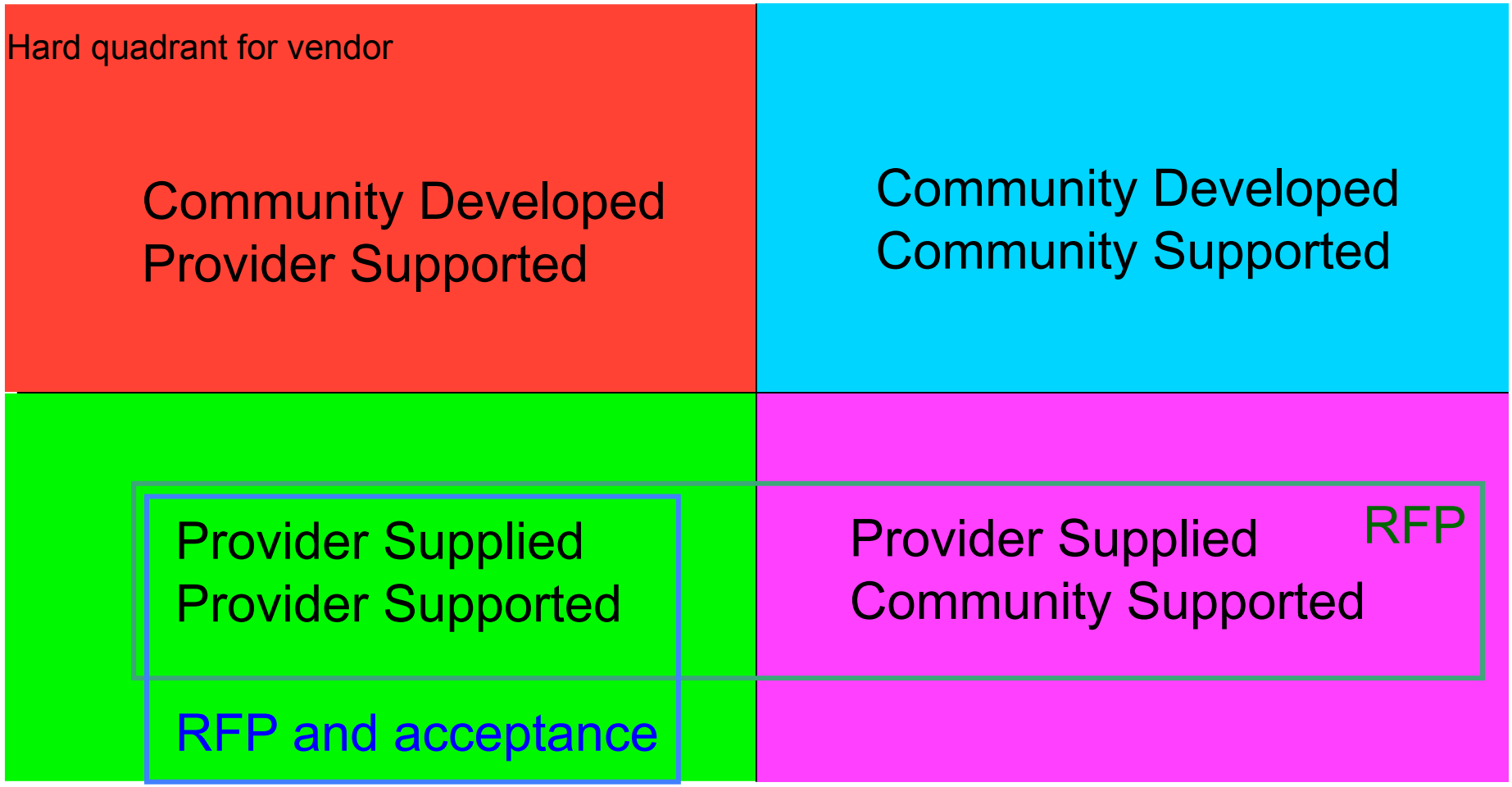
Requested Ecosystem of Software in HPC Expanding

- **Important for vendor in light of expanding ecosystem**
- **Exascale software community is interested**
- **So what are the challenges**
  - **Agreeing on a common API**
  - **Coordinated interlocked effort between vendors, community, and facilities**
  - **Time for research versus getting development done early enough**
  - ❖ **Support**

Vendor Challenges Incorporating Community Code

- **Common fallacies: just provide common API**
  - **Who defines common API**
  - **Concern is that vendor spends effort and community modifies**
  - **How to tradeoff of the need for stability against need for exploration**
- **Support is hardest nut to crack**
  - **Option 1: Customer just say they take whatever**
    - **Write open source mailing list**
  - **Option 2: Vendor fully support**
  - **Neither viable what is middle ground**
  - **Is it possible to get concrete deliverables from IESP community for components?**
  - **Is it possible to get facilities to agree on taking "less" supported open source component?**
- **Coordinated interlocked effort between vendors, community, and facilities needed**
  - **Caution, if IESP appears/claims providing all xstack then harder for vendor initiated efforts**
- **Fragmentation**
  - **Must balance early investigation against significant investment**
- **IP, Open Source Model**

Suggested Model for software



*developed implies who implemented
*supplied could be co-developed

Mapping Components to Quadrants (work in progress)

- Operating systems
  - Could be co-developed (e.g., Linux)
- Runtime Systems
- I/O systems
- Systems Management
  - Low-level (RAS, power control, boot) vendor developed
  - Higher-level resource management, security, performance co-developed
- External Environments
- Programming Models
  - Industry Standard (OpenMP, MPI, COF)
  - Other (UPC, ARMCI)
- Frameworks
- Compilers
  - Different models work
- Numerical Libraries
  - Different models work
- Debugging tools
  - Different models work
- Application Element: Algorithms
- Application Support: Data Analysis and Visualization
- Application Support: Scientific Data Management
- Resilience
- Power Management
- Performance Optimization
- Programmability

Approaches/Recommendations

- **Community should produce methodology for categorizing software components into which development and which support model they will fit**
  - **Current breakdown will need refinement but can be used as starting point**

Requirements

- **Community wants**
  - **Does not want to be limited to fully proprietary solution**
  - **Flexibility to replace components of stack**
  - **Open API**
  - **Leverage Government investment**
  - **Protect Government investment**
  - **Applications to have common environment**
  - **Scientists need to know how their devices work for reproducibility**
- **Provider**
  - **Not be held responsible for components that do not have control over**
  - **Protect other provider proprietary information (low-level system design)**
- **Facility**
  - **Level of quality**
  - **Best value**

Methods

- **Open Source** – all software is buildable source – full right to change and use
- **Open Source with formal support** – all software is buildable source with a formal (paid) arrangement for support – e.g. Lustre
- **Open Software** – all APIs are published and supported
- **Collaborative Development** – joint ownership and responsibility with a formal agreement – e.g. HPSS Collaboration
- **Co-development** – ad hoc arrangements for joint efforts – e.g. MPICH
- **Proprietary Development** – funded or unfunded development where the provide retains IP
- **Proprietary Development with escrow** - funded or unfunded development where the provide retains IP but formally promises to release all SW without restriction if they leave the business

Requirement	Open Source	Open Source with formal support	Open Software	Collaborative Development	Co-Development	Proprietary Development	Proprietary Development with Escrow
Community							
Does not want to be limited to a fully proprietary solution	X	X	X	X	?		
Flexibility to replace components of the stack	X	X	X	X	?		
Open API	X	X	X	X	X		
Leverage Government investment	X	X		X	X	X	
Protect Government investment	X	X		X	X	X	X
Applications have common environment	X	X	X	X	X	?	?
Scientists need to how their devices work for reproducibility	X	X		X	?	?	?
Provider							
Not held responsible for components that they do not have control over		X	X	X		X	X
Protect other provider proprietary information				X		X	X
Facility							
Level of Quality		X		X	X	X	X
Best Value		X		X	X	X	X

Summary

- Co-Development (joint ownership and responsibility with a formal agreement) meets all requirements
- Open source with formal (paid) support agreements meet all but one requirement

Requirements

- **Rather than assume open source or other define what the requirements are**
  - **Community developed does not need to be open source**
- **Might be different at different levels**
  - **Closer to applications**
    - **•Makes sense to be more towards open source**
    - **•Should be applicable across architectures**
  - **Closer to hardware**
    - **•Can make sense to be less fully open source**
    - **•Provider may pay for this**
- **Earlier in timeline focusing on wider community need – program models**
  - **Allow early investigation**
  - **Later in timeline then vendor-specific implementation**
    - **•funding could potentially used for non open source intiative**
- **Example out in community about mixed development model**
  - **HPSS – not open source, but community has access**

Approaches/Recommendations

- **Linux is often cited example**
  - **Linux was mature – in existence for more than 10 years before considered**
  - **MPICH is another example**
- **Model is that open source should become proved and mature and then considered for inclusion into product**
  - **Difficult to commit in advance**
- **Have a solution that leverages commercial software investments**
  - **Better sustainability**

Approaches/Recommendations

- **Open Source**
  - **Is it required from funding agency perspective**
    - **Some components could be proprietary**
      - **Funding agency concern: need to be known up front**
    - **Scientists want to know details of equipment used to produce science**
    - **Two axes**
      - **Level of software component**
        - **Higher level should be open**
        - **Some lower levels may be closed**
      - **Timeline before machine release versus after machine released**
  - **Common APIs are desired**
  - **Lowest levels may make sense to be funded out of the hardware/architecture funding**
- **Licenses**
  - **Open source license should be vendor friendly**
    - **Track pedigree of code – contributor agreement**
    - **Gate keepers for individual components**
    - **License non-viral**
      - **If incorporated into product need to be able to charge for product**

Approaches/Recommendations

- **IP**
  - **Money provided by funding agencies for software efforts does not result in particular vendor owning**
  - **Affected by which funding agency provides resource**
    - **Ministry of Science mission different than Ministry of Economic Affairs**
  - **Exceptions are okay but should be identified early in the process**
  - ❖ **Recommendation is to work early on draft IP agreements**
    - ❖ **IESP produce framework and take back to each country, region, agency, etc.**
    - ❖ **Produce draft for bulk of terms or agreement**
- **Co-Design**
  - **Recommendation – software roadmap needs to identify key application characteristics**
    - **Computational science should form bridge between science and computer science to help provide classes of applications and identify a small set of characteristics for each class**
  - **Support for rich simulation environment**

Approaches/Recommendations

- **Software life cycle – software roadmap / Governance model**
  - **Front end – development**
    - **Need to points at which code comes together for test and integration**
  - **Back end – post deployment**
    - **Who is responsible, where does funding come from, on-going maintenance and support**
    - **Needs to have access to reference platform to test on**
    - **Need to have coordination between components**
  - **Good rule of thumb for support**
    - **1 Person researching and developing : 1 person testing, maintenance, and support**
  - **Interlocking milestone between community, facility, and vendor**
    - **Define what happens when milestones slip**
    - **Ensure adequate timeline for upfront research and exploration**
      - **Example: LSST large synoptic survey telescope – data pushed through software pipeline periodically to indicate progress**
  - **Should identify timeline of research phase**
    - **When each different components need to move from research to implementation**
    - **Software roadmap team needs to define timeline**
  - **Need intermediate trial points 10PF, 100PF, etc, that software stack runs on and can be tried by community (also should run on TFLOP desk side machines)**
  - **Need to identify open source phasing**
    - **Default could be open and vendors indicate which areas would be challenging**

Summary Recommendations

- **Community should produce methodology for categorizing software components into which development and which support model they will fit**
  - **Produce initial breakdown in time for input to Roadmap committee**
  - **Current breakdown will need refinement but can be used as starting point**
- **Use interlocking (between vendor, community, facility) in time milestones**
  - **Co-Development (joint ownership and responsibility with a formal agreement) meets all requirements**
- **Produce model that allows for components to become mature before inclusion**
- **Co-design: software roadmap needs to identify key application characteristics**
- **Funding agencies should apply resources to integration, maintenance, support**
  - **Rule of thumb: 1 Person researching and developing to 1 person testing, maintenance, and support**
- **Open source license should be vendor friendly**
- **Work early on draft IP agreements**
- **Roadmap committee needs to produce software roadmap by September 2010**
  (covered in funding agency section)

Conclusion

- **Good potential but need to figure out how to integrate and effectively work together between vendors, community, and facilities**