What Does “Influential” Mean in Software Engineering Research?

Judy Stafford
Software Engineering Institute
CMU 17-939A
November 5, 2001

What does Influential mean?

- Parnas: Paper is considered influential if it influenced the way that programs are written.
  - He wants to see a direct influence from the paper at ICSE to the work of programmers.
  - What do we think about that?
  - What other metrics can we come up with to define influence?

What was Dave really upset about?

- Had he simply lost patience?
- Does he not recognize the immaturity of the field?
- Did he feel
  - the wrong papers are being published?
  - the wrong research is being done?
  - the wrong courses are being taught?

Science or Engineering

- Debate over whether SE is engineering
  - Is ICSE
    » a conference on the science of software engineering or
    » a meeting place for software engineers to improve their craft?
  - Are WE software engineers or software scientists?
  - My bias is that we are software scientists working at creating an engineering discipline for developing software.
The Impact

- Lots of discussion and debate among SE researchers
- New tracks at ICSE
- Impact project

ICSE2002

- Submission Categories
  - Technical Papers
  - Tutorial Proposals
  - Workshop Proposals
  - State-Of-The-Art Proposals
  - Panel Proposals
  - Doctoral Symposium
  - Posters and Demos
  - Industry Participation
  - Student Volunteers

Panel at OOPSLA 2001

- Lee Osterweil
  - Dave Thomas, Bedarra Corp. and Carleton University
  - Richard Gabriel, Sun Microsystems, Inc.
  - Alexander L. Wolf, University of Colorado, Boulder
  - Barbara G. Ryder, Rutgers University

An Argument: Research/Product Timing

- Research initiative was shared between academia and industry
- Some research tools were seriously used in practice
  - Make, RCS, Odin, Adele ...

Products

- Make
- RCS
- Odin
- Vodoo
- Jasmine
- NUCS

Research

- CCC/CC
- Harvest
- NSE
- CCC
- PSI
- VODOS
- DSEE

Time

- From A. van der Hoek
- Progress

Architectures for Software Systems

CMU-CS 17-655,17-755

April 17, 2001
Some Lessons Learned (1)

- Vendors tend to consider that impact is restricted to…
  - algorithms (e.g., differencing)
  - pieces of reusable code (e.g., RCS)
  and not…
  - concepts (e.g., hierarchical workspaces)
  - architectures (peer-to-peer repositories)
  which are often seen as “engineering common sense”

Some Lessons Learned (2)

- Researchers tend to consider that…
  - precedence
  - concepts
  - prototypes
  are sufficient as impact and ignore…
  - efficiency
  - usability
  - reliability
  dismissing them as “engineering common sense”

Some Lessons Learned (3)

- Both are right, both are wrong
- A good idea is had more than once
- Vendors have disincentives for distributing credit for ideas
- Researchers have incentives for claiming credit for ideas
- Research and productization both require engineered creativity

Conclusion

- SCM is a successful field
- Research provided many inputs and was clearly influential
- Vendors successful in finding/adapting ideas to fit customer needs
- Many ideas tried by researchers have not (yet) found their way into products/practice
- Interplay between vendors and researchers exists, but not any easy relationship