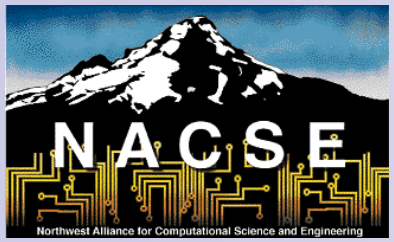
HPC Productivity – Are We ~~Addressing the Right Issues~~?

Cherri M. Pancake

Northwest Alliance for Computational Science & Engineering (NACSE)

Oregon State University

pancake@nacse.org



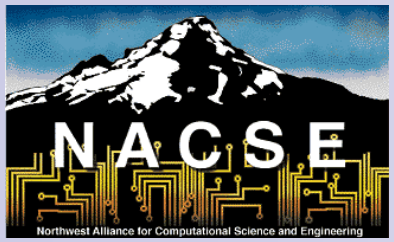
HPC Productivity – Are We Asking the Right Questions?

Cherri M. Pancake

Northwest Alliance for Computational Science & Engineering (NACSE)

Oregon State University

pancake@nacse.org



We're Fellow Travelers …

- HPC isn't a goal – it's a road
  - High(way to) Performance Computing
- But …this year's talks all cited "productivity" too
- Is productivity just a new word for performance?

[Still using the same metrics and asking the same questions]

(1) Why Are We on the Road ?

[aka "why do we need performance?"]

a) **Because we're nerds and like new technology**

b) **Because we think speed and power are cool**

c) **Because current systems can't do what we need from them**

IMPLICATIONS

- **We should focus on what's really needed and why – not just where technology can take us**

Focus on … Why We Need Performance

- We're heading for stormy weather
  - HPC users are becoming an endangered species
    - Users want to be productive, not just cool
  - ROI for human effort is too low
    - "I spent a whole week, to get a microscopic improvement"
  - "Ramp-up cost" is way too high
    - "I had to take a course to learn how to use XXX, and then it didn't solve my problem"
  - It's pointless to talk about attracting new users when we're having trouble keeping current ones
    - "I've switched back to Matlab – it takes days to run, but I can spend my time doing important things"

Focus on … Why We Need Performance

- ***Getting it right #1:*** **Start with how we could improve user productivity <u>today</u>**
  - **Better ROI on human investment**
    - **Encourage being realistic (modest) about expectations**
    - **Discourage investing effort in "improvements" that may yield only marginal results**
  - **Lower ramp-up cost**
    - **Stop developing and recommending "do-all" tools**
    - **Develop some shortcuts that do simple things easily**
      - **e.g., scripts that invoke a complex tool behind-the-scenes – so users don't have to learn how**

(2) Can We Do a Better Job of Navigating?

[aka "aren't we forgetting some key approaches?"]

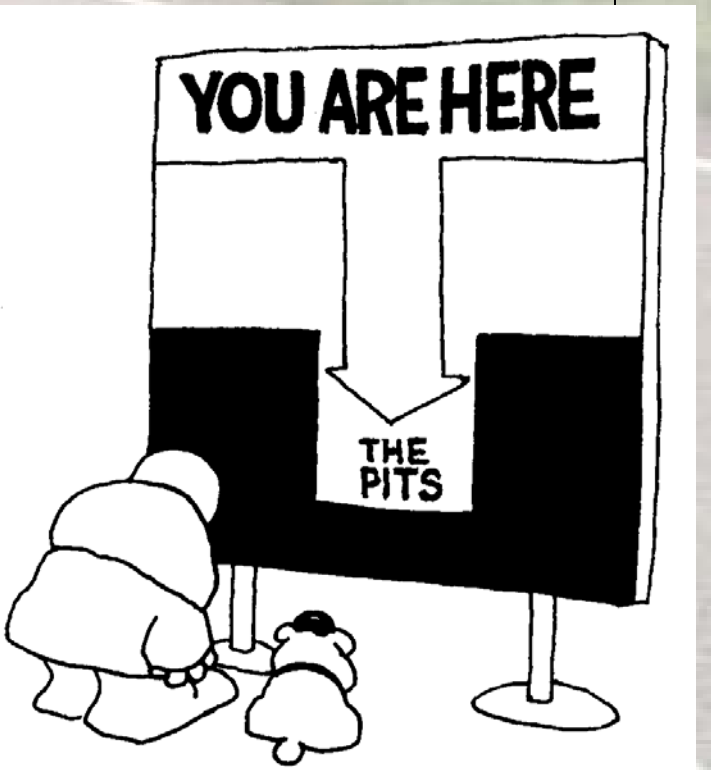
a) **No, we've thought of everything**

b) **No, automated is the only way to go**

c) **Yes, we could focus on Tool <u>plus</u> Human**

IMPLICATIONS

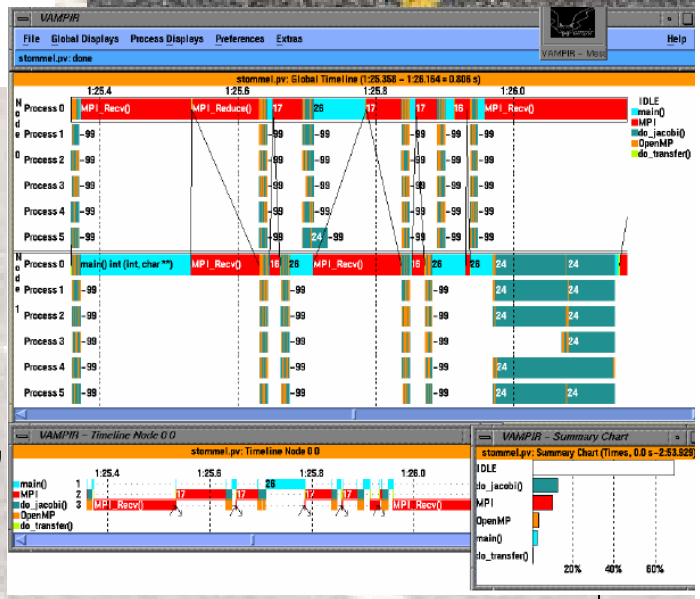
- **We should capitalize on what the user knows about his/her code**



Reexamine … Neglected Approaches

- We're wearing "blinders"
  - Tool-builders rule #1: get a good name

PETrAT
Performance **E**vent **Tr**ace **A**nalysis **T**ool



  - Tool-builders rule #2: show the user everything or nothing
  - Tool-builders rule #3: the real goal is new technology
    - "Self-propelled instrumentation"
    - "Autonomous data analysis"
    - "Automatic pattern analysis"

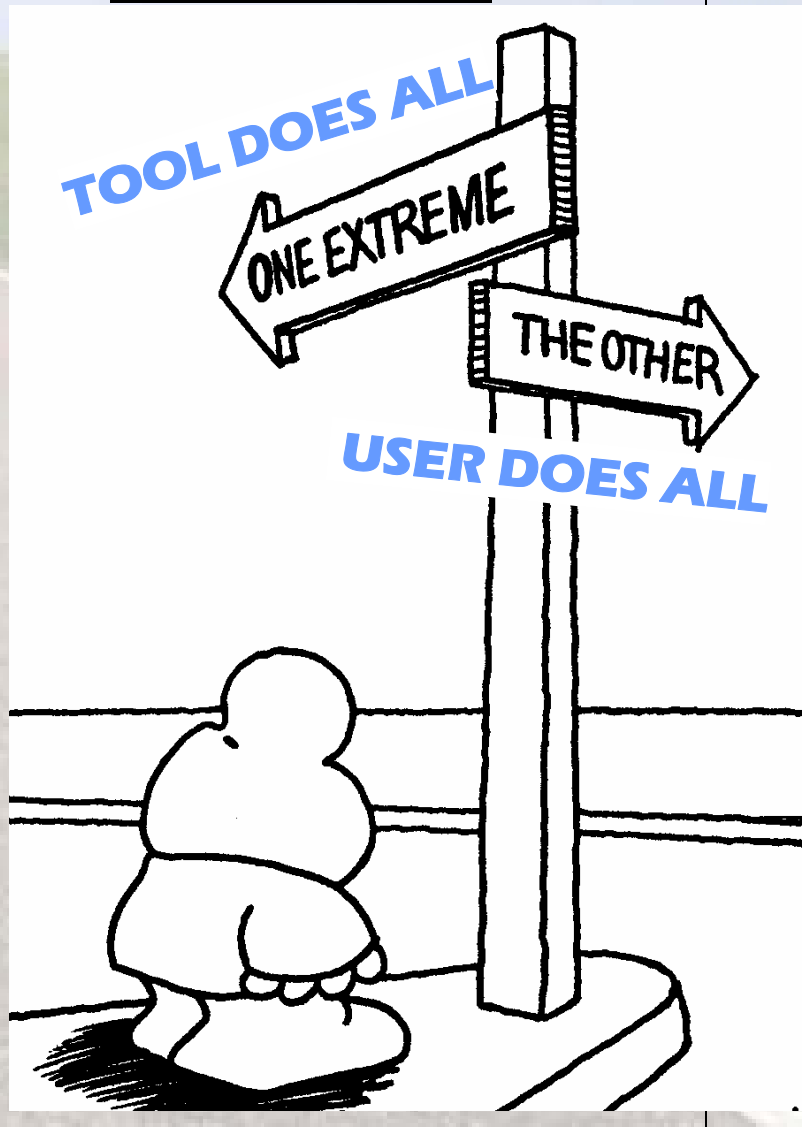
Why Don't
Users Use Tools Tool Developers Develop?

oops!
slide from 1993

Why do we think
tools know so much more than humans?

Reexamine … Neglected Approaches

- *Getting it right #2:* Create a real partnership between tool and user
  - Forget show-all and show-nothing approaches
    - "Only tool developers like having a dozen windows that pop up all over the place"
    - "I don't know what the compiler did, but when I change that one line it trashed the performance"
  - Model displays on the successful "wizard" style
    - Step users through logical process – with good defaults



Reexamine … Neglected Approaches (2)

- Exploit fact that users know their codes better than anyone (anything) else
  - Tools are making it harder than it needs to be
    - Have to assume "all behaviors/values are equally possible"
    - Reality may actually be much easier to analyze
  - Why not prompt the user for key information to improve optimizations
    - "Will this loop execute >1000 times (a) every time it runs, (b) often, (c) sometimes, (d) rarely?"
    - "What is the highest value loop index I will take under normal conditions?"
  - And to streamline performance analysis
    - "Were the inputs for this run (a) typical, (c) somewhat representative, (c) atypical?"
    - "This loop took 87% of total runtime. Is that (a) typical …?"

(3) Will We Know When We Get There?

[aka "is anything less than perfection a success?"]

a) **Only if we reach HPC Heaven – it's all or nothing!**

b) **Never mind – "It's the journey, not the destination"**

c) **We <u>need</u> midway points – otherwise, we can't get the users on board**

IMPLICATIONS

- **Incremental steps – small, practical tools that really address user priorities – would make the journey faster**

Aim for … Saving Users' Time Now

- Stop looking for perfection (silver bullets)
  - Instead, do more about what's hardest for users now
  - Stop focusing on "new and sexy" or "publishable"
  - The right incremental improvements could
    - Stem user attrition
    - Get us closer to productivity

Aim for … Less than Perfection

- ***Getting it right #3:*** **Start addressing where users spend (waste) the most time**
  - **Many sinkholes are simple – and addressable now**
    - **Rebuilding application after tiny incremental code changes**
    - **Simply finding where standard libraries/files are on different machines**
    - **Lightweight corefile concept (quick, cheap answer to "where did my code crash?")**
  - **Start doing it the way users have been asking for almost 20 years**
    - **Split mega-tools into pieces with simpler scope, so they can be easier and faster to use**

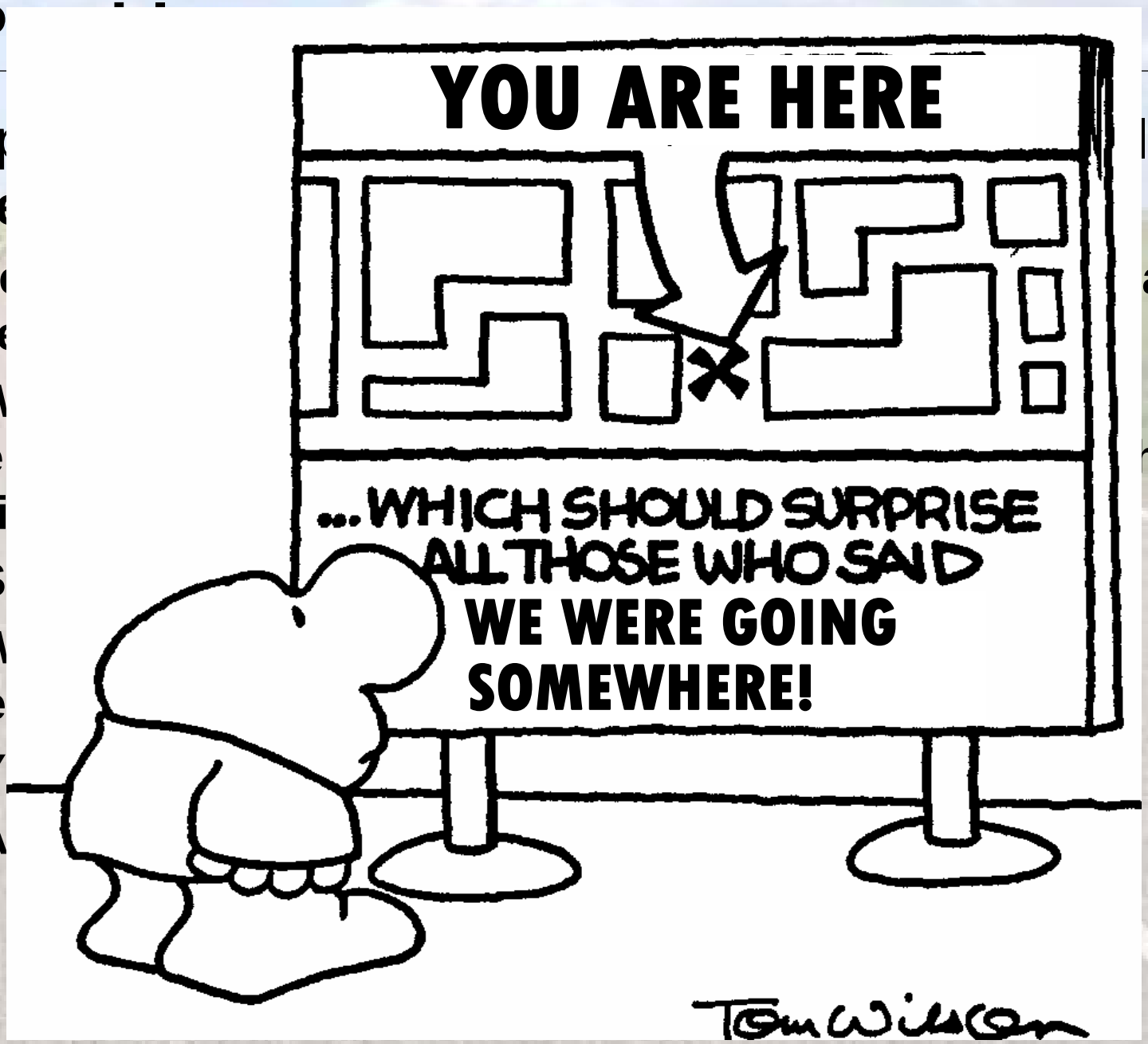
A P

[Insp
Ge

- Fr
the
- "W
ve
thi
- "S
- "W
de
- "Y
- "A

at

hole



Conclusions

- It's not a matter of "scaling tools up" for Pflop computing – they don't cut it now
- Need to focus on what's really needed and why
  - Users want to be productive (not cool) scientists
  - Attrition is not a coincidence
  - Must get better ROI on human investment
- To be workable, tools should
  - Partner with users to exploit their knowledge
  - Start with productivity sinkholes that can be addressed now
  - Do it the way users want: simple units for specific needs

Special thanks to Tom Wilson, creator of Ziggy