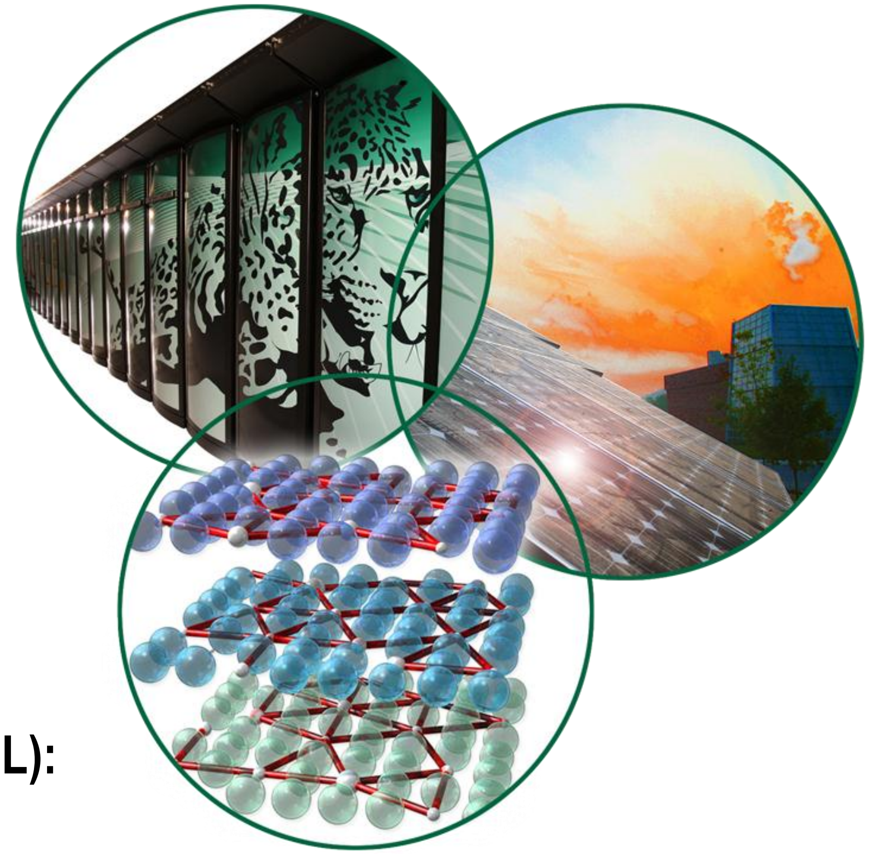
CAEBAT Open Architecture Software (OAS)

CAEBAT Kick-off Meeting

2 Aug. 2011
ORNL
Oak Ridge, TN



OAS Team (ORNL):

- John Turner
- Sreekanth Pannala
- Srikanth Allu
- Partha Mukherjee
- Wael Elwasif
- David Bernholdt

U.S. DEPARTMENT OF ENERGY

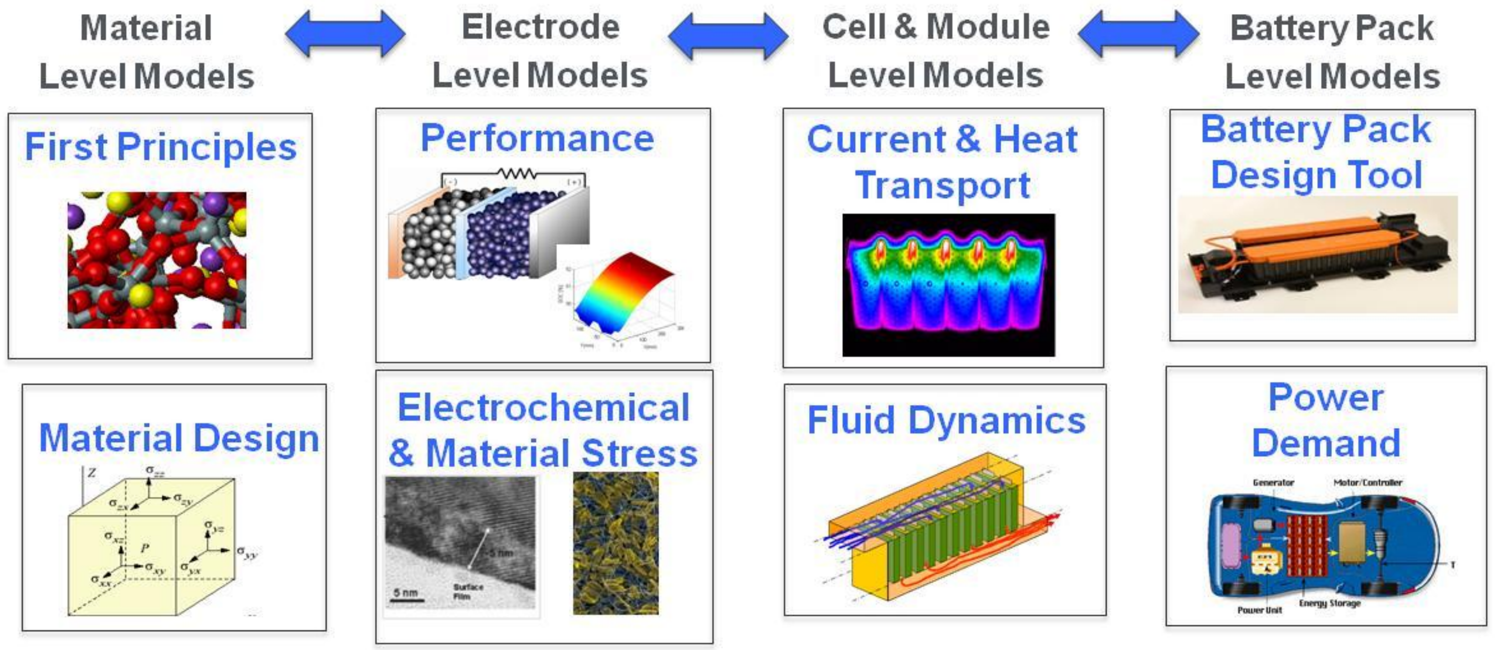
OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

The Need for a Focused Project on Battery CAE

- In previous years, under DOE funding, national laboratories, industry and universities have developed several tools for cost, life, performance electro-thermal, electrochemical, and abuse reaction modeling of lithium-ion batteries.
- The concern has been that these models were not all integrated and additional tools were needed.
- DOE has been evaluating approaches to integrate these battery modeling activities and make them more accessible as design tools for industry
- In April of 2010, the DOE VT Energy Storage Program initiated the multi-year CAEBAT project to
  - Develop battery design tools that could be used across many scales and many physics, and
  - Develop an open architecture software framework that would enable disparate models to interface with each other.

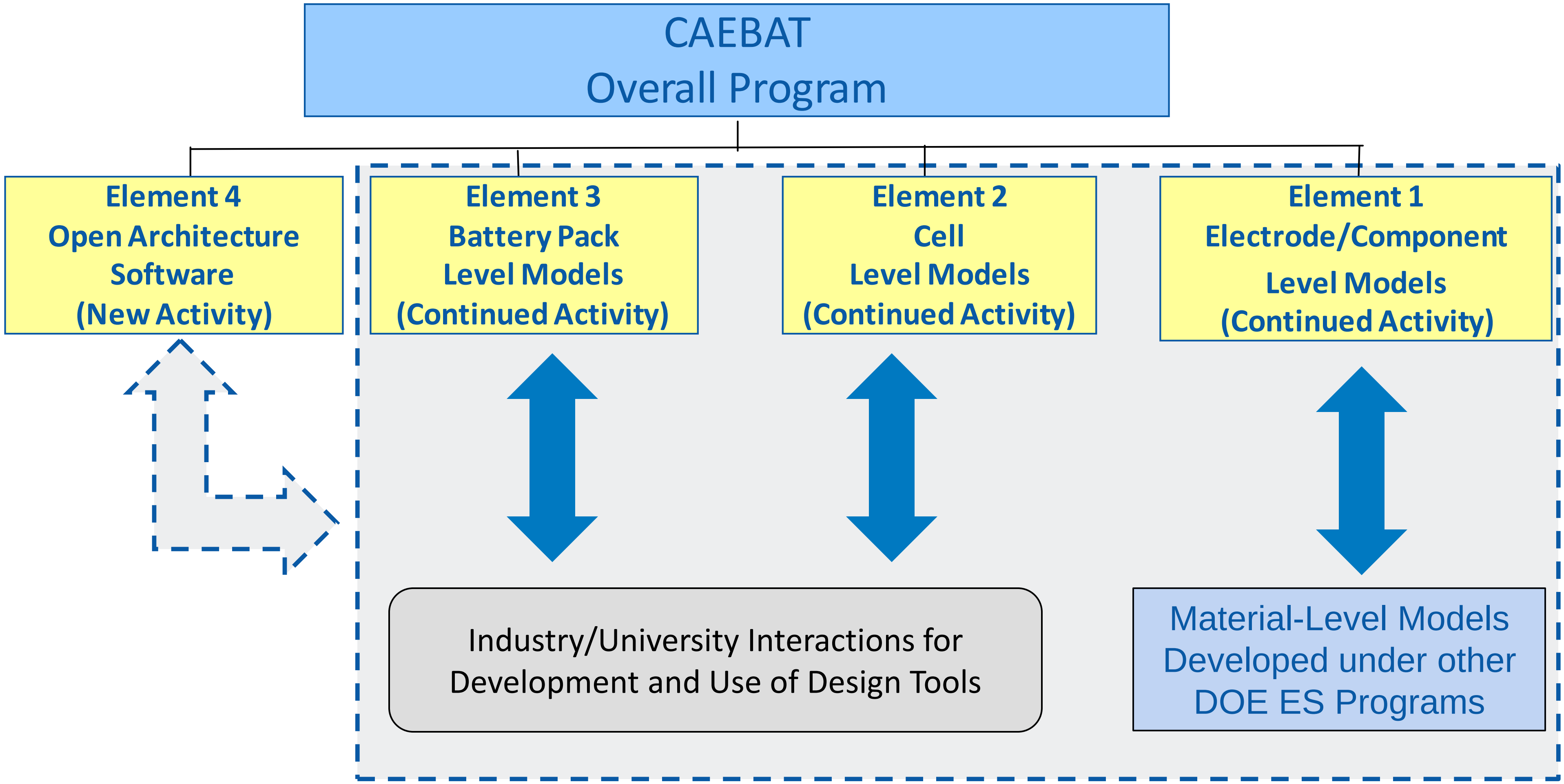
Objectives of the CAEBAT Project

- The objective of CAEBAT is to incorporate existing and new models into design suites/tools with the goal of shortening design cycles and optimizing batteries (cells and packs) for improved performance, safety, long life, and low cost.
- The software suites would include material properties, electrode design, pack design for thermal management purposes, load profiles, cost information, and aging data as input, and could greatly speed up the design of new batteries and provide critical guidance to developers.



Courtesy:Ahmad A. Pesaran, CAEBAT Coordinator

CAEBAT Project Structure



- National Labs will perform in-house battery modeling (existing or new)
- Coordination and exchange with organizations doing fundamental materials modeling
- Collaborations with industry through competitive solicitations
- Development of an interface platform for interactions among all models

CAEBAT Program Goals

- **ultimately, the CAEBAT program will deliver 4 simulation tools:**
  - one from each of the RFP teams
  - one based on an Open Architecture Software (OAS) infrastructure
    - we are calling this the Virtual Integrated Battery Environment (VIBE)
- **each will (ultimately) be fully capable**
  - RFP tools focused on delivering a cell and pack modeling tool for industry
  - OAS tool integrates modules from RFP teams as well as Lab and University efforts beyond the RFP teams
- **coordination and collaboration across teams will be critical to overall success of CAEBAT**
  - standardization of input
  - standardization of "battery state" database
  - standard test problem(s)
  - standardized interfaces for cell, pack, etc. models

Overarching goal: open architecture to integrate battery modeling components and aid battery design

- **Access to commercial and non-commercial software through standardized interfaces**
  - battery designer cannot be locked in to one particular vendor or software
  - ability to pick (and combine) the best software components available
  - standardize the design process
- **Access to latest numerical methods and algorithms**
  - rapidly advance the state of the art
  - provide the best software tools to the battery designer
- **Verified and Validated**

Next steps

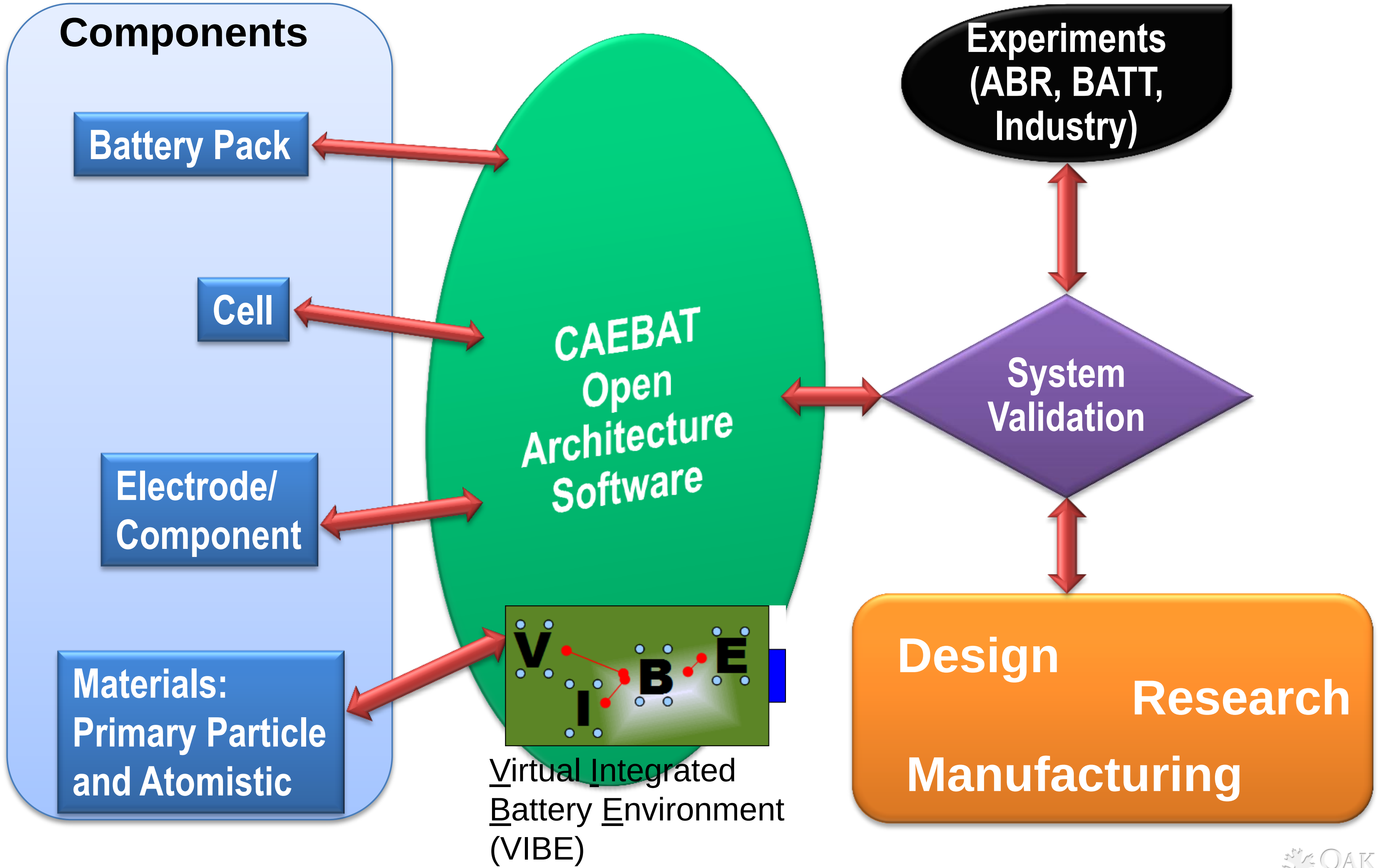
- **integrate additional components**
  - NREL's MSMD
  - RFP teams
  - other Labs and Universities
- **intellectual property agreements (where necessary)**
- **standardization of input, battery state and interfaces**
- **validation, sensitivity analysis (SA), uncertainty quantification (UQ)**
- **Windows porting**
  - we are talking to Microsoft's HPC server development team

Supplemental

Modeling and Simulation Terminology

- **Model**
  - mathematical representation of physical phenomena
- **Method**
  - numerical algorithms (discretization, solution methods, etc.)
- **Code / Component / Application**
  - software implementation of particular model or set of models
  - can be source code, compiled library, or executable
- **Simulation**
  - use of code to perform analysis / design
  - requires integration with experimental program
  - should provide information on data sensitivities and uncertainties

CAEBAT Open Architecture Software: vision


Components
Battery Pack
Cell
Electrode/ Component
Materials: Primary Particle and Atomistic
CAEBAT Open Architecture Software
V I B E
Virtual Integrated Battery Environment (VIBE)
Experiments (ABR, BATT, Industry)
System Validation
Design
Research
Manufacturing

CAEBAT OAS simulation platform has two aspects

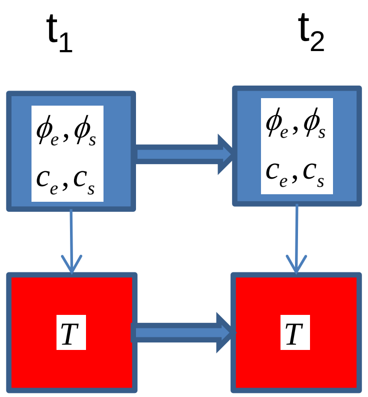
Software Infrastructure

- **flexible**
  - language-agnostic
  - multiple modeling approaches
  - combine appropriate component models for problem at hand
  - support integrated sensitivity analysis and uncertainty quantification
- **extensible**
  - ability to add and combine proprietary component models
- **scalable from desktop to HPC platforms**
  - hardware architecture-aware

Numerical coupling/Scale-bridging approach(es)

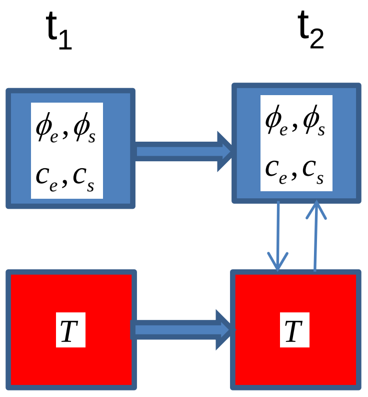
- **flexible coupling strategy**
- **ability to transfer information across different models in a mathematically / physically consistent fashion**
- **similarly for bridging time-scales**

Coupling scenarios in battery modeling

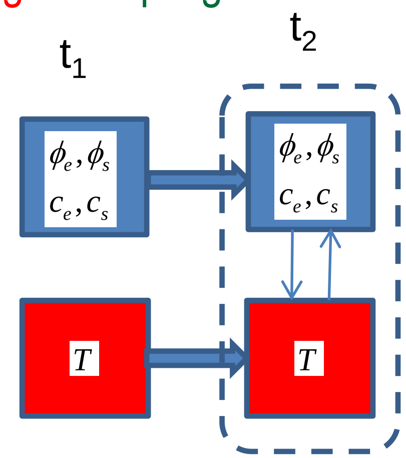


One-way Coupling

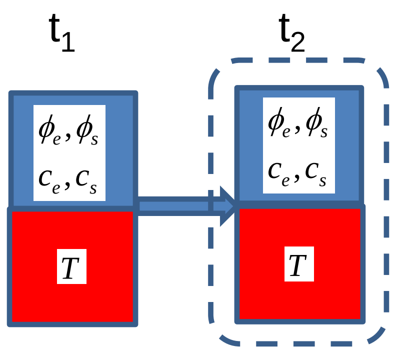
Two-way Loose Coupling



Two-way Tight Coupling



Picard
self-consistent iterations to some convergence criteria



Fully Implicit
Consistency at each iteration across the physics in terms of full non-linear residual

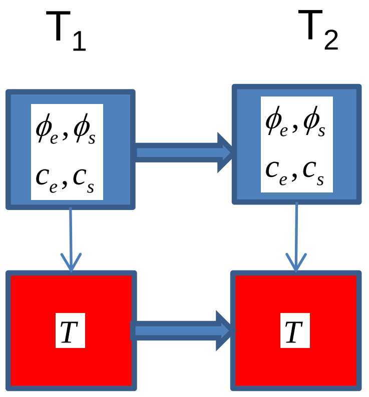
Loose Coupling

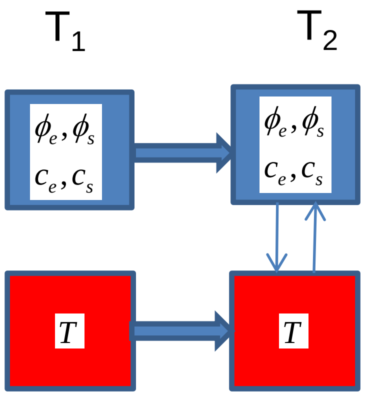
One-way coupling

- There is only one-way propagation of information
- Easy to implement
- Efficient for scenarios where the reverse coupling is weak

Explicit/Semi-implicit

- Each physics marches in time explicitly or implicitly on a component basis
- There is no reconciliation of field variables at each time-step
- Easy to implement and scale
- Depending on the nature of the coupling the answers might be wrong

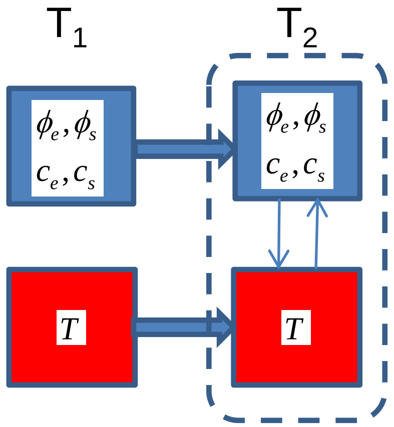




Tight (full) Coupling

Segregated (Picard) solve

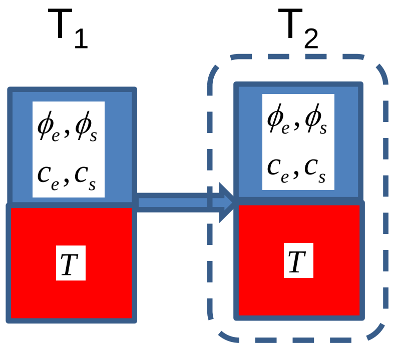
- Semi-implicit formulation
- Reconciliation of the error across the variables to certain tolerance
- Can lead to non-convergence or poor convergence (linear to flat)
- For certain cases, more efficient than fully implicitly coupled



Self-consistent iterations to some Convergence criteria

Fully implicit solve

- Fully implicit formulation
- More expensive per time-step
- Can take much larger time-steps
- Construction of a good preconditioner is key to good convergence (quadratic)



Consistency at each iteration across the physics in terms of full non-linear residual

VIBE builds on the Integrated Plasma Simulator (IPS) for Fusion
Background and Motivation

- **coupling of disparate codes in various languages**
- **allow for proprietary codes/components**
  - work with executables and open interfaces
  - however, the infrastructure itself is open and vendor neutral
- **heavily used, mature, long-lived codes**
  - occasional two-way coupling
- **different characteristics and capabilities**
  - parallelism, data format, execution work flow,..
- **no need to re-factor major existing or new codes**
- **assume codes WILL change during the project lifetime**
  - avoid forking and loss of new features.

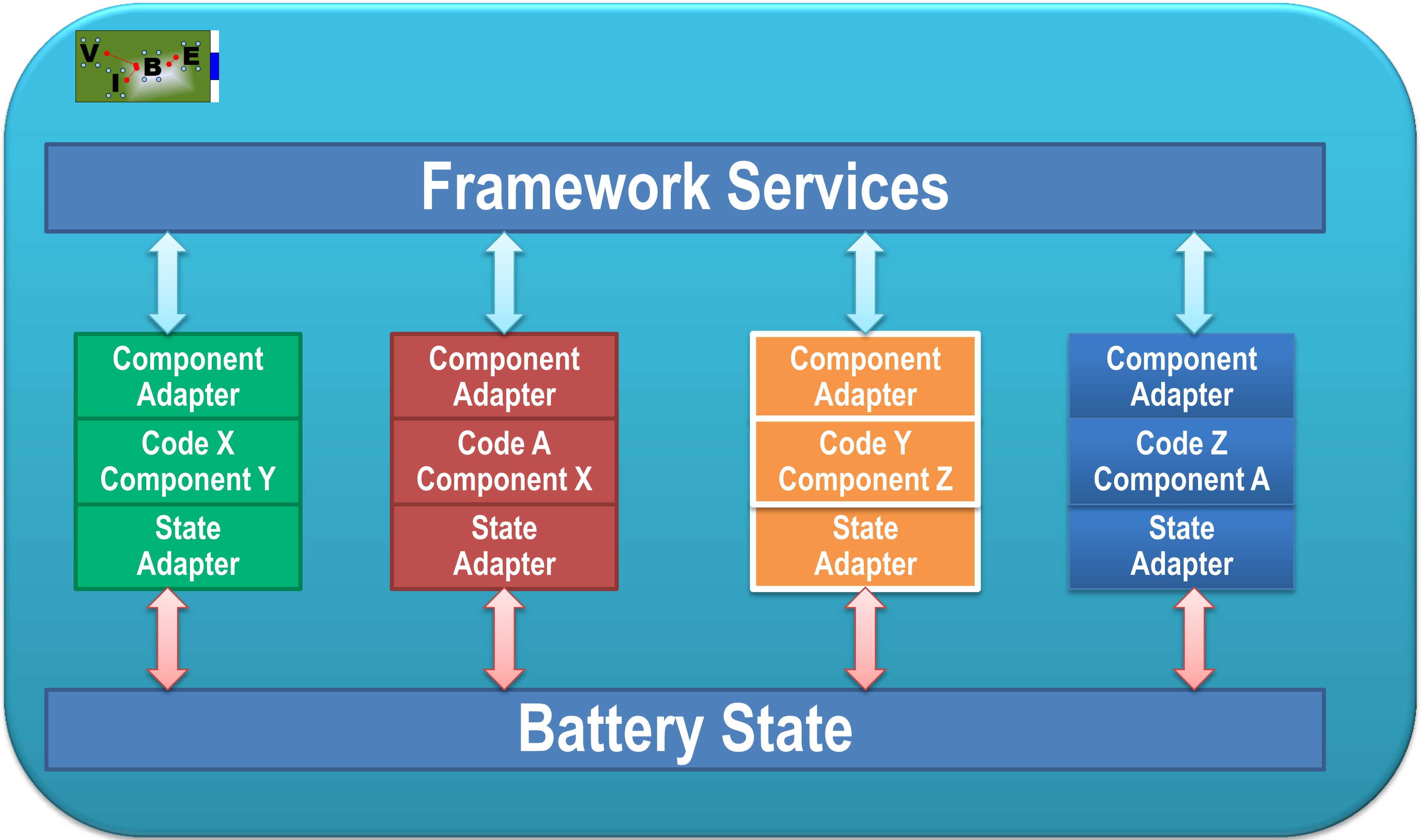
VIBE: Integration Philosophy & Approach for CAEBAT

- **Minimize level of effort to bring in physics components**
  - wrappers around unmodified components
  - use application-native I/O, transform to shared data using state adapters
- **Design for multiple implementations (possibly from different vendors) of each physics component**
  - well-defined component interfaces
  - accommodate reduced models, inter-comparisons (V&V), etc.

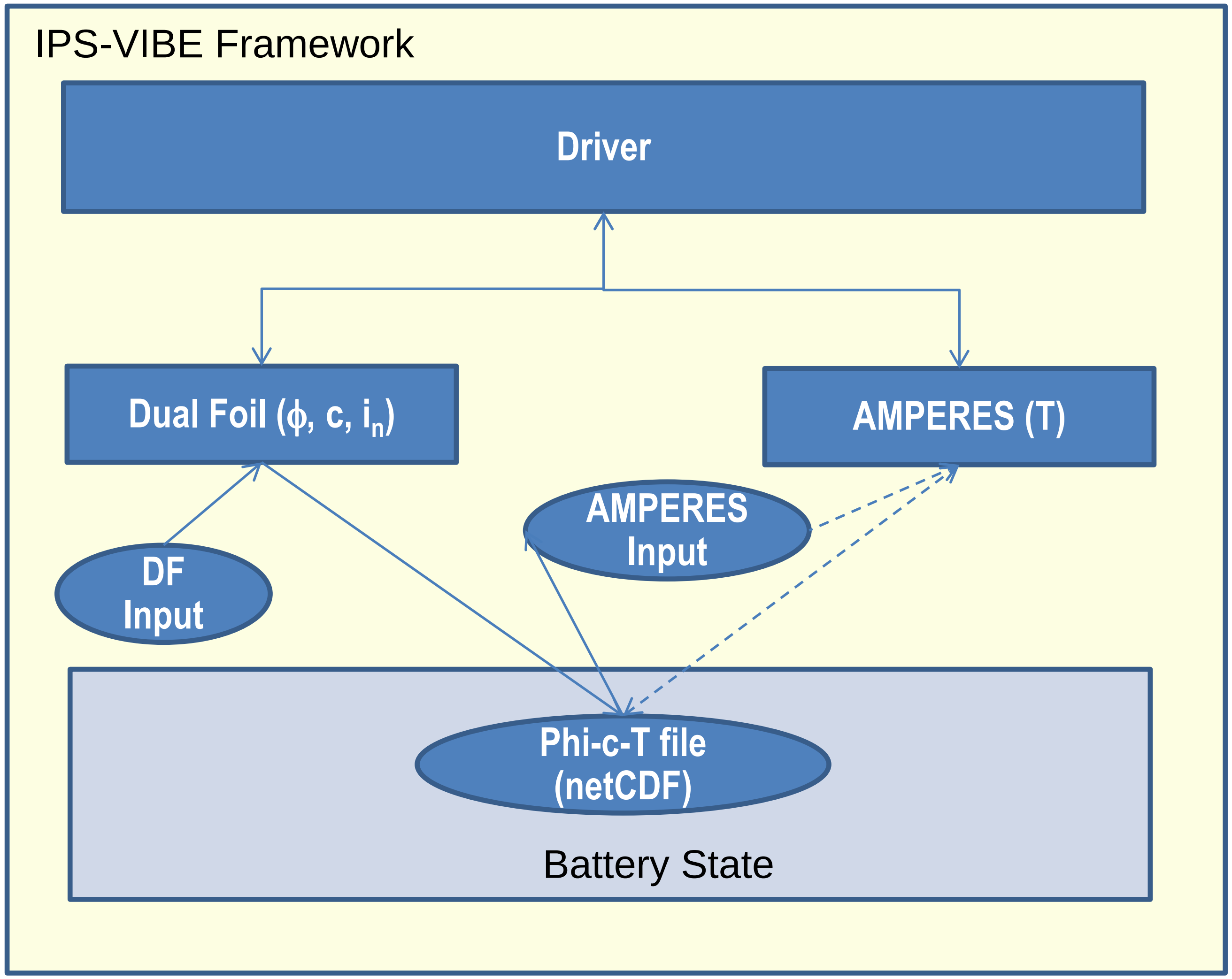
VIBE: Integration Philosophy & Approach (2)

- **Design for broad range of integrated simulation**
  - driven by battery design requirements, but extensible
  - target loose coupling initially, but with concepts that "scale" to stronger coupling
- **Component Approach**
  - based on Common Component Architecture (CCA) concepts
  - simplified implementation, focusing on concepts, key features
- **Ability to perform design optimization, SA, UQ**
  - links to DAKOTA and other optimization/UQ/sensitivity analysis tools
- **Light-weight**
  - easy to install and use, with low overhead

VIBE Software Platform for CAEBAT



VIBE Software Platform for CAEBAT (demo: Dualfoil/AMPERES)



Possible Scenarios Explored:

a) DualFoil for entire duration followed by AMPERES
b) Automated parameter sweep

Advantages of this approach

- **Component-based approach**
  - *Extensibility*, V&V, independent development.
- **Common solution (battery) state layer**
  - Data repository.
  - Conduit for inter-component data exchange.
- **File-Based data exchange**
  - No change to underlying codes.
  - Simplify *"unit testing"*
- **Scripting Based Framework (Python)**
  - Rapid Application Development (RAD).
  - Adaptability, changeability, and flexibility.
- **Simple component connectivity pattern**
  - Driver/workers topology.
- **Codes as components:**
  - Focus on code-coupling vs physics-coupling as first step.
- **Simple unified component interface**
  - init(), step(), finalize().