the functionality of the existing codes, e.g. *NEMO1D, NEMO3D, OMEN, OMEN3d_par,…*
- should replace nanohub tools, e.g. *1D Hetero, OMEN wire*

- Modular, clearly written and easily maintainable, documented
- Single code supported by many people

- Allows for expansions: new physics must be added easily

- Must be user friendly: clear script/GUI, easily visualized output

clear script/

Inputdeck

examples/BugInputDecks/Local_Band_Structure/LGB.in

```
tructure
{
    Material
    {
        name           = AlGaAs
        tag            = Substrate
        crystal_structure = zincblende
        regions        = (1)
        x = 0
        mole_fraction = 0
        disorder_type = totally_random_alloy //disorder
        seed = 1000
      //  Bands:TB:sp3sstar:param_set =  param_NEMO1D
        Bands:TB:sp3d5sstar_SO:param_set =  param_Klimeck
    }

    Material
    {
        name           = AlGaAs
        tag            = cap1
        crystal_structure = zincblende
        regions        = (2)
        x = 0
        mole_fraction = 0
        disorder_type = totally_random_alloy //disorder
        seed = 1000
     //   Bands:TB:sp3sstar:param_set = param_NEMO1D
        Bands:TB:sp3d5sstar_SO:param_set =  param_Klimeck
    }

    Material
    {
        name           = GaAs
        tag            = cap2
        crystal_structure = zincblende
        regions        = (3)
      //  Bands:TB:sp3sstar:param_set =  param_NEMO1D
        Bands:TB:sp3d5sstar_SO:param_set =  param_Klimeck
        Lattice:strain_beta_As_Al_Ga= sqrt(AlAs:Lattice:strain_beta_Al_As_As * GaAs:Lattice:\
strain_beta_Ga_As_As)
        Lattice:strain_alpha_Al_As= AlAs:Lattice:strain_alpha_As_Al
        Lattice:strain_beta_As_Al_Al=8.94
         Lattice:strain_alpha_In_As = 0.55 * 35.18
         Lattice:strain_alpha_As_In = 0.55 * 35.18
        Lattice:strain_beta_As_Al_In=strain_beta_As_Ga_In



    }
 crystal_structure = zincblende
        regions        = (4)
        x = 0.15
        mole_fraction = 0.15
        disorder_type = totally_random_alloy //disorder
        seed = 1000
      //  Bands:TB:sp3sstar:param_set =  param_NEMO1D
        Bands:TB:sp3d5sstar_SO:param_set =  param_Klimeck
        Lattice:strain_alpha_In_As =  0.55 * 35.18
        Lattice:strain_alpha_As_In =  0.55 * 35.18
      // Lattice:strain_alpha_Ga_As = 77.87673

    }
    Material
    {
        name           = InAs
        tag            = dot
        crystal_structure = zincblende
        regions        = (5,6)
     //   Bands:TB:sp3sstar:param_set =  param_NEMO1D
        Bands:TB:sp3d5sstar_SO:param_set =  param_Klimeck
        Lattice:strain_alpha_In_As =  0.55 * 35.18
        Lattice:strain_alpha_As_In =  0.55 * 35.18
        Lattice:strain_alpha = 0.55 * 35.18
        Lattice:strain_beta_As_In_Al = strain_beta_As_In_Ga


    }

    Domain
    {
        name        = atomic_structure
        type        = pseudomorphic
        base_material = Substrate
        dimension    = (10,10,120)   // dimensions in unit cells
        periodic     = (true,true,false) // to rlax in the z direction

        crystal_direction1 = ( 1,0,0 )
        crystal_direction2 = ( 0,1,0 )
        crystal_direction3 = ( 0,0,1 )
        space_orientation_dir1 =  (1, 0, 0)
        space_orientation_dir2 =  (0, 1, 0)
        origin = (0, 0, 0)
        passivate = true
        random_alloy=true
        regions = (1,2,3,4,5,6)
        geometry_description   = simple_shapes
    }
```

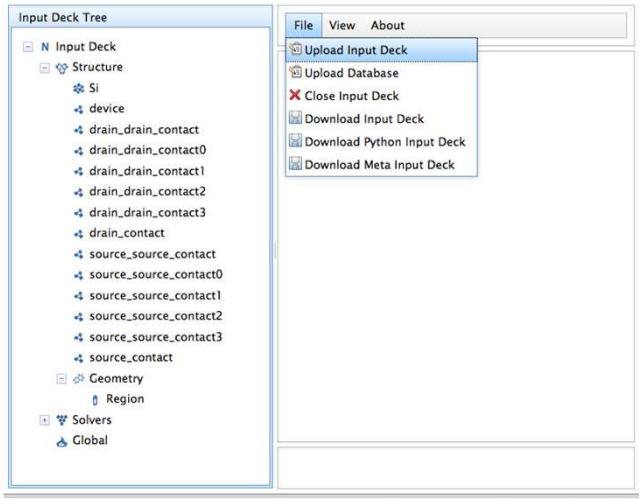
```
   Hamilton_constructor = schroeddd0

                  NEGF_object_list    = (contact_retarded_self_Fermiondd)

                    contact_retarded_self_Fermiondd_constructor   = Propagation4dd
                    contact_retarded_self_Fermiondd_solver       = Propagation4dd

                  contact_retarded_self_Fermiondd_lead_domain = drain_drain_contact1

                    // the next few lines obtain the Hamiltonian for the four atomic layers
                    // see equation (9) of reference [1]
                    Hamilton_constructor0 = schroeddd0
                    Hamilton_constructor1 = schroeddd1
                    Hamilton_constructor2 = schroeddd2
                    Hamilton_constructor3 = schroeddd3
                    lead_domain0 = drain_drain_contact1
                    lead_domain1 = drain_drain_contact2
                    lead_domain2 = drain_drain_contact3
                    lead_domain3 = drain_contact

                    ratio_of_eigenvalues_to_be_solved = 1.0 //between 0-1, usually in exact solution: 0.6, in LRA 0.05
                    //    LRA=false //true, false
                    Parallelizer   = d1:Propagation_Parallelizer
                    retarded_lead_method = transfer_matrix

                    //    do_output = false
                    Hamilton_momenta     = (fake_momentum_2D)
                    energy_constructor   = d1:Propagation_Parallelizer
                    //    e_momentum_1D_constructor = Propagation1
                    momentum_names       = (energy,fake_momentum_2D)
                    fake_momentum_2D_constructor = d1:schroed1
                   }

                      // ============================
                  // Schroedinger in the DD0 layer
                      // ============================
                   solver
                   {
                    job_list=(assemble_H,passivate_H)

                    output=(energies,k-points,Hamiltonian)

                    name = schroeddd0
                    type = Schroedinger
                    domain = drain_drain_contact0
                    active_regions = (1)

                    tb_basis = sp3sstar_SO
              parallelize_here = false

              Hamilton_constructor_drain_drain_contact1 = schroeddd1

                    k_points = [(0,0,0)]

 threshold_energy = -15
                    number_of_nodes = 1
                   }
           }
          }
     }

// ####################################################################################################
// this solver obtains the self energy for the source contact
// calculates sigmas=Ts*Gs*Ts' with T the coupling Hamiltonian between source contact and device
//
==========================================================================================
     solver
     {
      name      = Propagation4s
      type      = Self_Propagation
      domain    = device
      active_regions = (1)
      Hamilton_constructor = d1:schroed1

      NEGF_object_list     = (retarded_Green_Fermion_device0,contact_retarded_self_Fermions)

      contact_retarded_self_Fermions_constructor   = Propagation4s
      contact_retarded_self_Fermions_solver        = Propagati iterator
              {
               set{}
               set
               {
                       domain = device_0
               }
               set
               {
                       domain = device_1
              Hamilton_constructor_device_0 = d1:Propagation_rgf_ret:schroed_neighbor
               }
              loop(2)
```

```
fake_momentum_2D_constructor = d1:schroed1

           retarded_Green_Fermion_last_domain_inversion = exact //exact,LRA

       transmission = false
       selfenergy1_for_transmission = contact_retarded_self_Fermiond_last
       contact_retarded_self_Fermiond_last_solver = d1:Propagation_last:Propagation_last_inv:Propagation4d
       selfenergy2_for_transmission = contact_retarded_self_Fermion_rgf
       contact_retarded_self_Fermion_rgf_solver = d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self

       _step = (Propagation_last_inv)
      _disable_reinit = true
      _disable_solve = true

       // =======================
       // iterate towards the last subdomain
   // =======================
       iterator
       {
                set
                {
                      _disable_solve = true
                      _disable_output = true
                      transmission = false
                }
                set
                {
                      _disable_reinit = false
                      _disable_solve = false
                      _disable_output = false
                      domain = device_3
                      output_suffix = last
                      transmission = true
                      do_output = true
                }
          }

       // =================================================================
       // this solver obtains the inverse Green's function for the subdomains
       // calculates invGr=E-H-sigma
       // =================================================================
       solver
       {
        name          = Propagation_last_inv
        type          = Green_Propagation
       domain         = device
        active_regions = (1)
        Hamilton_constructor = d1:Propagation_rgf_ret:schroed_rgf

        NEGF_object_list     = (inverse_retarded_Green_Fermion_last_domain, contact_retarded_self_Fermion_rgf,
contact_retarded_self_Fermiond_last)

        contact_retarded_self_Fermion_rgf_constructor   =
d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self
        contact_retarded_self_Fermion_rgf_solver        =
d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self

    contact_retarded_self_Fermion_rgf_lead_domain = device_2
 {
                   job_list=(assemble_H,passivate_H)

                   output=(energies,k-points,Hamiltonian)

                   name = schroeddd3
                   type = Schroedinger
                   domain = source_source_contact3
                   active_regions = (1)

                   tb_basis = sp3sstar_SO
                   parallelize_here = false

                   Hamilton_constructor_source_contact = d1:Propagation4s:schroedd
                   Hamilton_constructor_source_source_contact2 = schroeddd2

                   k_points = [(0,0,0)]
                   k_space_basis = cartesian
                   chem_pot = 0
                   temperature = 300
                   threshold_energy = -15
                   number_of_nodes = 1
                  }
            }
          } name          = Repartition
       type          = Repartition
       domain        = device
               start_from_one_contact = true
      }

      // ###// ###############################################################
```

```
 }

     // =================================================================
     // this solver obtains the inverse Green's function for the subdomains
     // calculates invGr=E-H-sigma
     // =================================================================
     solver
     {
      name          = Propagation_rgf_inv
      type          = Green_Propagation
            domain        = device
      active_regions = (1)
      Hamilton_constructor = d1:Propagation_rgf_ret:schroed_rgf

      //NEGF_object_list     = (inverse_retarded_Green_Fermion_device_rgf,contact_retarded_self_Fermion_rgf)
      NEGF_object_list     = (inverse_retarded_Green_Fermion_device_rgf,contact_retarded_self_Fermion_rgf)

      contact_retarded_self_Fermion_rgf_constructor   =
d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self
      contact_retarded_self_Fermion_rgf_solver        =
d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self

              contact_retarded_self_Fermion_rgf_constructor   =
d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self
      contact_retarded_self_Fermion_rgf_solver        =
d1:Propagation_rgf_ret:Propagation_rgf_inv:Propagation_rgf_self


        contact_retarded_self_Fermion_rgf_lead_domain = source_contact


      inverse_retarded_Green_Fermion_device_rgf_constructor = Propagation_rgf_inv
        inverse_retarded_Green_Fermion_device_rgf_solver      = Propagation_rgf_inv

      Parallelizer   = d1:Propagation_Parallelizer

        //do_output = true

      Hamilton_momenta     = (fake_momentum_2D)
        energy_constructor   = d1:Propagation_Parallelizer
      momentum_names       = (energy,fake_momentum_2D)
        fake_momentum_2D_constructor = d1:schroed1

        // =======================
        // iterate over subdomains
        // =======================
            _step = (Propagation_rgf_self)
        iterator
        {
                set
                {
                      _disable_solve = true
                }
                set
                {
                      domain = device_0
                }
                loop(3)
 active_regions = (1)
              Hamilton_constructor = d1:Propagation_rgf_ret:schroed_rgf

              NEGF_object_list     = (contact_retarded_self_Fermion_rgf,retarded_Green_Fermion_device0)

              contact_retarded_self_Fermion_rgf_constructor   = Propagation_rgf_self
              contact_retarded_self_Fermion_rgf_solver        = Propagation_rgf_self

              contact_retarded_self_Fermion_rgf_lead_domain = source_contact

              retarded_Green_Fermion_device0_constructor = d1:Propagation4s:Propagation2
              retarded_Green_Fermion_device0_solver      = d1:Propagation4s:Propagation2

              Parallelizer   = d1:Propagation_Parallelizer

              iterative_damping = 1 //0.67032 //exp(-0.4)
                //do_output = true
              energy_constructor   = d1:Propagation_Parallelizer
              //e_momentum_1D_constructor = Propagation1
```

■ ■ ■

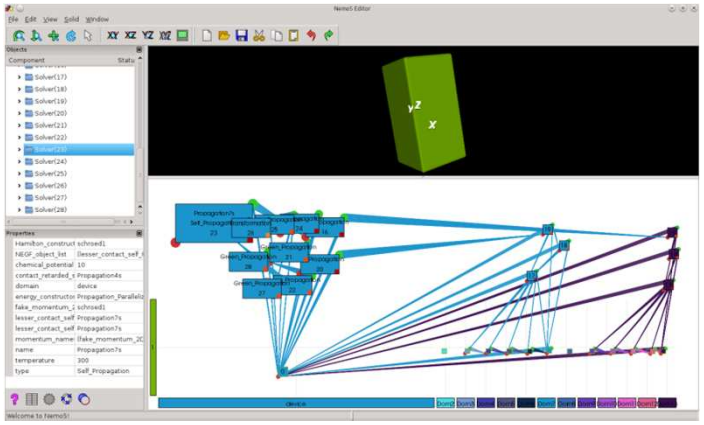
Prototype #1 (InputDeck Editor)

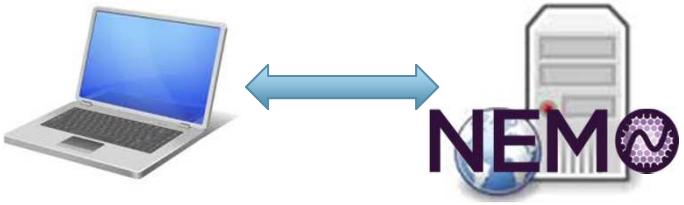


- Features:
  - » Load inputdecks (.in) and database (.mat)
  - » Modify/Add options
  - » Download ‘new’ inputdeck (.in, .py)
- Good
  - » No software (client) installation required
  - » Portable (web browsers)
  - » Visualize inputdeck hierarchy
- Bad
  - » Web-server is required + php interpreter installed
  - » No device visualization
  - » No user feedbacks
  - » No entities (Solvers, Domains, …) state
  - » No Nemo5 execution
  - » Session files
  - » It doesn’t work for recent regions descriptors

PURDUE UNIVERSITY

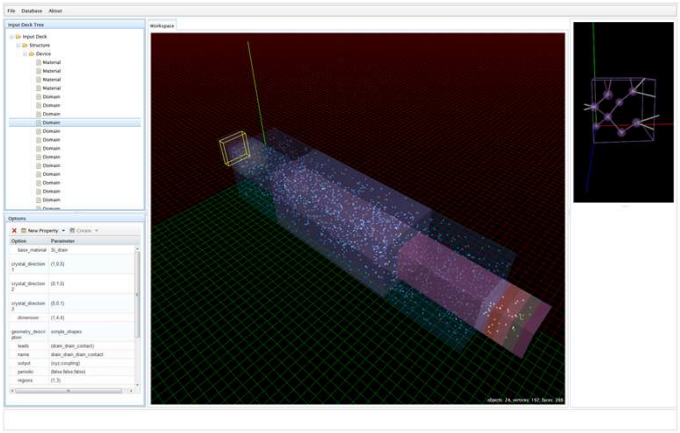
Prototype #2 (NemUI)

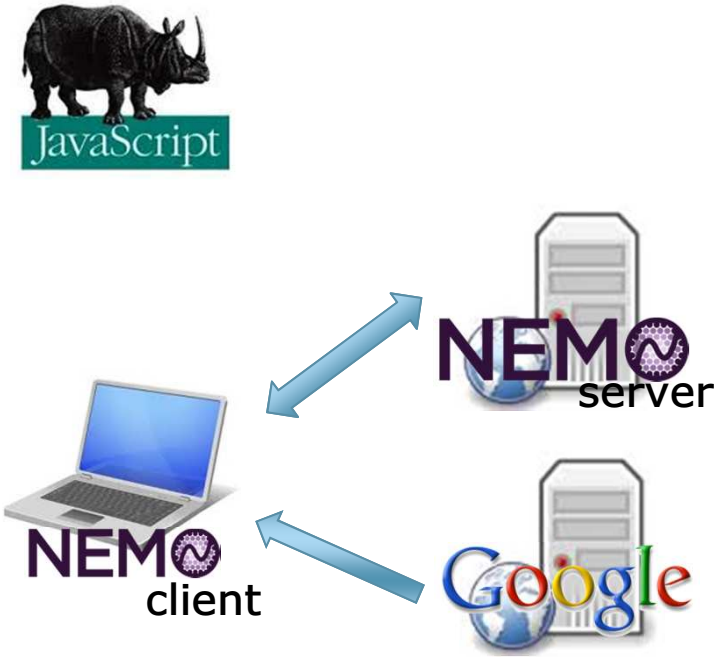




- Features:
  - » Load + Visualize inputdecks
  - » Visualize 3D Device representation
  - » Save modified inputdeck (.in, .py)
- Good
  - » Nemo5 as a library
  - » Entities Wizards
  - » Visualize inputdeck hierarchy
  - » Visualize solvers dependencies
- Bad
  - » Additional libraries required (WxWidget + GTK)
  - » No portable
  - » No user help for new options
  - » 3D user interaction slow (remote conections)

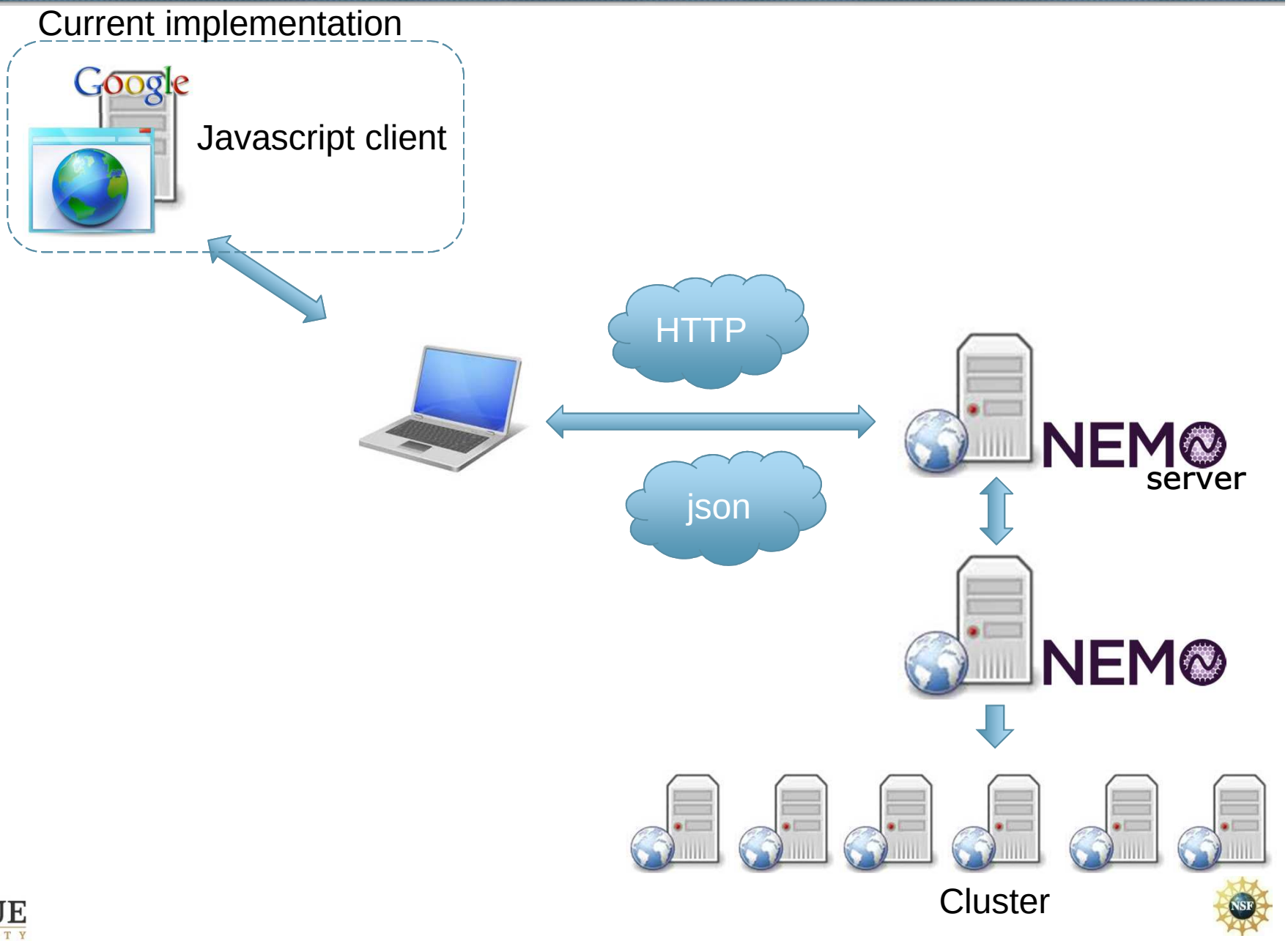
Prototype #3 (Nemo Server)





- Features:
  - » Load + Visualize +Interact with inputdecks
  - » Visualize 3D Device representation
  - » Visualize Atomistic representation
  - » Save modified inputdeck (.in, .py)
- Good
  - » Nemo5 as a library
  - » Visualize inputdeck hierarchy
  - » Portable
  - » Additional software (client) installation is optional
  - » Extensible
  - » Connected with Nemo 5 auto-documentation
  - » ‘unroll’ Device templates
- Bad
  - » Still under development, pre-alpha version
  - » It doesn’t work for recent regions descriptors

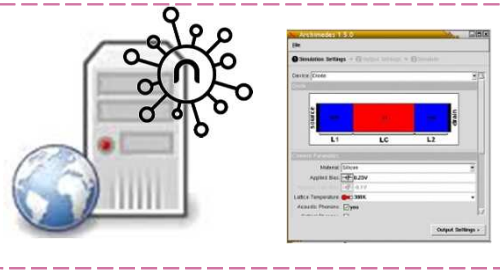
Nemo Server Architecture (1)



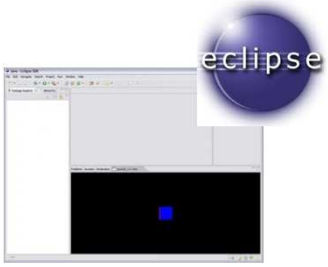
Nemo Server Architecture (2)

Current implementation

Javascript client



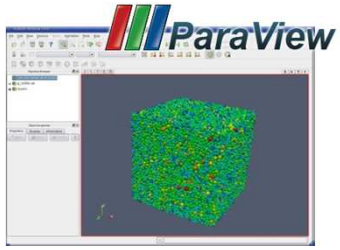
Auto-generation of rappture XMLs is possible



Eclipse Plugin

HTTP

json

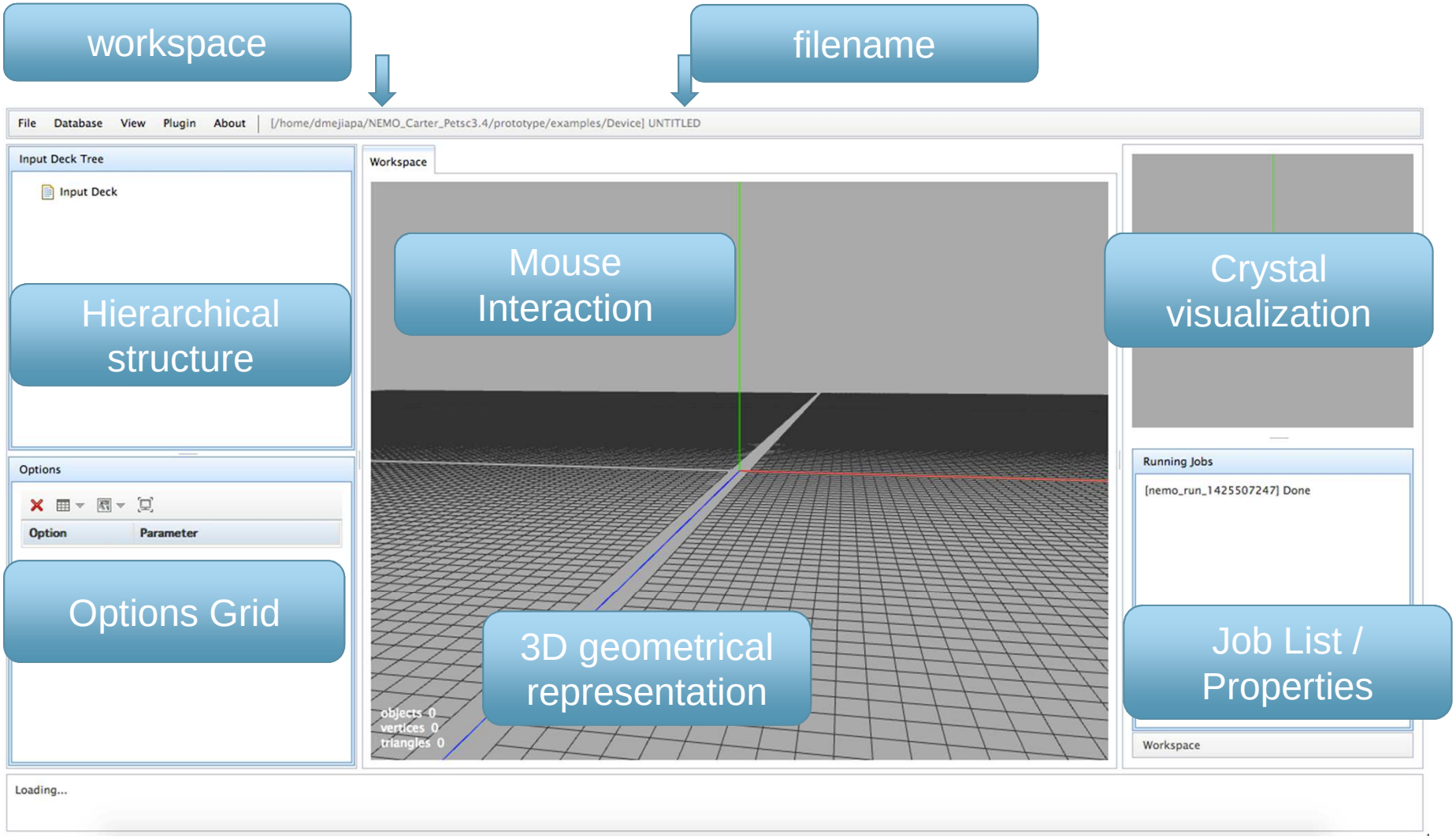


Paraview Plugin

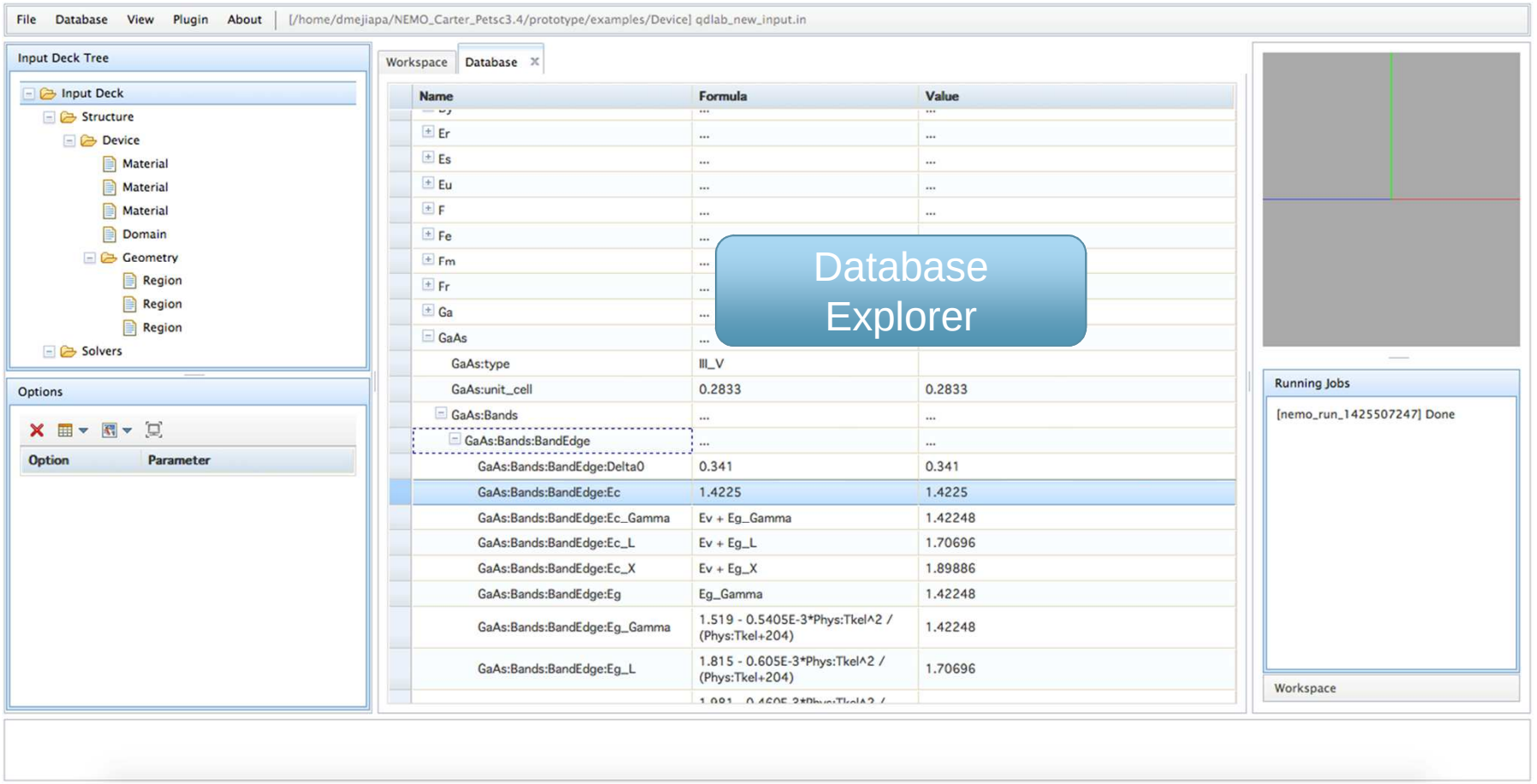
Cluster

PURDUE UNIVERSITY

CANVAS


workspace
filename
File
Database
View
Plugin
About
[/home/dmejiapa/NEMO_Carter_Petsc3.4/prototype/examples/Device] UNTITLED
Input Deck Tree
Input Deck
Hierarchical structure
Workspace
Mouse Interaction
Crystal visualization
Options
Option
Parameter
Options Grid
objects 0
vertices 0
triangles 0
3D geometrical representation
Running Jobs
[nemo_run_1425507247] Done
Job List / Properties
Workspace
Loading...

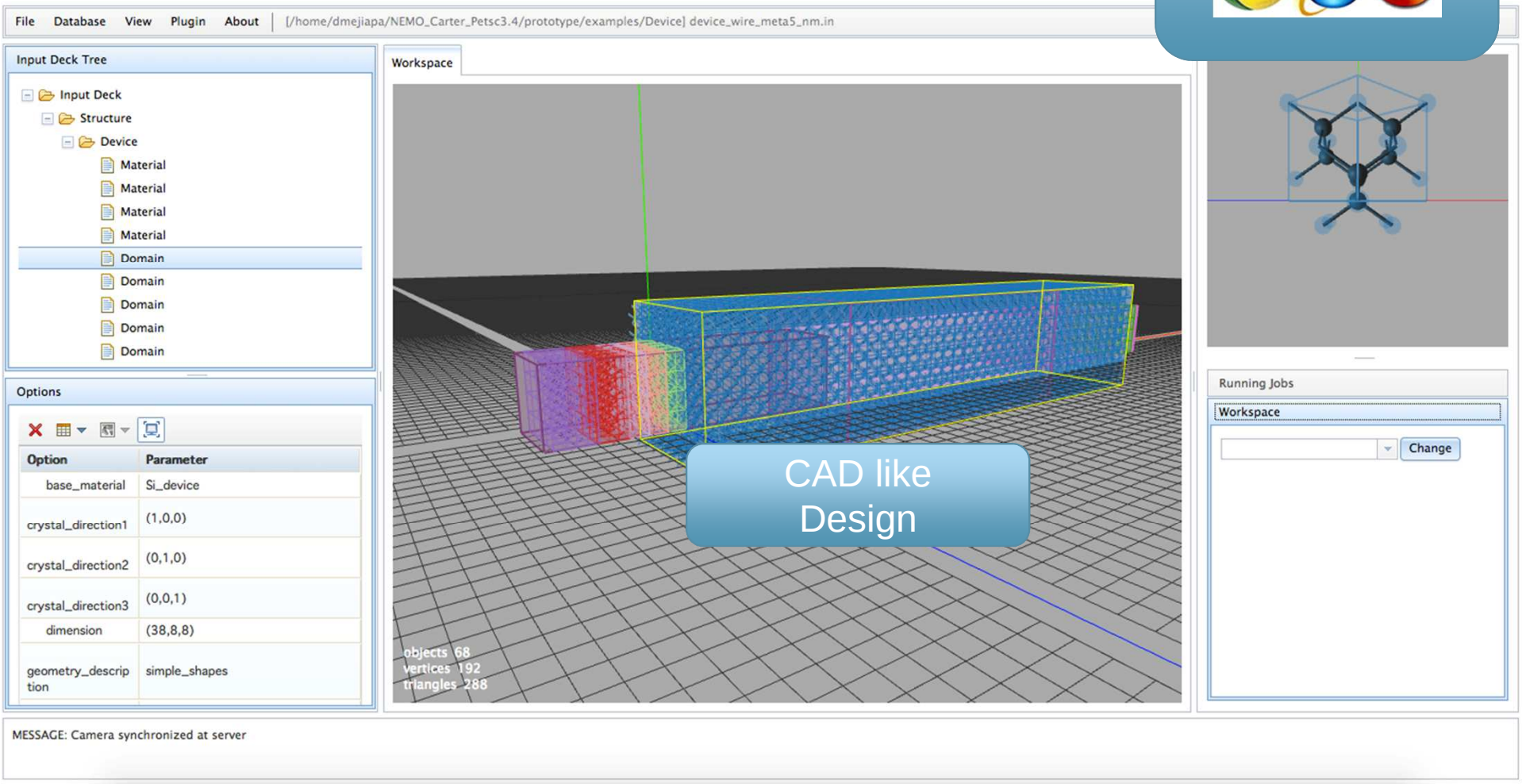
DATABASE EXPLORER



PURDUE UNIVERSITY

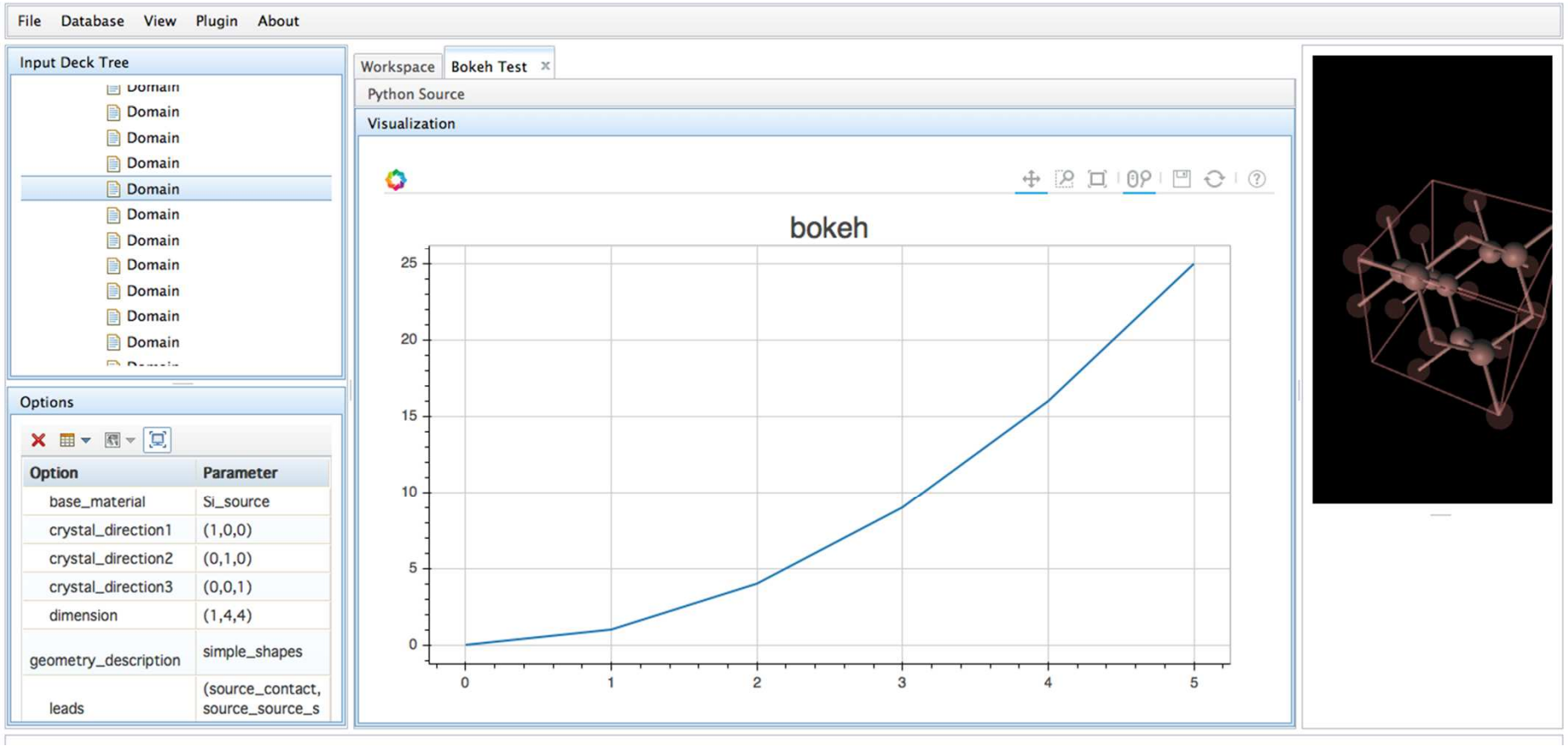
WEBGL

WEBGL



PURDUE UNIVERSITY

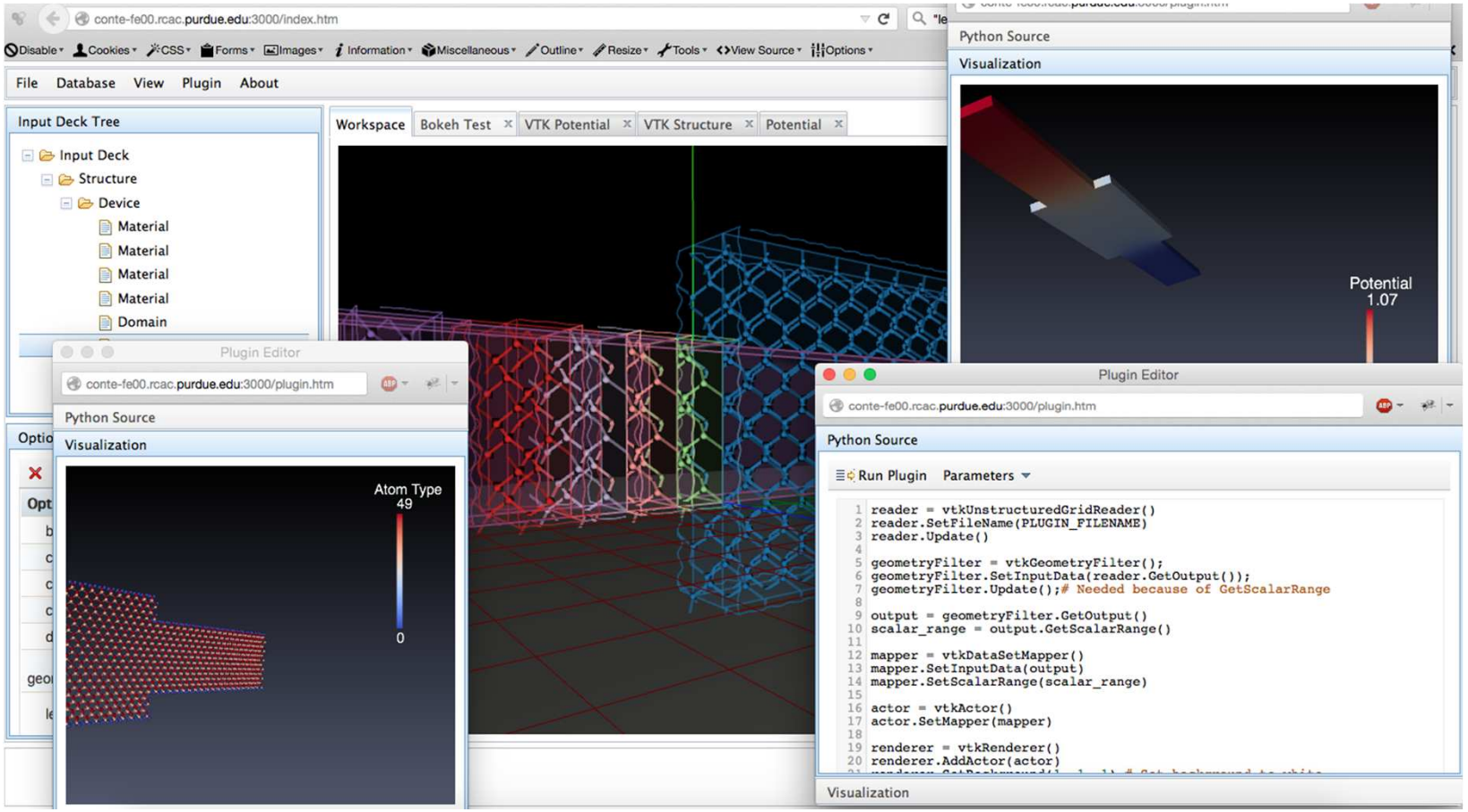
PLUGINS



Python Plugins supported for postprocessing

PURDUE UNIVERSITY

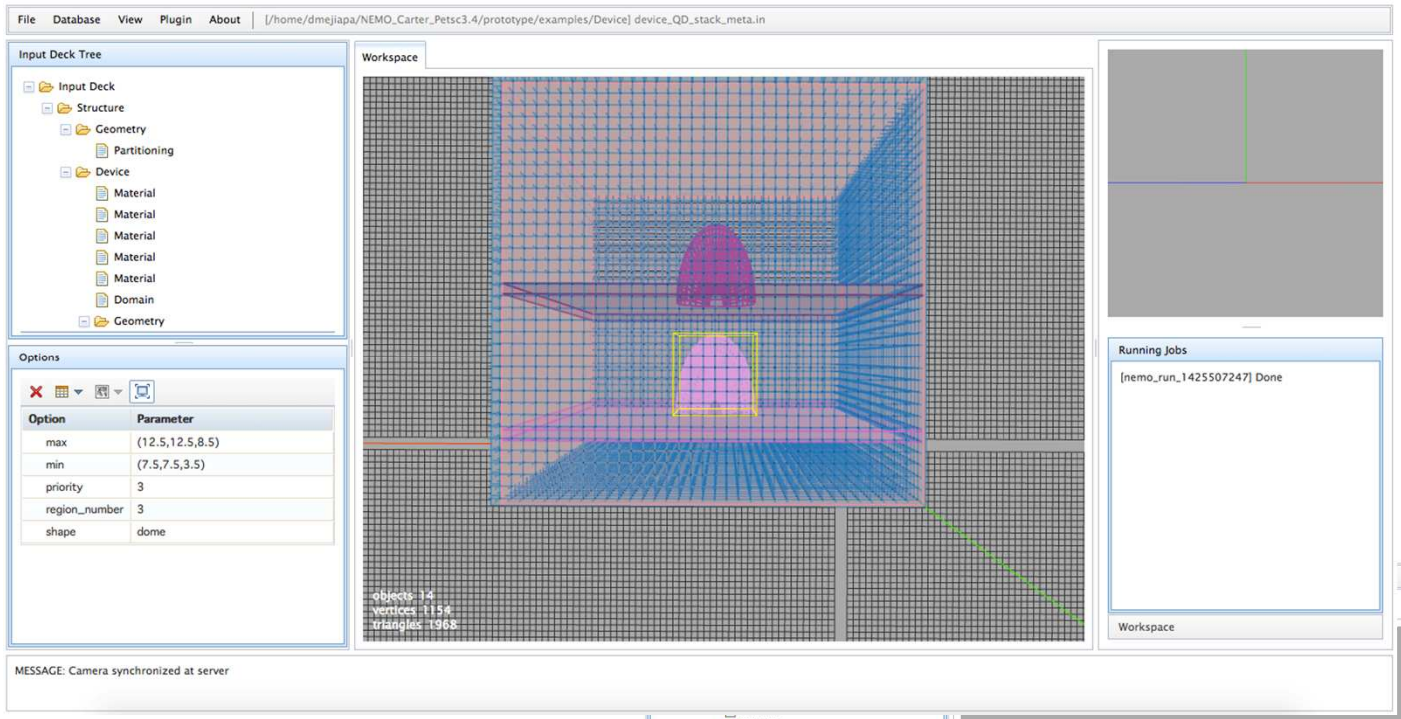
PLUGINS PARAVIEW

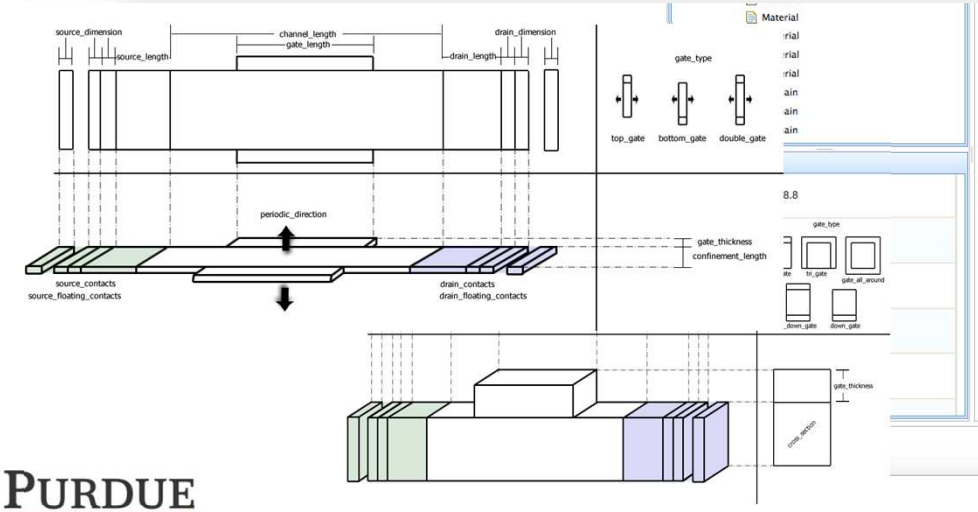


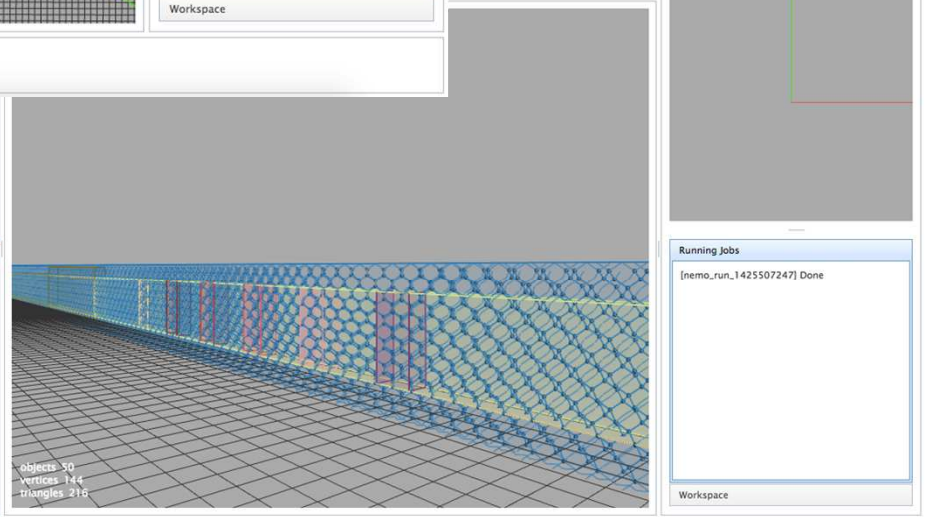
Plugin embedded or as ‘pop-ups’

Paraview python-state scripts supported as a plugin

TEMPLATE DEVICES

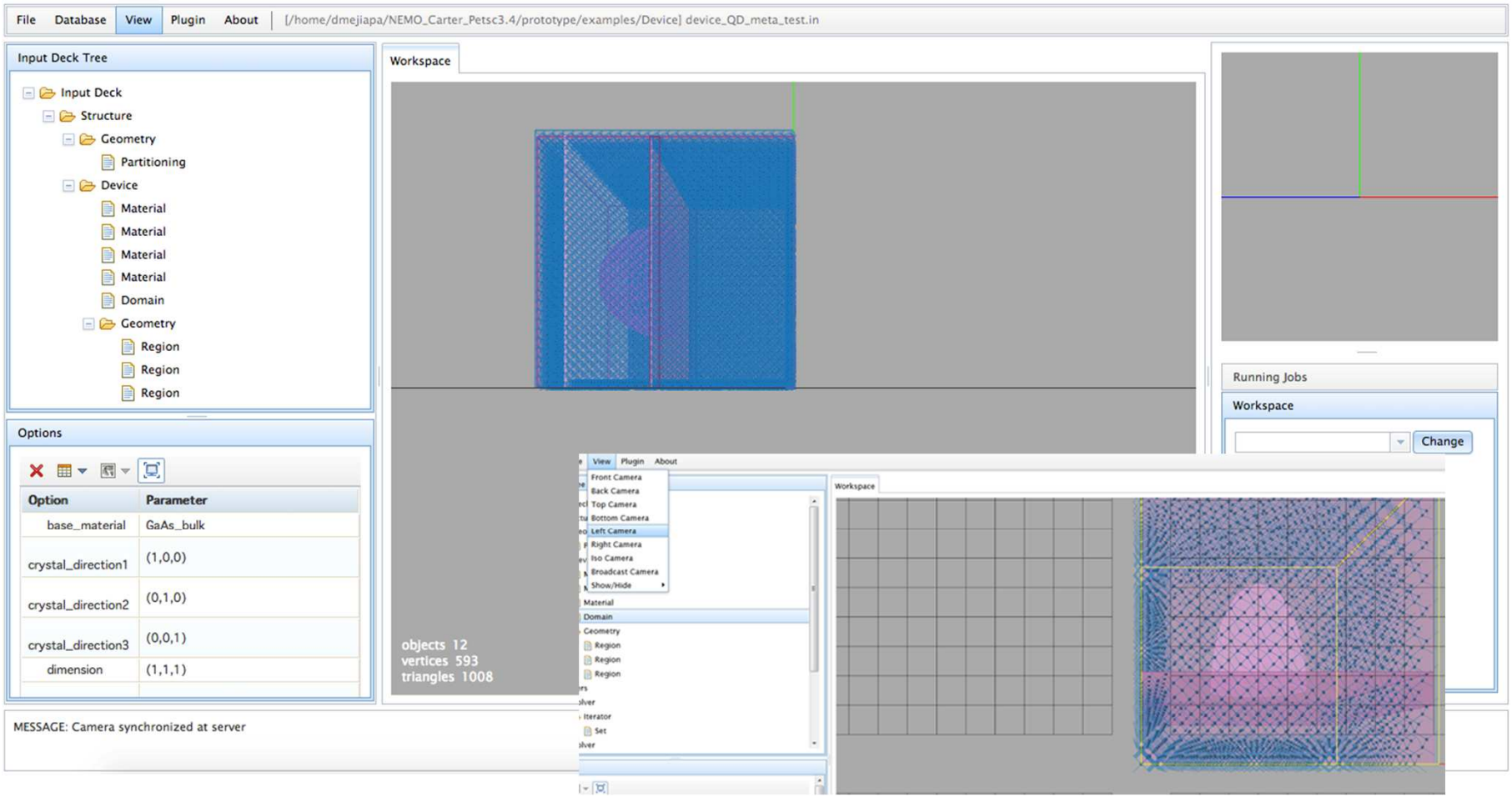






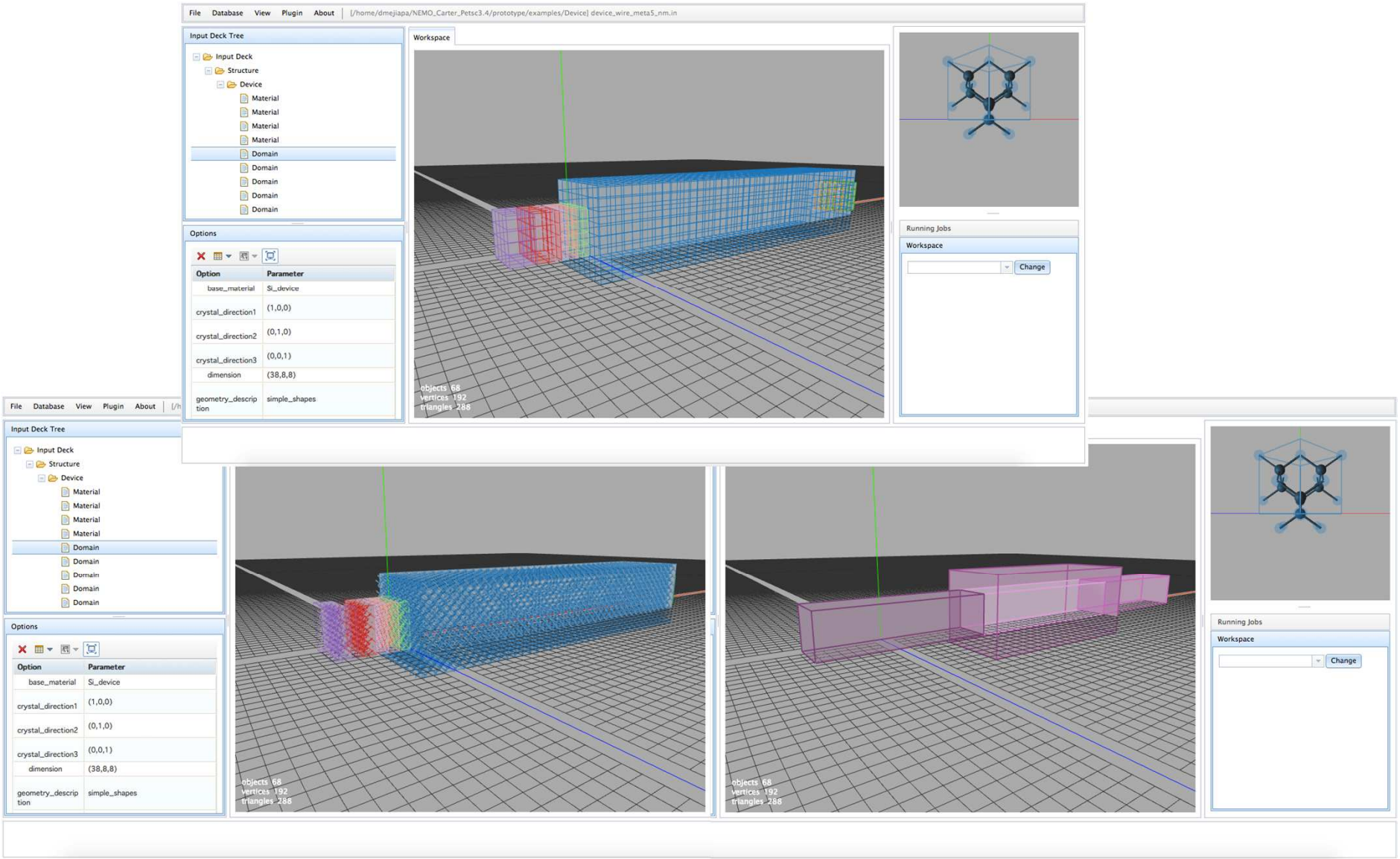
PURDUE UNIVERSITY

PREDEFINED VIEWS



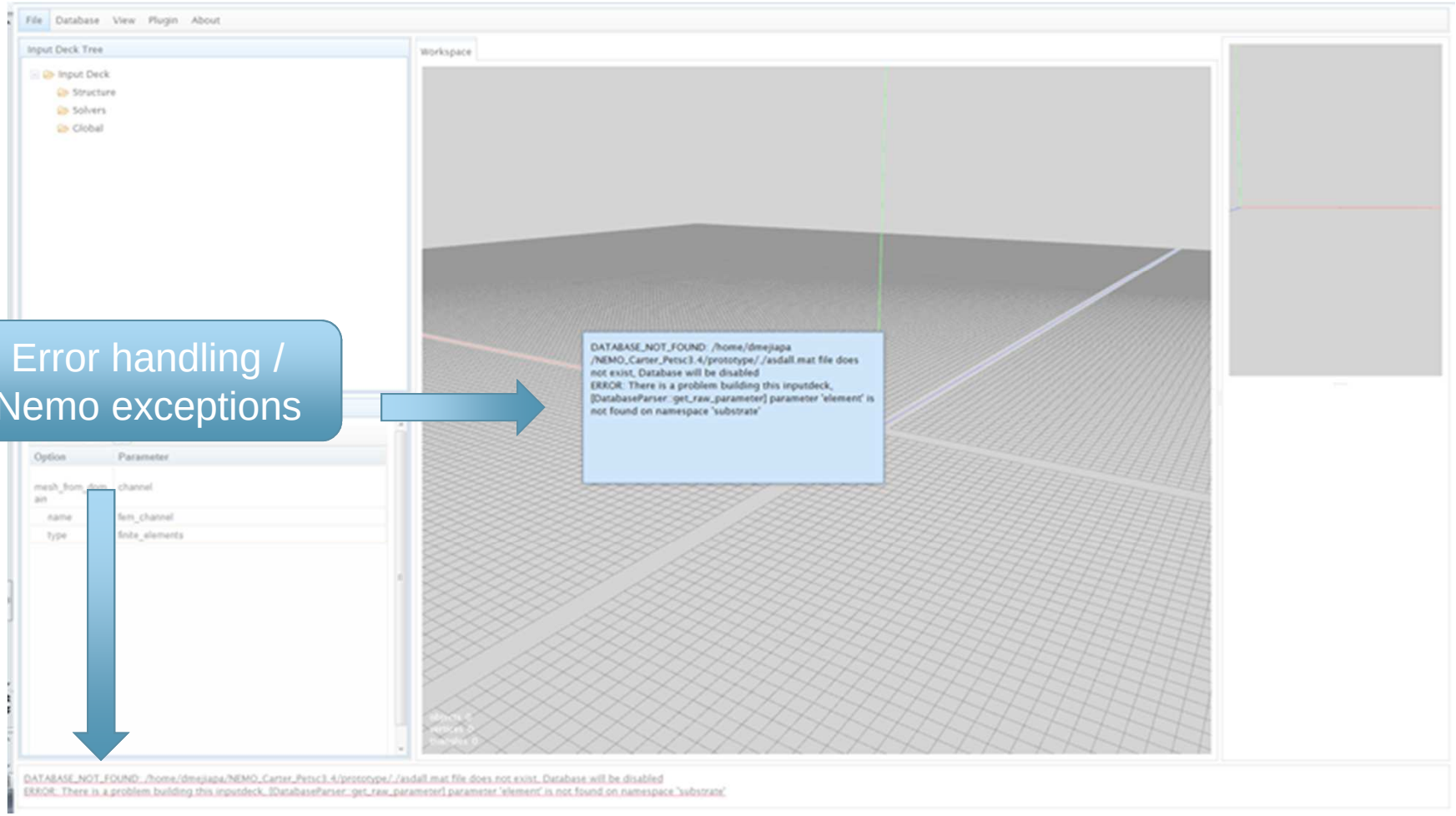
PURDUE UNIVERSITY

HIDE/SHOW GROUPS



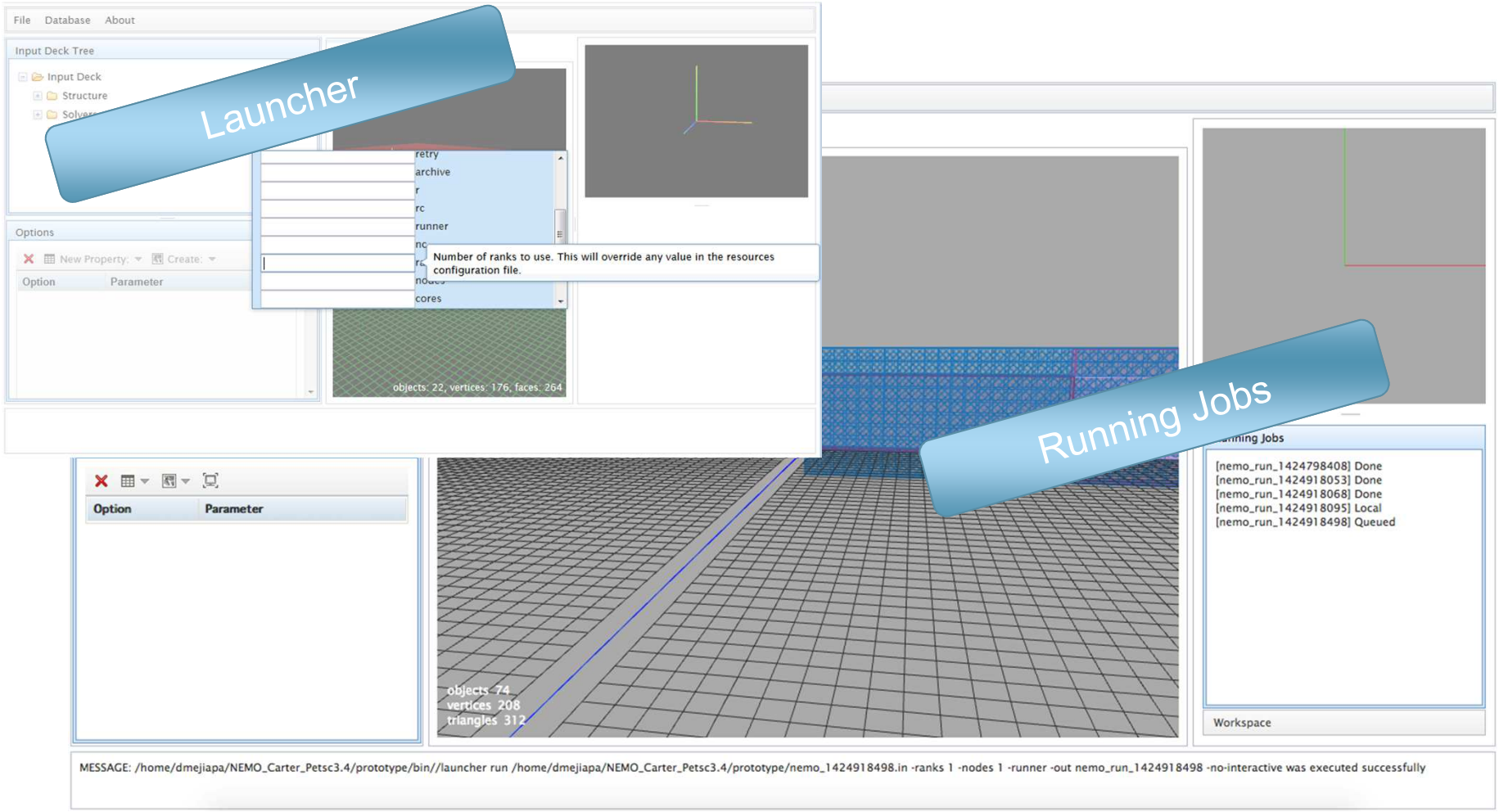
PURDUE UNIVERSITY

ERROR HANDLING



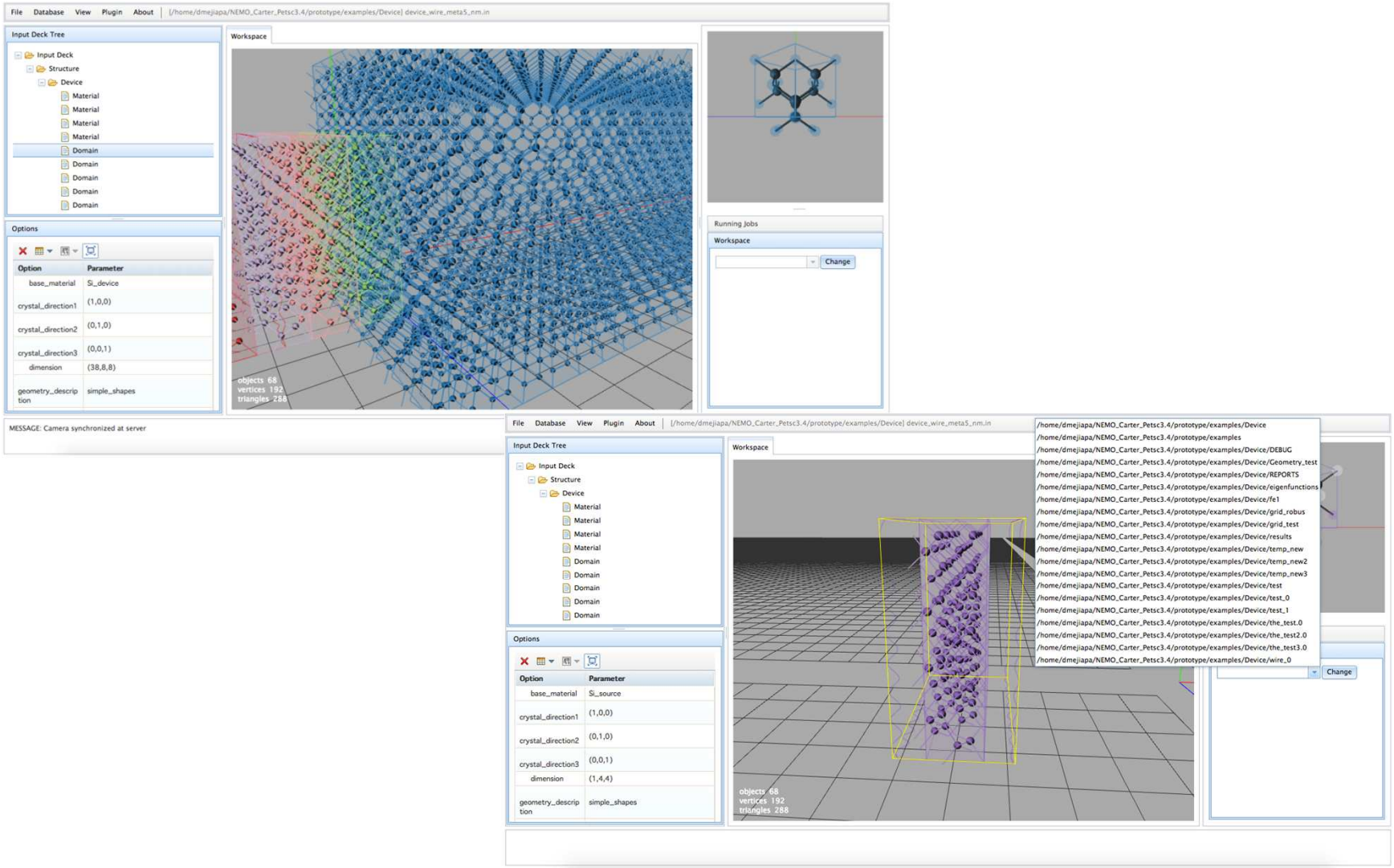
PURDUE UNIVERSITY

LAUNCHER



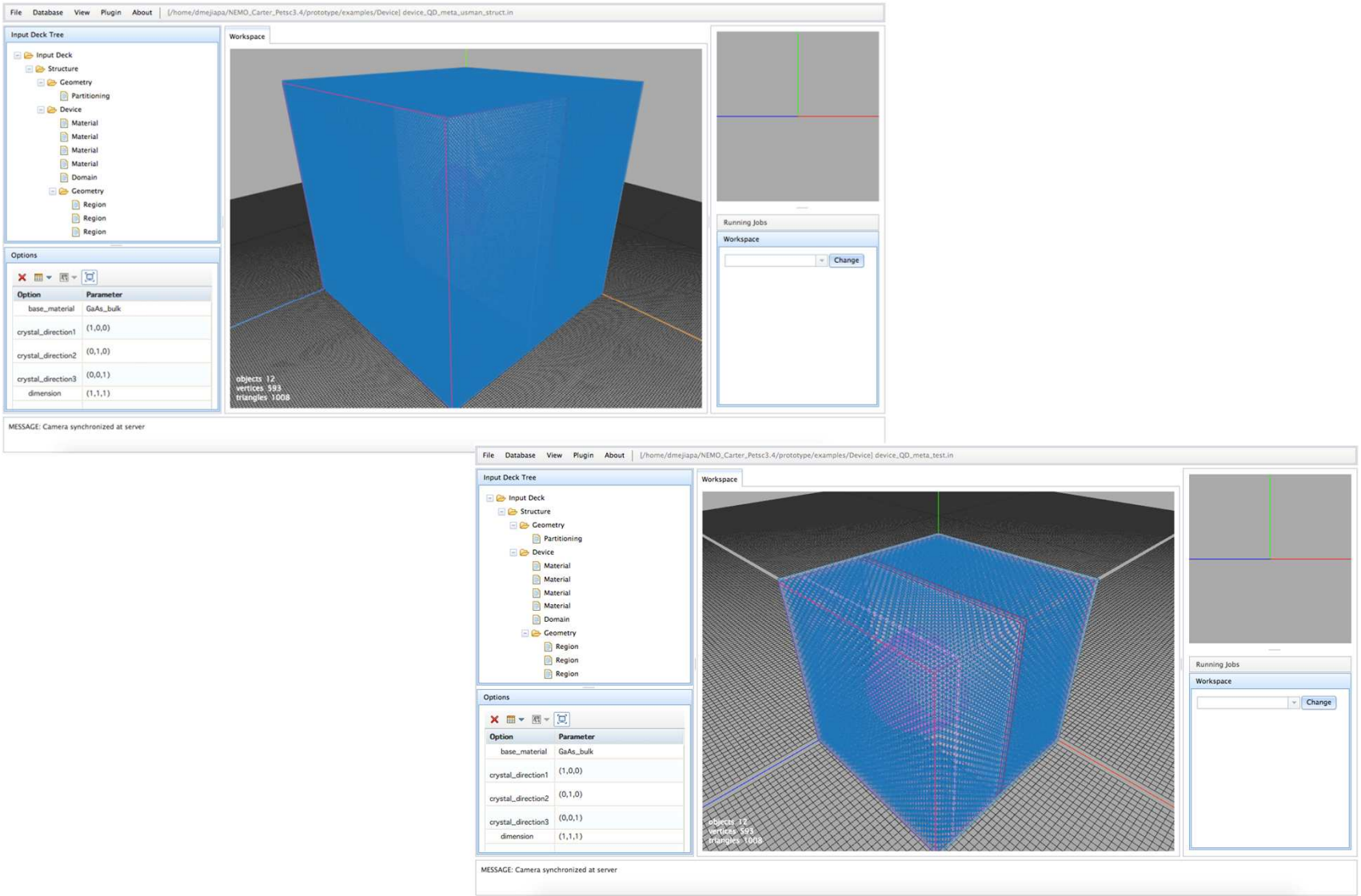
PURDUE UNIVERSITY

GALLERY I

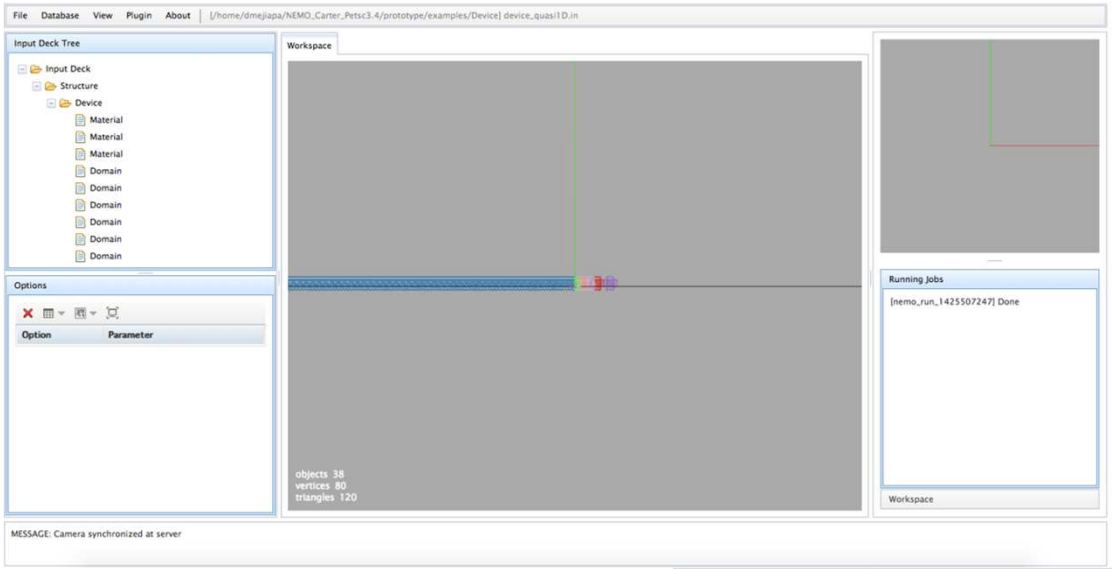


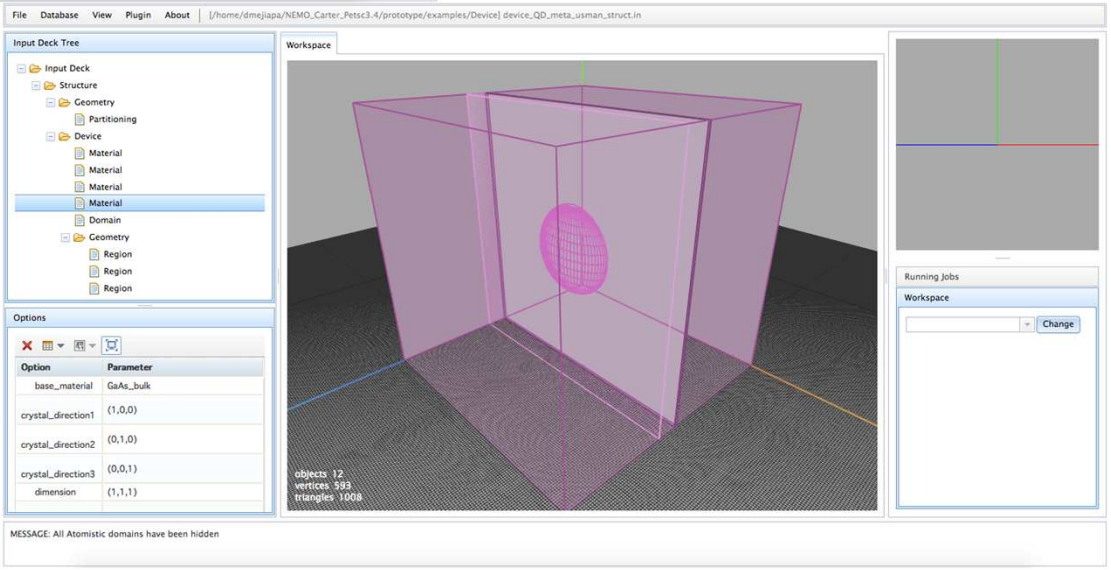
PURDUE UNIVERSITY

GALLERY II


File Database View Plugin About
Input Deck Tree
Input Deck
Structure
Geometry
Partitioning
Device
Material
Material
Material
Material
Domain
Geometry
Region
Region
Region
Workspace
Running Jobs
Workspace
Change
Options
Option
Parameter
base_material
GaAs_bulk
crystal_direction1
(1,0,0)
crystal_direction2
(0,1,0)
crystal_direction3
(0,0,1)
dimension
(1,1,1)
objects 12
vertices 593
triangles 1008
MESSAGE: Camera synchronized at server

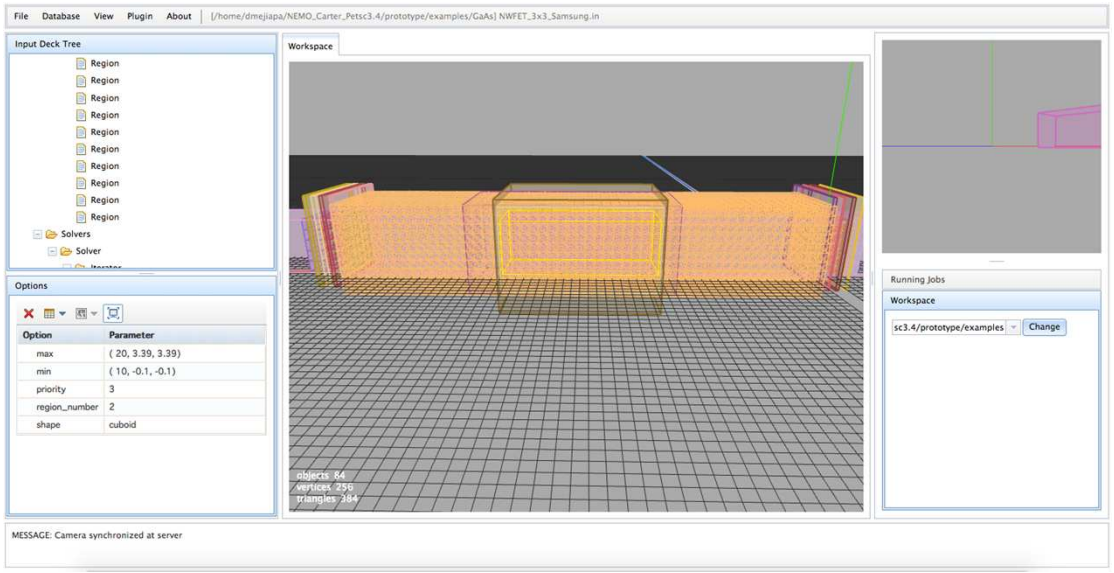
GALLERY III

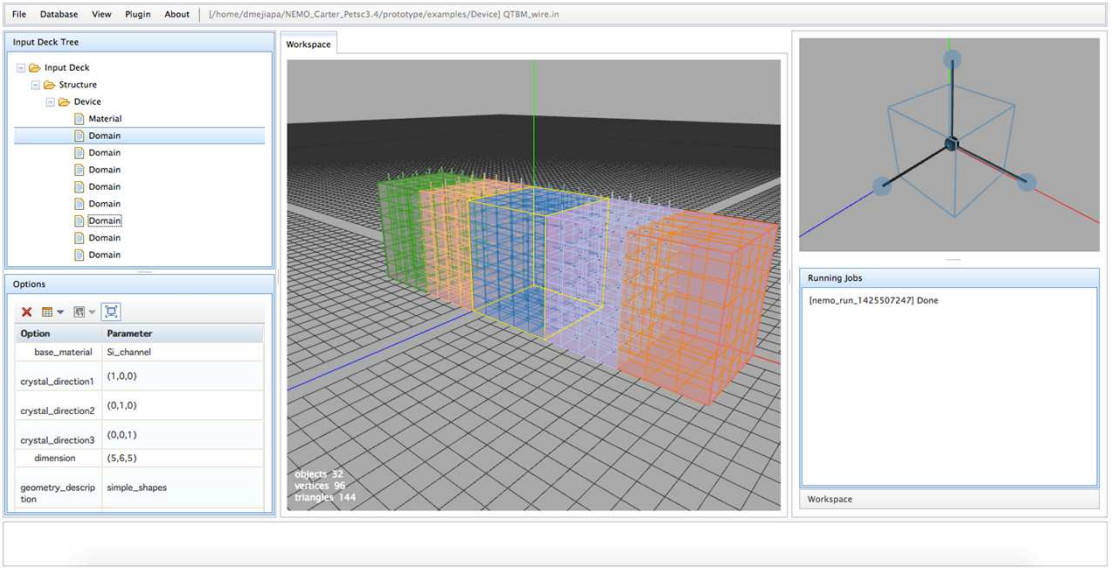




PURDUE UNIVERSITY

GALLERY IV

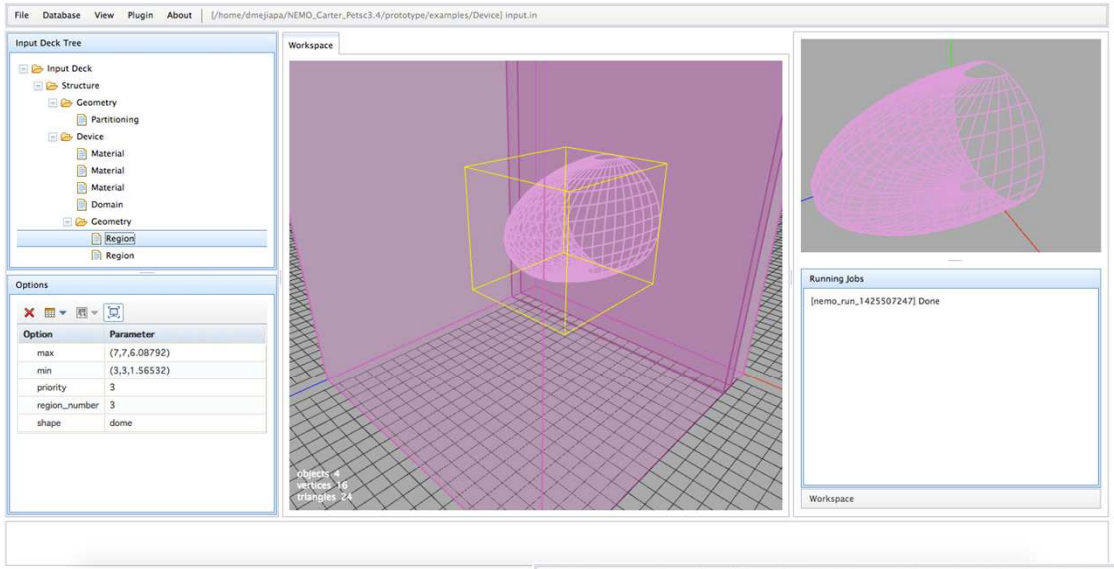


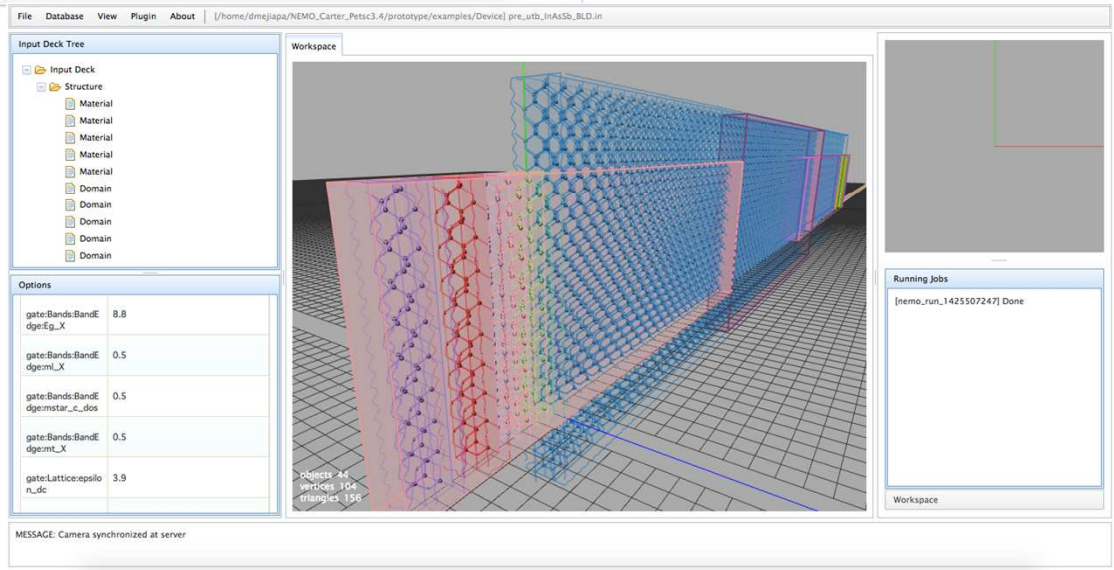


PURDUE UNIVERSITY

NSF

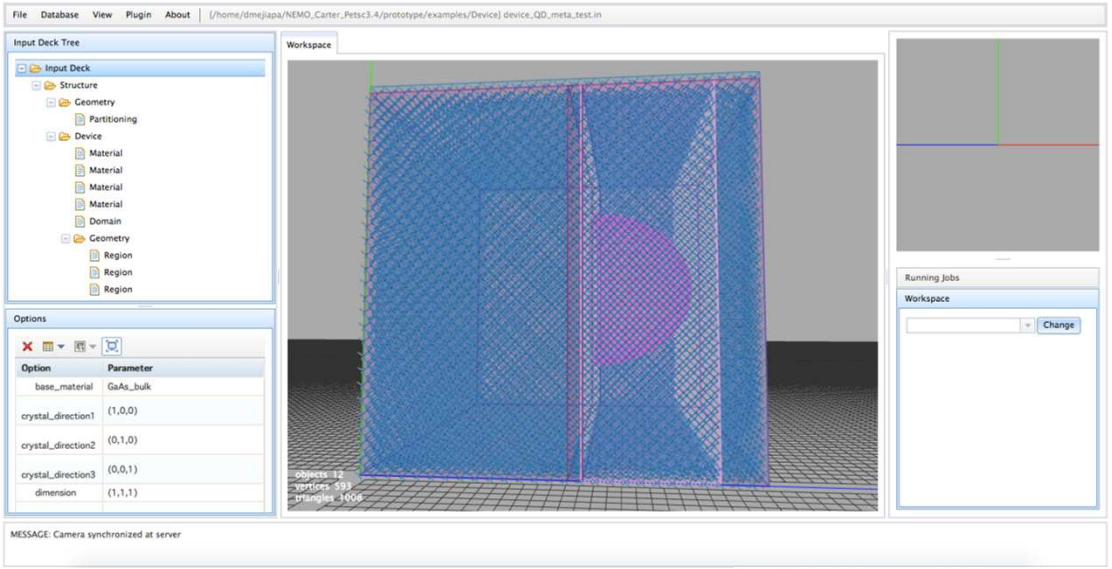
GALLERY V

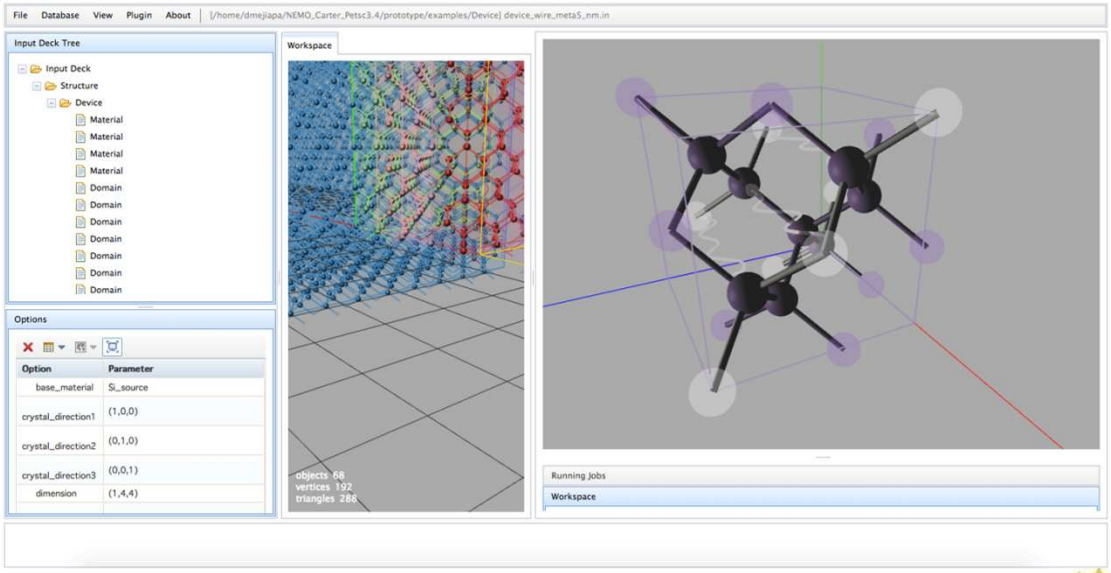




PURDUE UNIVERSITY

GALLERY VI





PURDUE UNIVERSITY

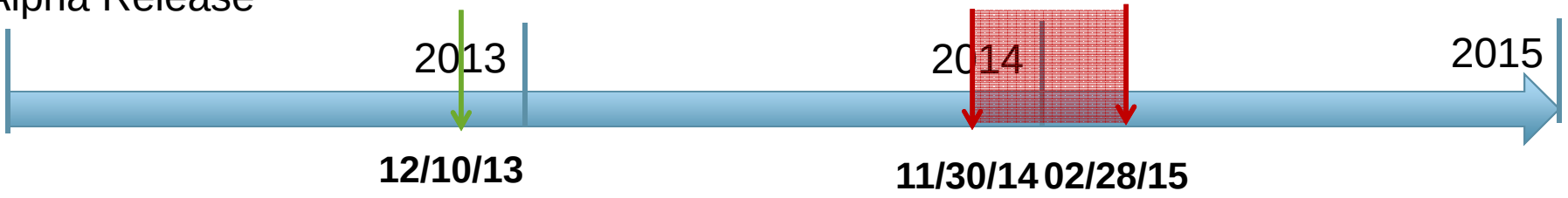
NSF

Nemo5 GUI Other Features

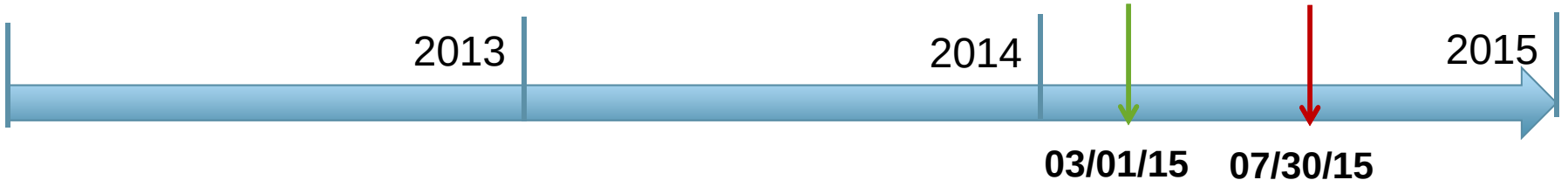
- Multiple users/session annotations
- Lightly Webserver implementation
- Support Templates
- Multi-threaded boost/thread:
- Nemo5 no thread-safe avoided with mutex boost/mutex and boost/lock
- Running lastest version of Three.js
- Error handling Json protocol

Nemo5 GUI - RoadMap

Alpha Release



Beta Release



Official Release V0.1

PURDUE UNIVERSITY

Alpha Release - TO-DOs

- Launcher non-blocking ☑
- Launcher prefixed folder names ☑
- Change Workspace directory ☑
- BUGs BUGs and more Bugs
- Migrate Nemo Scripts to error handling infrastructure ☑

How to use it

- COMPILE
  - » make server METHOD=XXX –j N
- RUN
  - » ./bin/nemo_server [URL PORT THREADS DOC_ROOT WORKSPACE_PATH LAUNCHER_PATH]
- ENJOY
  - » http://<<PURDUECLUSTER.rcac.purdue.edu>>:<<PORT>>/index.htm
    - ✓ **DEBUG:** default port : **3000**, default number of listening threads: **1 (**2 threads total**)**
    - ✓ **RELEASE:** default port : **Random Available**, default number of listening threads: **4 (**5 threads total**)**

summary

- We have presented a developing version of NemoServer and NemoClient as a Graphical user interface ('GUI') for Nemo5.

- Main features:
  - ✓ Only Libraries required for Nemo5 are required, No more.
  - ✓ 3D Visualization/Interaction of the device structure
  - ✓ Dynamic context layout;
  - ✓ Nemo5's auto-generated documentation is visualized
  - ✓ Database explorer
  - ✓ Resources configuration and Launcher
  - ✓ Plugins

- Future work:
  - ✓ Add solver's dependencies visualization / modifications
  - ✓ Add realtime resources visualization
  - ✓ Entities Wizards.
  - ✓ Export rapture tools
  - ✓ More + any suggestion from you….
  - ✓ …

Questions?