SAND2018-2551PE

Emerging Technologies and Productivity in HPC

Rob Hoekstra

PSAAP III Pre-Proposal Conference, March 14, 2017

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Complexity UP, Productivity DOWN

- HW is more complex
- Software stack is more complex
- Programming Environment/Model is more complex
- Execution/Operations Environment is more complex
- All these factors can negatively impact PRODUCTIVITY

Productivity has been declining rapidly in the HPC environment

- Dramatic increase in complexity of algorithms and applications coupled with a dramatic increase in complexity, diversity and scale of HW and execution environments

- AND CS/CSE research on productivity pays little attention to our HPC-specific problems (there are counter-examples such as IDEAS)

Even worse for our Mission Codes

- Complexity, size and dependencies of our codes is well above average even in the HPC community

- Verification/validation requirements create a much higher bar for incorporation of new capability whether it be physics, algorithms or performance optimization

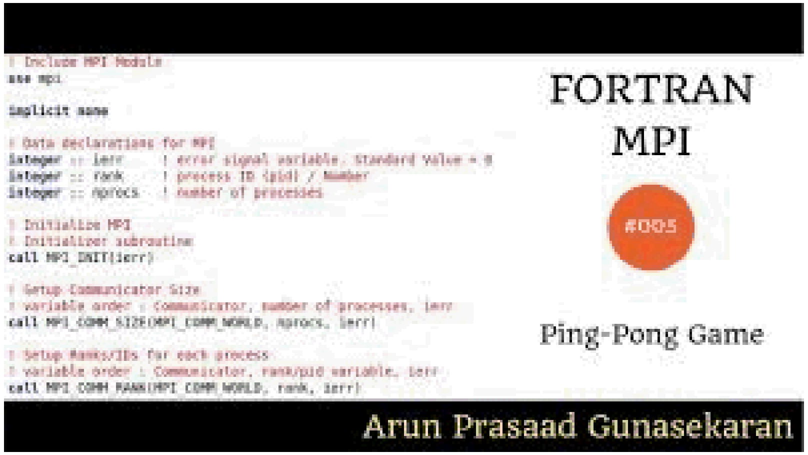
- And to make it worse, leadership-class platforms environments (SW stack, etc.) are often be more immature/fragile than average

Bottlenecks

- Code development
- Code correctness/testing
- Platform specific tuning/optimization
- Problem setup
- Job Execution & Steering
- Analysis & Viz

Code Development

- MPI/Fortran code



```
! Include MPI Module
use mpi

implicit none

! Data declarations for MPI
integer :: ierr      ! error signal variable, Standard Value = 0
integer :: rank      ! process ID (pid) / Number
integer :: nprocs    ! number of processes

! Initialize MPI
! Initializer subroutine
call MPI_INIT(ierr)

! Setup Communicator Size
! variable order : Communicator, number of processes, ierr
call MPI_COMM_SIZE(MPI_COMM_WORLD, nprocs, ierr)

! Setup Ranks/IDs for each process
! variable order : Communicator, rank/pid variable, ierr
call MPI_COMM_RANK(MPI_COMM_WORLD, rank, ierr)
```

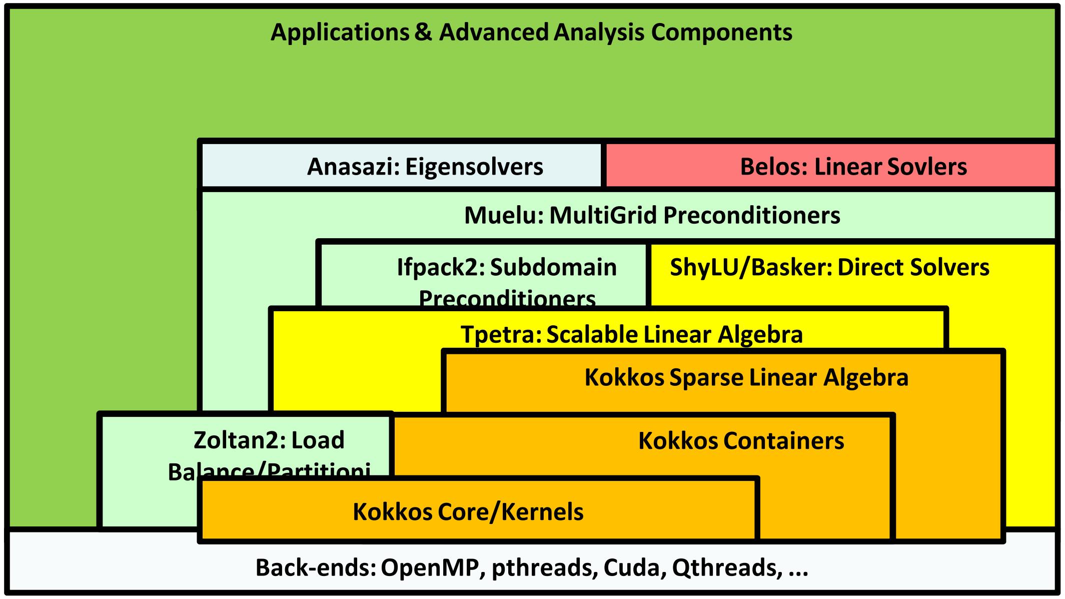
FORTRAN
MPI

#005

Ping-Pong Game

Arun Prasaad Gunasekaran

- C++, hierarchical parallel constructs, layered dependencies



Applications & Advanced Analysis Components

Anasazi: Eigensolvers

Belos: Linear Sovlers

Muelu: MultiGrid Preconditioners

Ifpack2: Subdomain Preconditioners

ShyLU/Basker: Direct Solvers

Tpetra: Scalable Linear Algebra

Kokkos Sparse Linear Algebra

Zoltan2: Load Balance/Partitioni

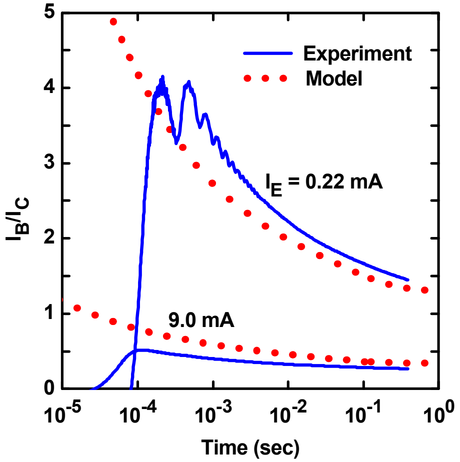
Kokkos Containers

Kokkos Core/Kernels

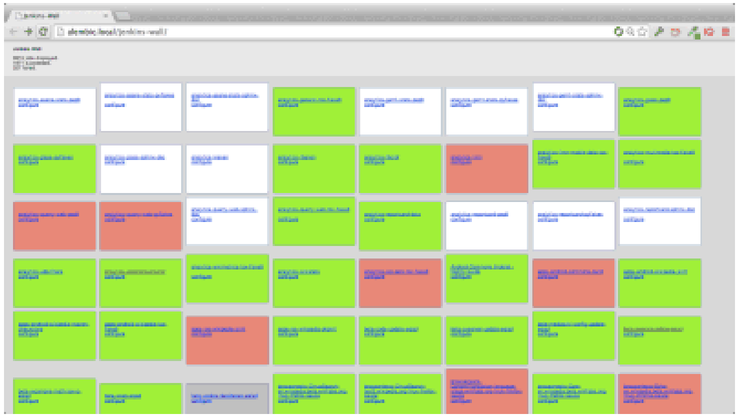
Back-ends: OpenMP, pthreads, Cuda, Qthreads, ...

Testing/Verification

- “Eyeball” Norm



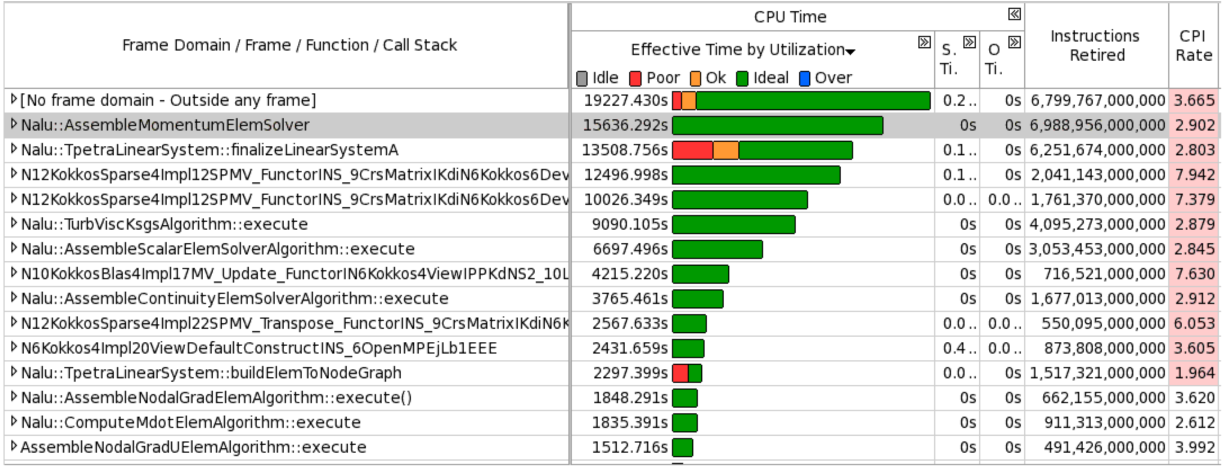
- Large verification test suites, non-reproducibility, etc.



Performance tuning/optimization

- PRINTF (still fall back to this many times:)

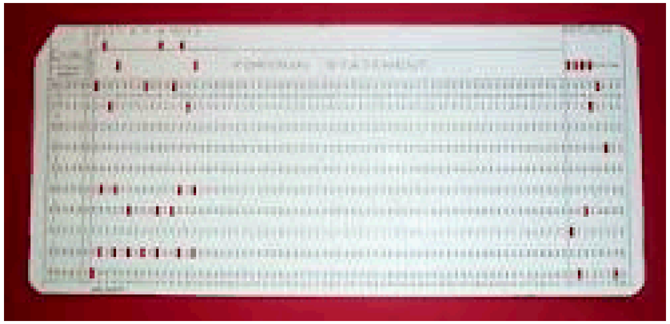
- Performance analysis and “divination”



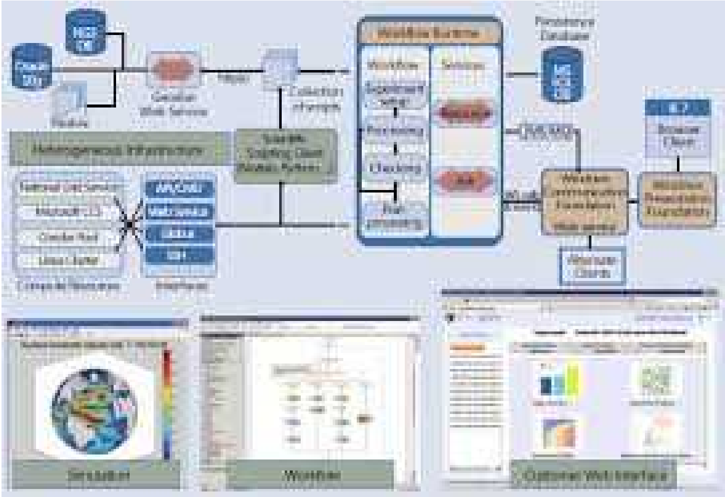
Frame Domain / Frame / Function / Call Stack	CPU Time			Instructions Retired	CPI Rate
	Effective Time by Utilization▾ ▨ Idle 🟥 Poor 🟧 Ok 🟩 Ideal 🟦 Over	S. Ti.	O Ti.		
[No frame domain - Outside any frame]	19227.430s	0.2 ..	0s	6,799,767,000,000	3.665
Nalu::AssembleMomentumElemSolver	15636.292s	0s	0s	6,988,956,000,000	2.902
Nalu::TpetraLinearSystem::finalizeLinearSystemA	13508.756s	0.1 ..	0s	6,251,674,000,000	2.803
N12KokkosSparse4Impl12SPMV_FunctorINS_9CrsMatrixIKdiN6Kokkos6Dev	12496.998s	0.1 ..	0s	2,041,143,000,000	7.942
N12KokkosSparse4Impl12SPMV_FunctorINS_9CrsMatrixIKdiN6Kokkos6Dev	10026.349s	0.0 ..	0.0 ..	1,761,370,000,000	7.379
Nalu::TurbViscKsgsAlgorithm::execute	9090.105s	0s	0s	4,095,273,000,000	2.879
Nalu::AssembleScalarElemSolverAlgorithm::execute	6697.496s	0s	0s	3,053,453,000,000	2.845
N10KokkosBlas4Impl17MV_Update_FunctorIN6Kokkos4ViewIPPKdNS2_10L	4215.220s	0s	0s	716,521,000,000	7.630
Nalu::AssembleContinuityElemSolverAlgorithm::execute	3765.461s	0s	0s	1,677,013,000,000	2.912
N12KokkosSparse4Impl22SPMV_Transpose_FunctorINS_9CrsMatrixIKdiN6K	2567.633s	0.0 ..	0.0 ..	550,095,000,000	6.053
N6Kokkos4Impl20ViewDefaultConstructINS_6OpenMPEjLb1EEE	2431.659s	0.4 ..	0.0 ..	873,808,000,000	3.605
Nalu::TpetraLinearSystem::buildElemToNodeGraph	2297.399s	0.0 ..	0s	1,517,321,000,000	1.964
Nalu::AssembleNodalGradElemAlgorithm::execute()	1848.291s	0s	0s	662,155,000,000	3.620
Nalu::ComputeMdotElemAlgorithm::execute	1835.391s	0s	0s	911,313,000,000	2.612
AssembleNodalGradUElemAlgorithm::execute	1512.716s	0s	0s	491,426,000,000	3.992

Problem Setup

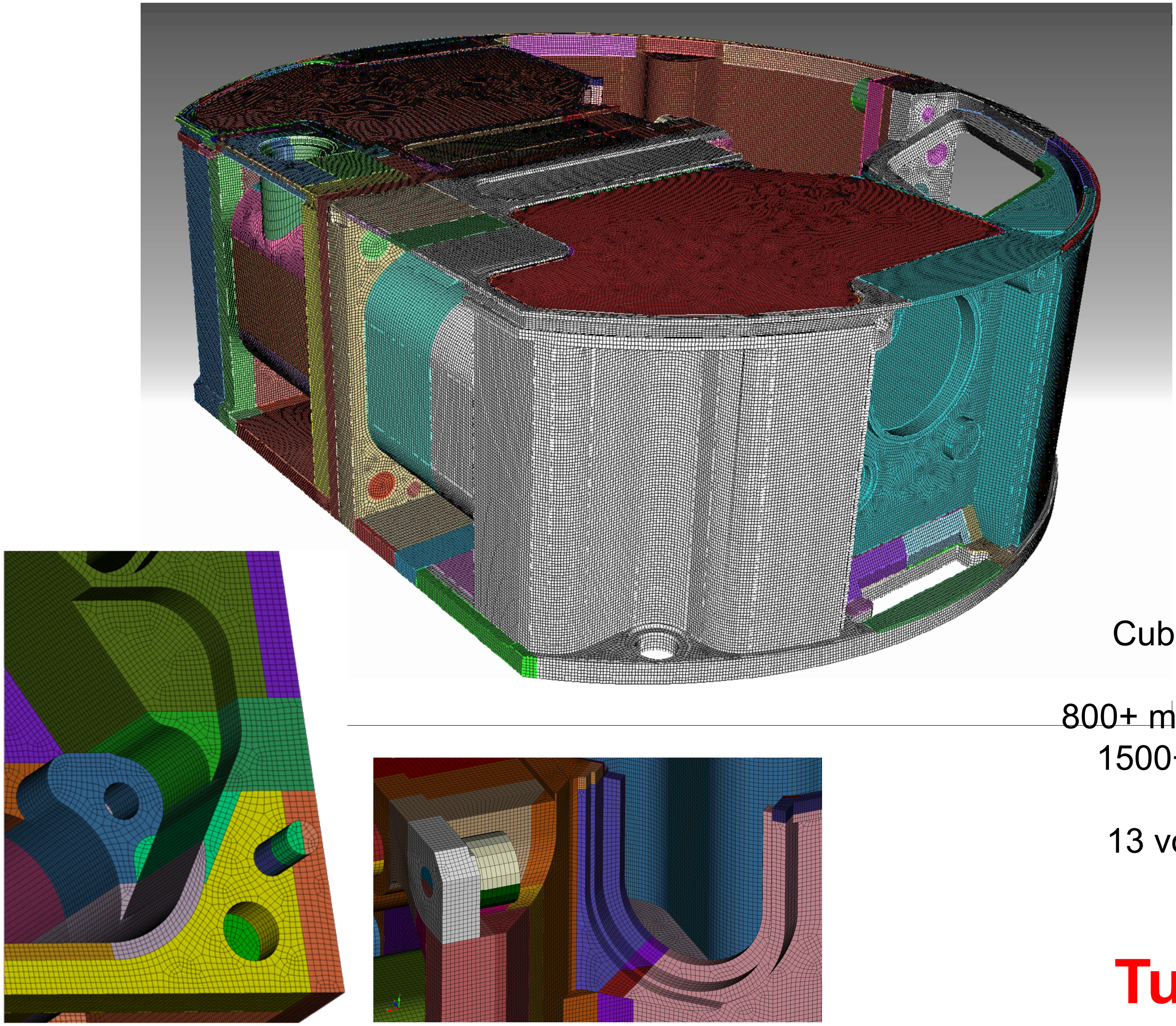
- Card Deck



- Complex workflow with geometry/meshing, etc.



Cubit Hex Meshing Capability



FCU housing geometry has 13 'volumes'

decomposed into meshable volumes

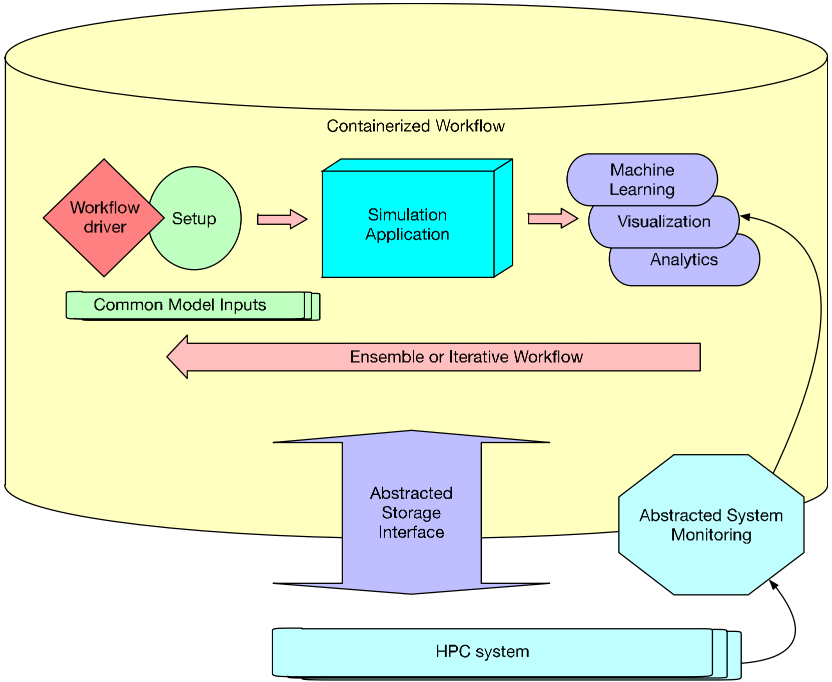
Cubit Journal file – 6200 lines long
Manually constructed
800+ manually specified webcuts defined
1500+ geometry cleanup commands
500+ meshing commands
13 volumes to 500 webcut volumes
1000+ hours of tedium

Turn around time: 9 months

Job execution/steering

- C:> Run app.exe

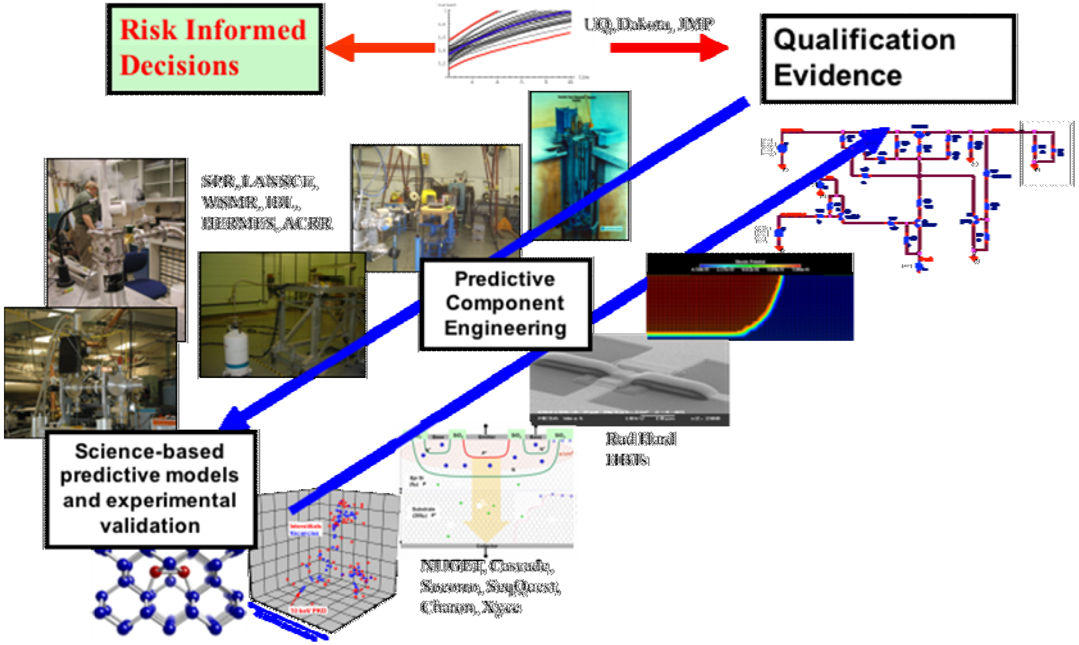
- Complex workflows of multi-physics, multiple codes, steering, data collection



Analysis/Viz

- Quantity = X

- Complex data flows/viz packages/UQ/validation



Risk Informed Decisions

UQ, Dakota, JMP

Qualification Evidence

SPR, LANSCE, WSMR, IBL, HERMES, ACRR

Predictive Component Engineering

Rad Hard HBTs

Science-based predictive models and experimental validation

NUGET, Cascade, Socorro, SeqQuest, Charon, Xyce

Areas of opportunity

- What is the future HPC “High Productivity” Programming Model?

- What is the future HPC “High Productivity” Development Environment?

- What is the future HPC “High Productivity” Runtime/Execution Environment?

- AND is there a more coherent unification of design time, compile time and runtime environments/tools?

Programming Models

- What is the future HPC “High Productivity” Programming Model?
  - Portability Abstractions
  - Async Multi-Tasking
  - DSLs
  - Component-based development

DARMA

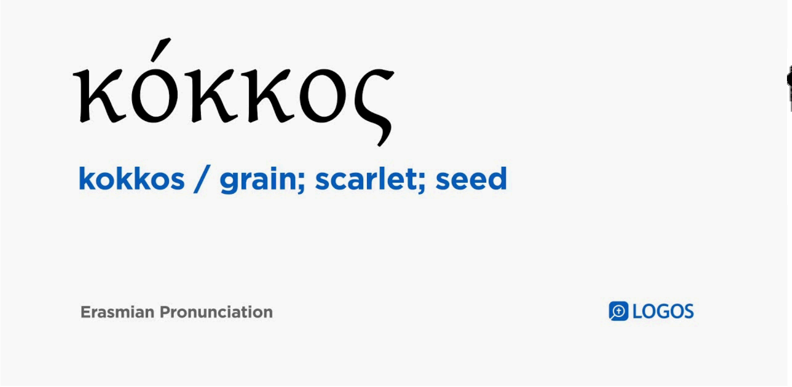
Uintah

Charm++

Legion

A Data-Centric Parallel Programming System

RAJA



κόκκος

kokkos / grain; scarlet; seed

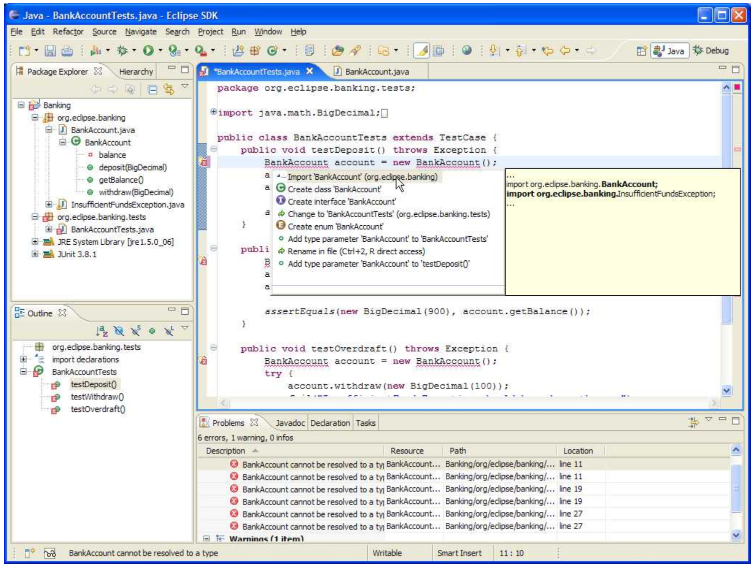
Erasmian Pronunciation

LOGOS

FleCSI

Development Environment

- What is the future HPC “High Productivity” Development Environment?
  - IDEs
  - Auto-tuning
  - Higher-level languages/scripting
  - Open compiler environments
  - Automated testing
  - CSE SW Engineering “Best Practices”



```
package org.eclipse.banking.tests;

import java.math.BigDecimal;

public class BankAccountTests extends TestCase {
    public void testDeposit() throws Exception {
        BankAccount account = new BankAccount();
        a
        a

        a
    }

    publi
        B
        a
        a

        assertEquals(new BigDecimal(900), account.getBalance());
    }

    public void testOverdraft() throws Exception {
        BankAccount account = new BankAccount();
        try {
            account.withdraw(new BigDecimal(100));
```

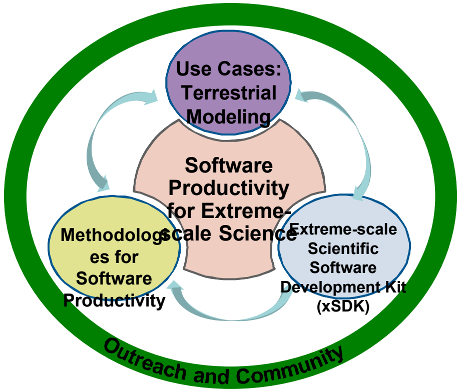
Import 'BankAccount' (org.eclipse.banking)
Create class 'BankAccount'
Create interface 'BankAccount'
Change to 'BankAccountTests' (org.eclipse.banking.tests)
Create enum 'BankAccount'
Add type parameter 'BankAccount' to 'BankAccountTests'
Rename in file (Ctrl+2, R direct access)
Add type parameter 'BankAccount' to 'testDeposit()'

...
import org.eclipse.banking.**BankAccount;**
import org.eclipse.banking. InsufficientFundsException;
...

6 errors, 1 warning, 0 infos

Description	Resource	Path	Location
BankAccount cannot be resolved to a ty|	BankAccount...	Banking/org/eclipse/banking/...	line 11
BankAccount cannot be resolved to a ty|	BankAccount...	Banking/org/eclipse/banking/...	line 11
BankAccount cannot be resolved to a ty|	BankAccount...	Banking/org/eclipse/banking/...	line 19
BankAccount cannot be resolved to a ty|	BankAccount...	Banking/org/eclipse/banking/...	line 19
BankAccount cannot be resolved to a ty|	BankAccount...	Banking/org/eclipse/banking/...	line 27
BankAccount cannot be resolved to a ty|	BankAccount...	Banking/org/eclipse/banking/...	line 27
Warnings (1 item)			

BankAccount cannot be resolved to a type | Writable | Smart Insert | 11 : 10


Use Cases: Terrestrial Modeling
Software Productivity for Extreme-scale Science
Methodologies for Software Productivity
Extreme-scale Scientific Software Development Kit (xSDK)
Outreach and Community

IDEAS productivity
www.ideas-productivity.org

PIs: Michael Heroux (SNL) and Lois Curfman McInnes (ANL)

Co-PIs: David Bernholdt (ORNL), Todd Gamblin (LLNL), Osni Marques (LBNL),
David Moulton (LANL), Boyana Norris (Univ of Oregon)

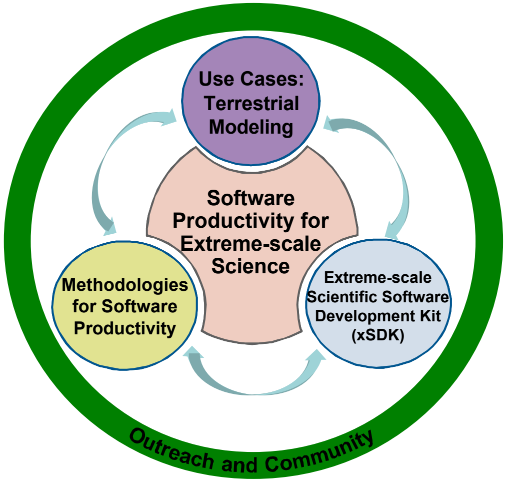
IDEAS: Interoperable Design of Extreme-scale Application Software

- Project began in Sept 2014 as ASCR/BER partnership to improve application software productivity, quality, and sustainability

Resources: https://ideas-productivity.org/resources, featuring

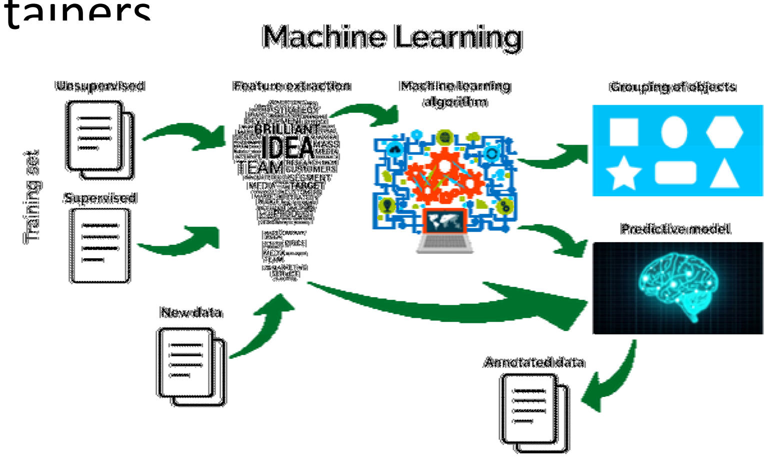
- ***WhatIs and HowTo* docs:** concise characterizations & best practices
  - *What is Software Configuration?*
  - *How to Configure Software*
  - *What is CSE Software Testing?*
  - *What is Version Control?*
  - *What is Good Documentation?*
  - *How to Write Good Documentation*
  - *How to Add and Improve Testing in a CSE Software Project*
  - *How to do Version Control with Git in your CSE Project*

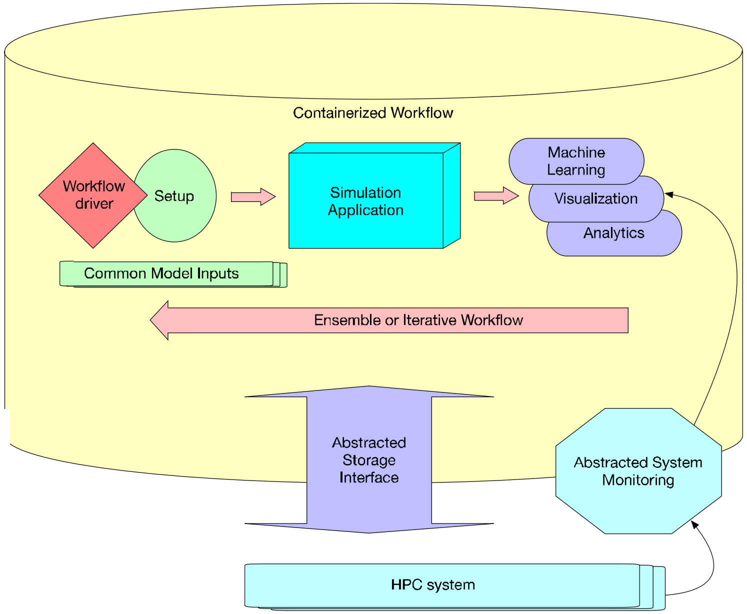
…. More under development


Use Cases: Terrestrial Modeling
Software Productivity for Extreme-scale Science
Methodologies for Software Productivity
Extreme-scale Scientific Software Development Kit (xSDK)
Outreach and Community

Runtime/Execution Environment

- What is the future HPC “High Productivity” Runtime/Execution Environment?
  - Workflows
  - Tasking
  - Machine Learning
  - Problem Setup
  - Containers





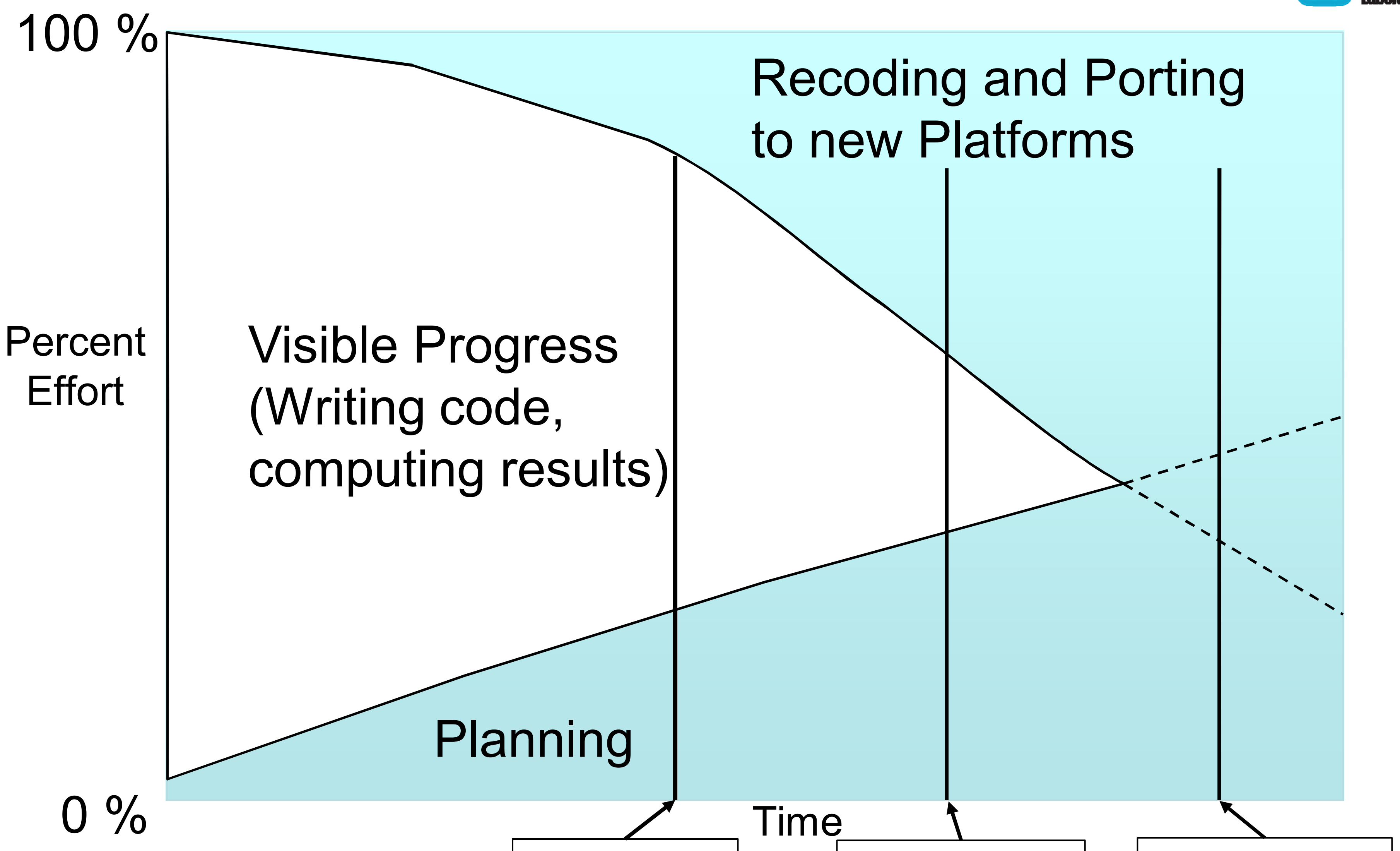
Productivity improvement as a common thread in center activities

- "Focus" on productivity enhancing technologies that are highly synergistic with other goals
  - Workflows
  - Programming Models/Environments
  - Machine Learning
  - Component-based Approaches

- Tell us how your center will leverage research in these areas will have a big positive impact on PRODUCTIVITY.

Questions?

Code-and-Fix Development Approach



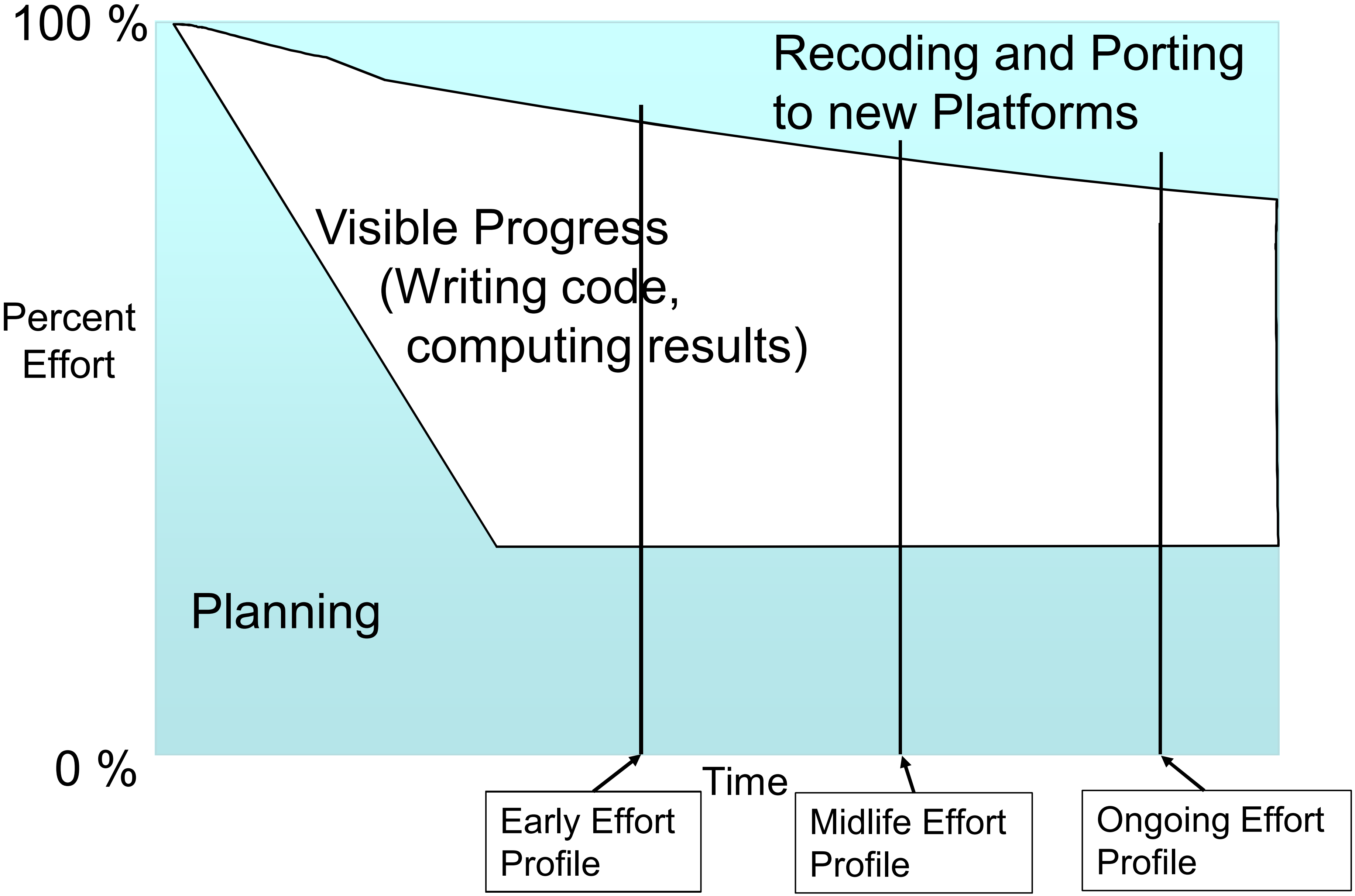
Early Effort Profile

Midlife Effort Profile

Endlife Effort Profile

Adapted from Software Project Survival Guide, Steve McConnell

Simple Planned Development Approach


100 %
Recoding and Porting to new Platforms
Visible Progress (Writing code, computing results)
Percent Effort
Planning
0 %
Time
Early Effort Profile
Midlife Effort Profile
Ongoing Effort Profile