What Our HPC Machines Are Busy Doing

- Installed base of medium - large machines
  - government research centers and national labs
  - academic institutions and centers
  - industry R&D sites
  - commercial and financial institutions

- How the machines are being used
  - (serial) batch server farms
  - interactive development / testing
  - parallel production runs

- Typical “prime” approximately 2 years
  » planned obsolescence
  » Mean-Time-to-Bankruptcy

HPC as a Capital Investment

The way to increase productivity is to substitute capital for labor

- HPC requires a significant capital investment -- does it increase productivity?
- Productivity presupposes
  • availability of key applications on HPC platforms
  • making (relatively) efficient use of expensive resources

... how good is HPC's track record?
The "Hidden" Cost of HPC

- Buying the machine vs. using it for something
  - capital costs vastly out-paced by human costs

- What parallel applications really cost (per line)
  - migrating a parallel application
  - developing a key application
  - best-in-the-business estimate (G. Montry)

- "We had this problem with vector computing..."
  - J. Worlton's estimate of per-line cost

$800 per line
$100
$25

- develop
- migrate
- best
Is Parallelism Affordable?

- Shell's success story
  - large-scale parallelism in daily production
  - more cost-effective than big iron
  - what they paid to get there
- A competitor’s story

(We may have parallel machines, but not necessarily parallel jobs)

“Crucial decisions along life’s highway”
How Did It Get This Way?

- **Symptom:**
  highly-paid scientists/engineers spending ridiculous amounts of time mastering details of HPC technology

- **Contributing factors:**
  under-estimation of effort required

  M. Ess (about business users of ARSC): “If they have considered the programming problem at all, they assume that it is someone else's problem or can be solved with the tools that came with the machine.”

  wildly distorted expectations

  C. Cook, C. Pancake, R. Walpole (about SC’93 attendees who want to use parallelism): “25% expect a factor of 76 or more improvement if their application runs on 100 processors; 12% thought they could achieve better than linear speedup!”

---

The Goal: A Healthy Software Infrastructure

"Worrton's Iceberg Model"
The Reality: A Faulty Software Infrastructure

Silicon (r)evolution has vastly out-paced software (r)evolution

What Application Developers Say about Software Support

- Today's system software and tools:
  - crash!
  - don't do what they're supposed to do
  - are too machine-specific
  - are too diverse and inconsistent
  - are not interoperable
  - are too complex for users to understand
  - don't scale to large applications
  - don't scale to large numbers of nodes

“Vendors always respond that there isn't a problem; you have to go to the user group meeting, and then you find out it's a problem for everyone!”
What HPC Vendors Say about Software Support

- System software and tool developers:
  - must scramble just to maintain the status quo
  - must be able to differentiate their products
  - get user requests that conflict with company priorities
  - get user requests that conflict with other user requests
  - are under-staffed because procurements don't make/break on software
  - receive mixed messages about whether users really want standards

“We're not trying to produce shoddy or difficult products!”

“If users can't make up their minds what they want, how can we give it to them?”

Whose Attitude Is The Problem?

Build it and they will come...

Whine enough, and they will build it...
What's Wrong with System Software and Tools?

- Common breakdown of software attributes (from classic software engineering texts)
  - reliability
  - maintainability
  - modifiability
  - efficiency
  - usability

- Usability -- last on the list, but implicit to them all

Why build a reliable, maintainable, easily changed, and efficient product -- if no-one can or will use it?

Designing for Usability

Usability doesn't just happen -- it's planned from the outset

- focus on the customer, not the service provided
- "demand-pull" rather than "supply-push"

“Darn these hooves! I hit the wrong switch again! Who designs these panels, raccoons?”
We Ignore the Key Resource for HPC Usability

- Parallel computers have been around for over 10 years
  - thousands of adventurous users have evolved successful strategies
  - vendors should enlist their collaboration

- Questions to ask before beginning software planning:
  - *Will this do something that users do now?*
    If so, what constrains the productivity of their current method?
  - *Else, is this something that users have specifically asked for?*
    If so, why aren't they doing it now?
      (too time-consuming, needs OS support, etc.)

Blueprint for Usability:

(1) The Best Strategy Is One That Works!

“Maybe we should write that spot down.”
**Blueprint for Usability:**

**2 Motivation Is a Fragile Thing...**

- Users choose whether or not to use a software product
- Understanding user strategies helps the designer know
  - how users subdivide the workload into tasks
  - what tasks need to be easiest
  - what tasks need to be fastest
  - what tasks are most subject to variation or indecision
  - what words need to be used
  - where automatic checkpointing is needed

**Blueprint for Usability:**

**3 It's Time to Recognize the "Cultural Chasm"**

<table>
<thead>
<tr>
<th>Developer Reactions</th>
<th>User Reactions to Same Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>“That’s a great way to show...”</td>
<td>“Why is it taking so long?”</td>
</tr>
<tr>
<td>“Can I see all eight views at the</td>
<td>“Which part am I supposed to be looking at?”</td>
</tr>
<tr>
<td>same time?”</td>
<td></td>
</tr>
<tr>
<td>“…And what does Ctl-Click do?”</td>
<td>“What do I do next?”</td>
</tr>
<tr>
<td>“This will make great screendumps!”</td>
<td>“Why is it showing me all of these things?”</td>
</tr>
<tr>
<td>“Can I launch all the other tools</td>
<td>“How is all that supposed to help improve my program?”</td>
</tr>
<tr>
<td>from this one?”</td>
<td></td>
</tr>
<tr>
<td>“How much of the appearance can I</td>
<td>“How much work does all that take?”</td>
</tr>
<tr>
<td>customize?”</td>
<td></td>
</tr>
</tbody>
</table>
Blueprint for Usability:
(4) Vendors & Users Have To Work Together...

“How social animals work together”
What Vendors Can Do to Help

HPC software support already has a bad reputation
--> *the burden of proof is on you*

Strategy for effective interaction with users:
- provide users with realistic estimates of level-of-effort required to implement system software and tools
- enlist their help in prioritizing needs
- get them to furnish specifics on
  - exactly what information they need
  - how much information is required to perform each task
- take it seriously when they ask for very simple support

What Vendors Can Do to Help

- Engineer your software for usability
  1. Don't even start until you can prove there's user demand
  2. Analyze and understand how users approach the task(s)
     - they won't change their patterns to fit your software
  3. Carry out early testing/evaluation with users
     - only way to ensure the system will actually meet their needs
  4. Design iteratively with many cycles of design/user-test/redesign
     - system progressively becomes more tailored to user needs

Usability achieved *iff* users want to do this task in this way
Who Should Pay for the Emperor's Clothes?

What Funding Agencies Can Do to Help

- Vendors can't foot the bill for openness / interoperability
  - you should be catalyst for change

- Strategy for enabling user/vendor interaction:
  - provide incentives for collaboration
    - make collaborative projects easier to propose
    - ask for (and reward) evidence of collaboration
  - pay for elements of standards efforts that don't directly benefit vendors
    - administrative and logistics costs
    - development of test suites
  - provide incentives for vendors to implement better software
    - make sure HPC procurements specify standards
    - make sure software support is a key purchasing criterion
We're All Guilty...

“Early experiments in transportation”

Oregon State

Parallel programmer

Purchasing authorities

Vendor representative

What Users Can Do to Help

• You're part of the problem
  --> you must take an active role in the solution

• Strategy for effective interaction with vendors:
  ▶ constructivism, not gripes
    » publicize development/debugging/tuning strategies that work
    » articulate your needs for simple, robust functionality
  ▶ don't just state things, explain them
    » remember the "cultural chasm"
    » in critiquing software, be specific about what you would/wouldn't do
  ▶ speak with dollars as well as words
    » vendors can't be expected to respond to single-case needs
    » if it's not on procurements, you won't get it
    » omission of standards from procurements sends a message, too
It’s Time to Re-Think...

- How we should measure HPC’s “success”
  - gains: impressive speeds
  - costs: users are less productive
- What constitutes a reasonable software infrastructure
  - want usability
  - but won’t collaborate to achieve it
- What message current procurements convey
  - purchases based on hardware speed
  - want, but don’t require better software infrastructure
  - want “standards” but don’t require existing ones

We’re All in This Together ...

productive application developers ↔ stronger HPC industry
Produced with (involuntary) artistic assistance from:
Nadine Bernard Westcott, creator of the emperor
Gary Larson, creator of Far Side
Tom Wilson, creator of Ziggy