E.T. International, Inc.

Pacific Northwest NATIONAL LABORATORY

Reservoir Labs

ILLINOIS UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Innovations in Programming Models, Compilers, and Runtime Systems for Dynamic Adaptive Event-Driven Execution Models

2012 X-Stack: Programming Challenges, Runtime Systems, and Tools

Brandywine Team

September 2012

E.T. International, Inc. Reservoir Labs ILLINOIS UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN Pacific Northwest NATIONAL LABORATORY

Principal Investigators

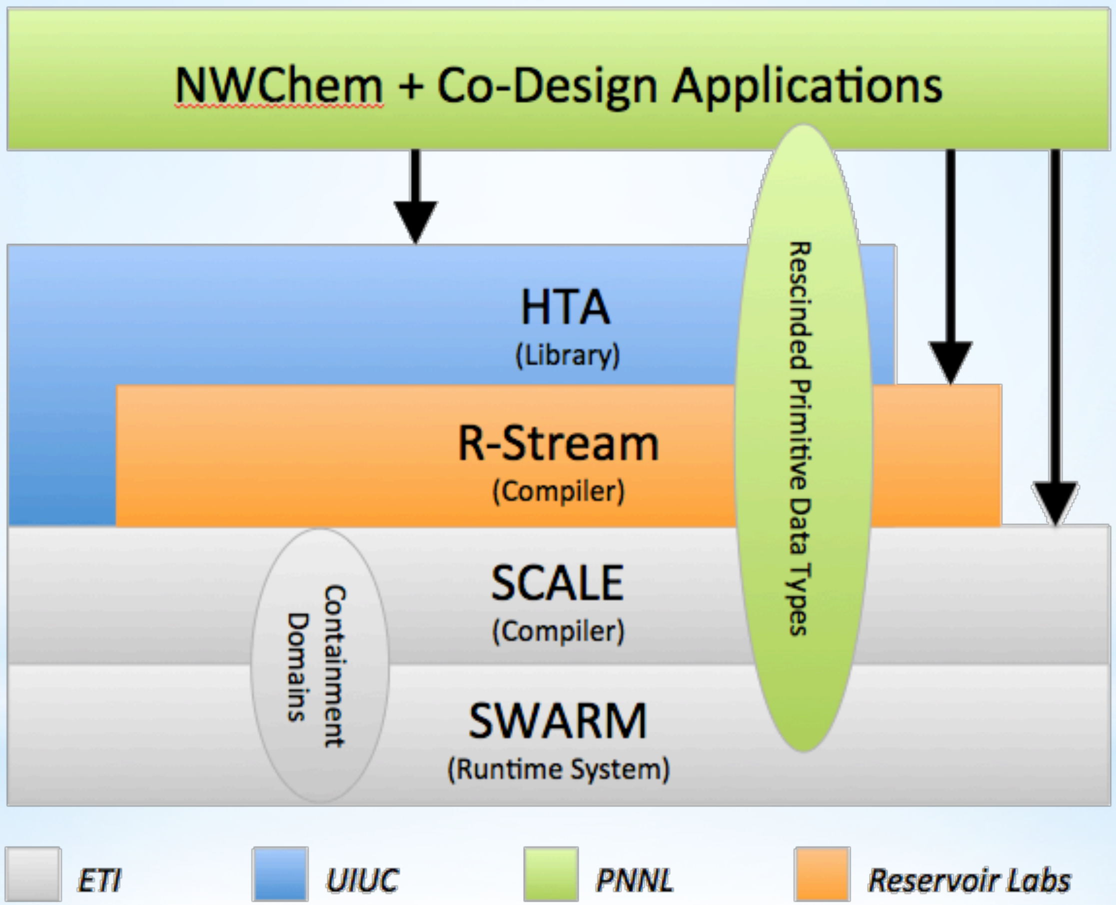
Rishi Khan (ET International)	Execution Model, Runtime Systems, Parallel Intermediate Language, Resilience
Benoit Meister (Reservoir Labs)	Programming Models, Loop Optimizations
David Padua (Univ. of Illinois)	High level data structures and algorithms for parallelism and locality
John Feo (PNNL)	Co-design and NWChem kernels for evaluation, energy efficiency

Objectives

Scalability	Expose, express, and exploit $O(10^{10})$ concurrency
Locality	Locality aware data types, algorithms, and optimizations
Programmability	Easy expression of asynchrony, concurrency, locality
Portability	Stack portability across heterogeneous architectures
Energy Efficiency	Maximize static and dynamic energy savings while managing the tradeoff between energy efficiency, resilience, and performance
Resilience	Gradual degradation in the face of many faults
Interoperability	Leverage legacy code through a gradual transformation towards exascale performance
Applications	Support NWChem

E.T. International, Inc.

Brandywine X-Stack Software Stack

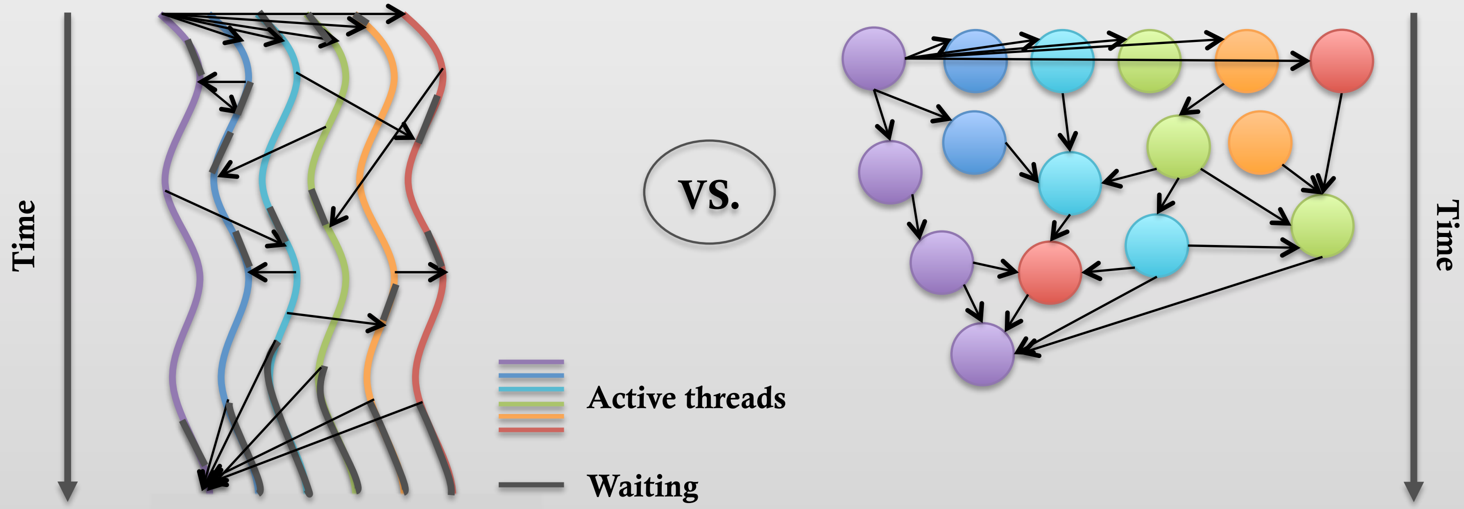


E.T. International, Inc. Reservoir Labs ILLINOIS UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN Pacific Northwest NATIONAL LABORATORY

SWARM

MPI, OpenMP, OpenCL

SWARM



- Communicating Turing Machines
- Bulk Synchronous
- Message Passing

- Asynchronous Event-Driven Tasks
- Dependencies
- Constraints
- Resources
- Active Messages

E.T. International, Inc.

Reservoir Labs

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Pacific Northwest
NATIONAL LABORATORY

SWARM

- Principles of Operation
  - Codelets
    - Basic unit of parallelism
    - Nonblocking tasks
    - Scheduled upon satisfaction of precedent constraints
  - Hierarchical Locale Tree: spatial position, data locality
  - Lightweight Synchronization
  - Active Global Address Space (planned)
- Dynamics
  - Asynchronous Split-phase Transactions: latency hiding
  - Message Driven Computation
  - Control-flow and Dataflow Futures
  - Error Handling
  - Fault Tolerance (planned)

E.T. International, Inc.

SCALE

- SCALE: SWARM Codelet Association LanguagE
  - Extends C99
  - Human readable parallel intermediate representation for concurrency, synchronization, and locality
  - Object model interface
  - Language constructs for expressing concurrency (codelets)
  - Language constructs to association codelets (procedures and initiators)
  - Object constructs for expressing synchronization (dependencies, barriers, and network registration)
  - Language constructs for expressing locality (planned)
- SCALECC: SCALE-to-C translator

Containment Domains

- Basic construct for resiliency
- Preserve -> Execute -> Validate -> Recover
- Hierarchical
  - Child can catch and recover from error or defer to parent
- Symmetric with try/catch error handling
- For X-Stack, SWARM/SCALE will support programmer-directed containment domains
- Dependency and locality information could be used for automatic preservation/ recovery steps (out of scope)

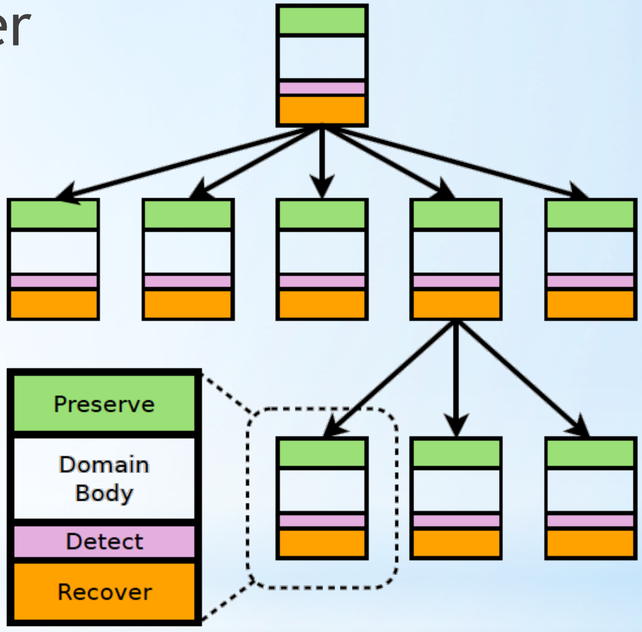


*Image from Containment Domains: a Full System Approach to Computational Resiliency, University of Texas

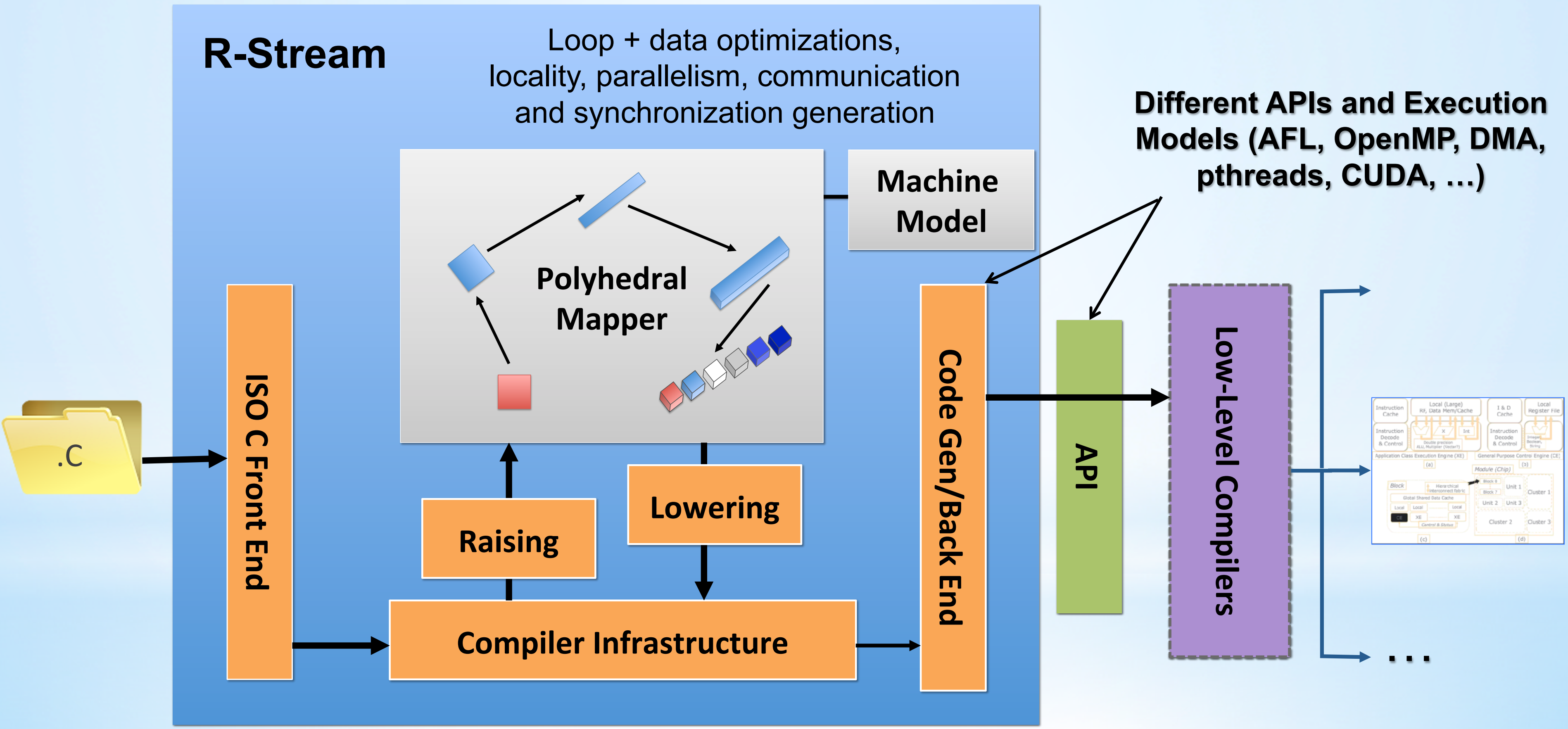
E.T. International, Inc.

Reservoir Labs

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Pacific Northwest
NATIONAL LABORATORY

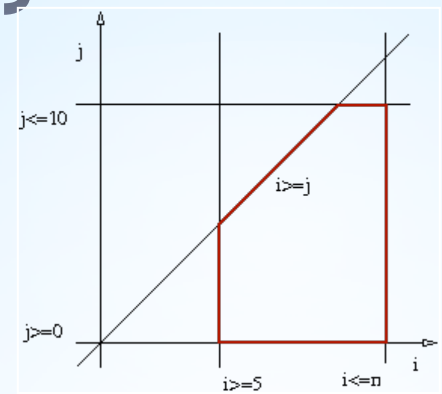
R-Stream Compiler

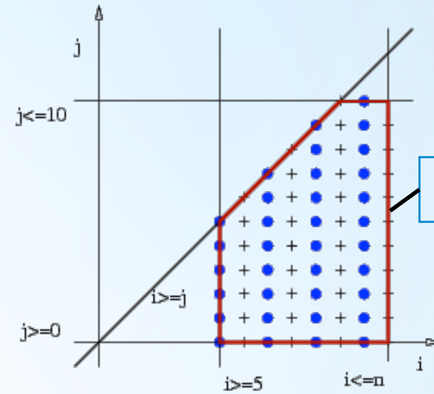


E.T. International, Inc. · Reservoir Labs · ILLINOIS (University of Illinois at Urbana-Champaign) · Pacific Northwest National Laboratory

Polyhedral Model

```
n = f();
for (i=5; i<= n; i+=2) {
  for (j=0; j<=i; i++) {
    if (j<=10) {
      … A[i+2j+n][i+3]…
  }
}
```





Variables and access functions as matrices

$$\begin{bmatrix} 1 & 2 & 1 & 0 \\ 1 & 0 & 0 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ n \\ 1 \end{bmatrix}$$

Affine schedules determine the execution order and place

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ M \\ N \\ 1 \end{bmatrix}$$

Dependence relations as polyhedra tie these components together

R-Stream: Current Capabilities

- Automatic parallelization and mapping
- Heterogeneous, hierarchical targets
- Automatic DMA/communications generation/ optimization
- Auto-tuning
- Scheduling with parallelism-locality-contiguity-data layout tradeoffs
- Corrective array expansion

R-Stream: Planned Capabilities

- Generate SCALE parallel codelet code from sequential programs
  - Extend thread generation techniques following Baskaran et. al. PPoPP'09 "Compiler Assisted Dynamic Scheduling" to generate codelets, extending for explicit data placement
  - Generate SCALE intermediate representation and hints for scheduling and data placement affinities
- Automatic optimization of irregular (sparse, mesh-based codes)

High Level Programming Notations

- Develop libraries in SCALE to enable the programming of codelets in the familiar notation of C/C++
- The libraries will represent parallelism using one or both of
  - Operations on arrays and sets
  - Parallel constructs such as parallel loops

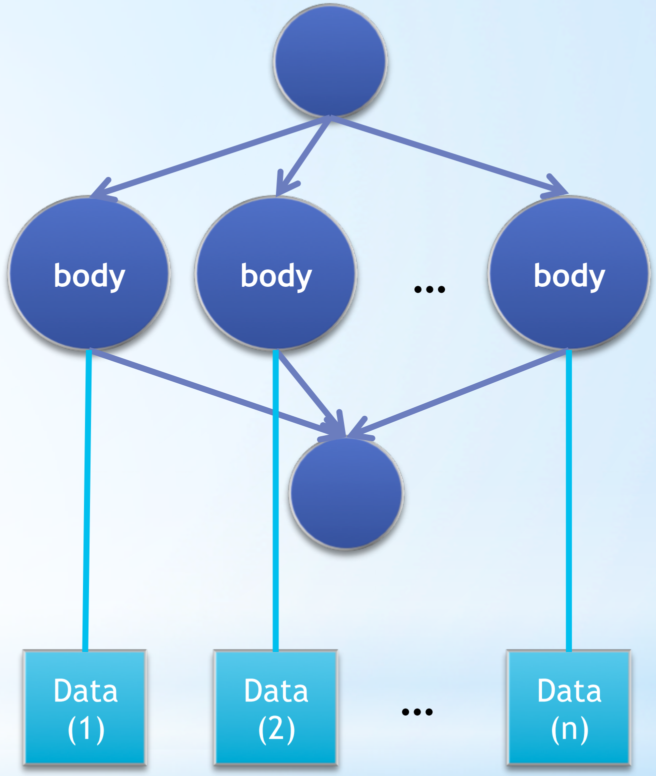
E.T. International, Inc. · Reservoir Labs · ILLINOIS University of Illinois at Urbana-Champaign · Pacific Northwest National Laboratory

High Level Programming Notations

- The routine

  parallel_for (<range>, <body>,<data array>)

  would create a codelet sequence according to the diagram on the right.



- Operations such as

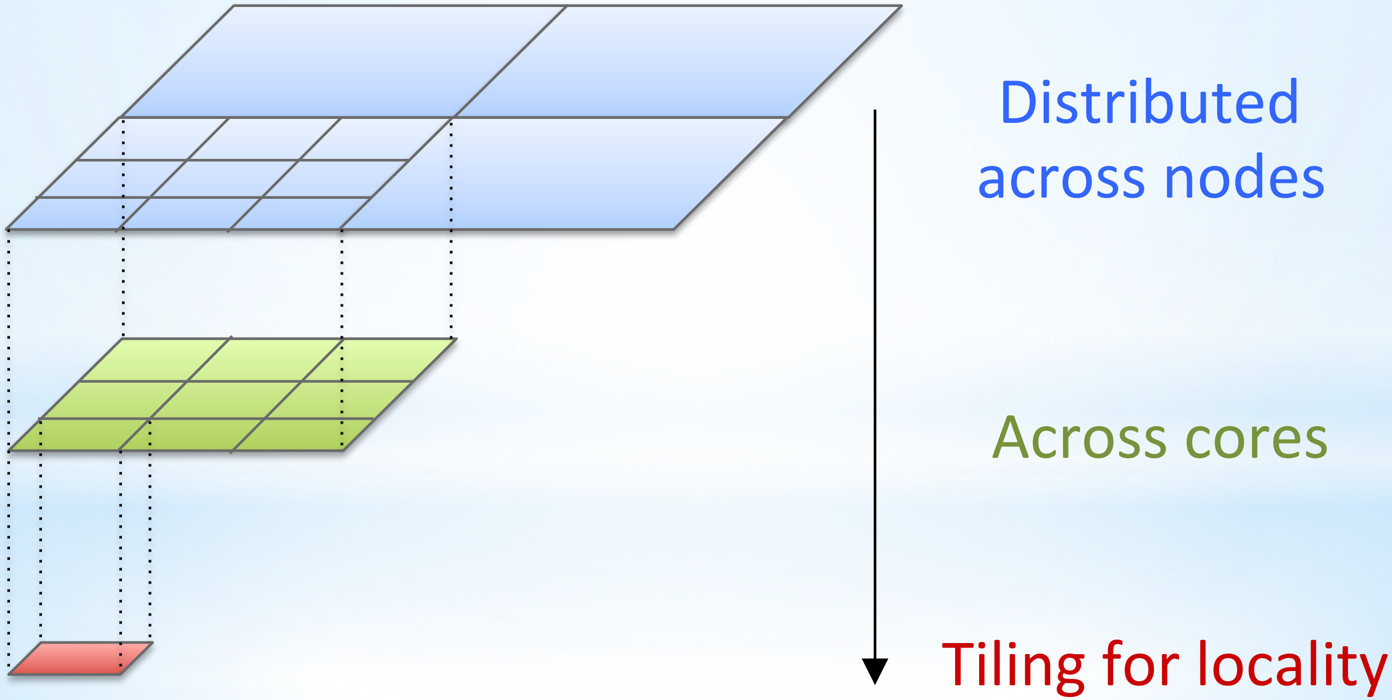
  A[:]=A[:]+1;

  would create similar codelet sequences with each Data Block representing a section (tile) of A and the body representing A[i]+=1;.

- Compiler optimizations such as fusion (which would eliminate unnecessary codelets) will be evaluated.

Hierarchical Tiled Arrays

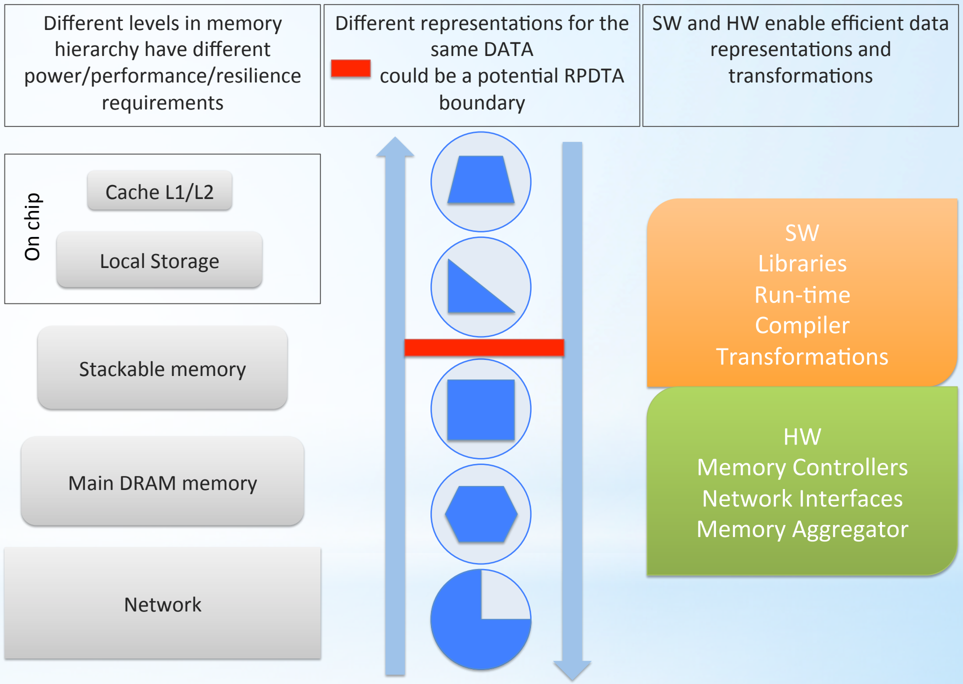
- Abstractions for parallelism and locality
  - Recursive data structure
  - Tree structured representation of memory



E.T. International, Inc. | Reservoir Labs | ILLINOIS UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN | Pacific Northwest NATIONAL LABORATORY

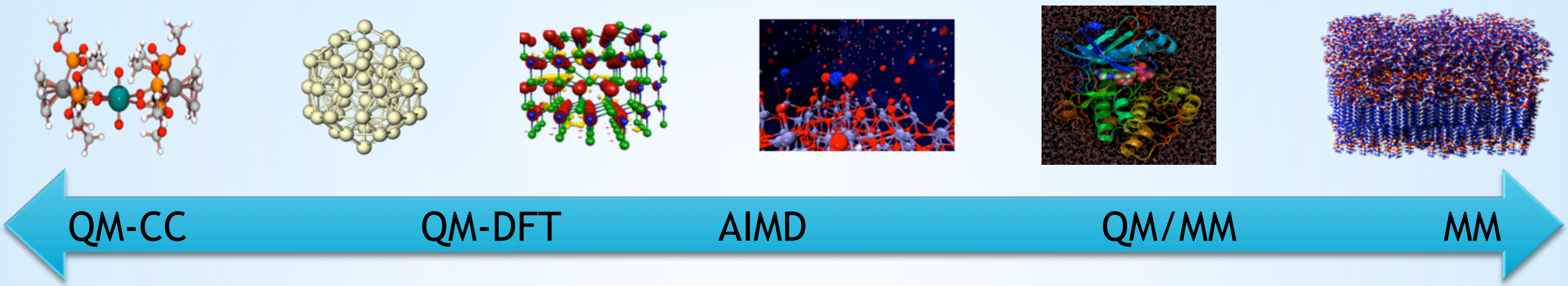
Rescinded Primitive Data Type Access

- Prevents actors (processors, accelerators, DMA) from accessing data structures as built-in data types, making these data structures opaque to the actors
- Redundancy removal to improve performance/energy
  - Communication
  - Storage
- Redundancy addition to improve fault tolerance
  - High Level fault tolerant error correction codes and their distributed placement
- Placeholder representation for aggregated data elements
  - Memory allocation/deallocation/copying
  - Memory consistency models



E.T. International, Inc.
Reservoir Labs
ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Pacific Northwest
NATIONAL LABORATORY

NWChem



- DOE’s Premier computational chemistry software
- One-of-a-kind solution scalable with respect to scientific challenge and compute platforms
- From molecules and nanoparticles to solid state and biomolecular systems
- Distributed under Educational Community License
- Open-source has greatly expanded user and developer base
- Worldwide distribution (70% is academia)

Co-design process for evaluation of stack

1. Accept co-design kernels with containment domains
2. Add rescinded primitive data types for energy efficiency
3. Add hierarchical tiled arrays where appropriate
4. Use R-Stream for loop optimization where appropriate
5. Convert the remaining portions to SCALE
6. Evaluate performance on x86 clusters and Runnemede simulator
7. Iterate

E.T. International, Inc. Reservoir Labs ILLINOIS UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN Pacific Northwest NATIONAL LABORATORY

3 Year Roadmap

	Year 1				Year 2				Year 3			
Task	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
T1.1	ETI	ETI	ETI	ETI								
T1.2	ETI	ETI	ETI	ETI								
T2.1					ETI	ETI						
T2.2					ETI	ETI						
T2.3							ETI	ETI				
T2.4			Reservoir				ETI	ETI				
T3.1	Reservoir	Reservoir										
T3.2		Reservoir	Reservoir									
T3.3				Reservoir								
T3.4					Reservoir	Reservoir	Reservoir					
T4.1	UIUC	UIUC	UIUC	UIUC								
T4.2					UIUC	UIUC	UIUC	UIUC				
T5.1	UIUC	UIUC										
T5.2			UIUC	UIUC	UIUC	UIUC						
T5.3							UIUC	UIUC	UIUC	UIUC	UIUC	UIUC

	Year 1				Year 2				Year 3			
Task	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
T5.7	UIUC	UIUC	UIUC	UIUC								
T5.8									Reservoir	Reservoir	Reservoir	
T5.9											Reservoir	
T6.1	PNNL											
T6.2		PNNL	PNNL									
T6.3		PNNL	PNNL	PNNL	PNNL							
T6.4				PNNL	PNNL	PNNL	PNNL					
T6.5				PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	
T6.6						PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL
T7.1	ETI	ETI	ETI	ETI	ETI	ETI	ETI	ETI				
T8.1									ETI	ETI	ETI	ETI
T8.2									ETI	ETI	ETI	ETI
T9.1	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL
T9.2	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL	PNNL
T10.1									ETI	ETI	ETI	ETI

Lead Color Legend: ETI Reservoir UIUC PNNL

1. Runtime Scheduling
2. Runtime Data Locality
3. Loop Optimizations for Scheduling and Placement
4. Sparse Arrays and Irregular Tiles
5. High Level Notations and Optimizations
6. Co-design Apps
7. Portability
8. Resilience
9. Energy Efficiency
10. MPI Interoperability

Acknowledgements

- Co-PIs:
  - Benoit Meister (Reservoir)
  - David Padua (Univ. Illinois)
  - John Feo (PNNL)
- Other team members:
  - ETI: Robin Lawton, Chanika Denny
  - Reservoir: Rich Lethin
  - Univ. Illinois: Adam Smith
  - PNNL: Andres Marquez
- DOE
  - Sonia Sachs, Bill Harrod

E.T. International, Inc.

Reservoir Labs

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Pacific Northwest
NATIONAL LABORATORY

Acknowledgements