ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Chair of Systems Design www.sg.ethz.ch

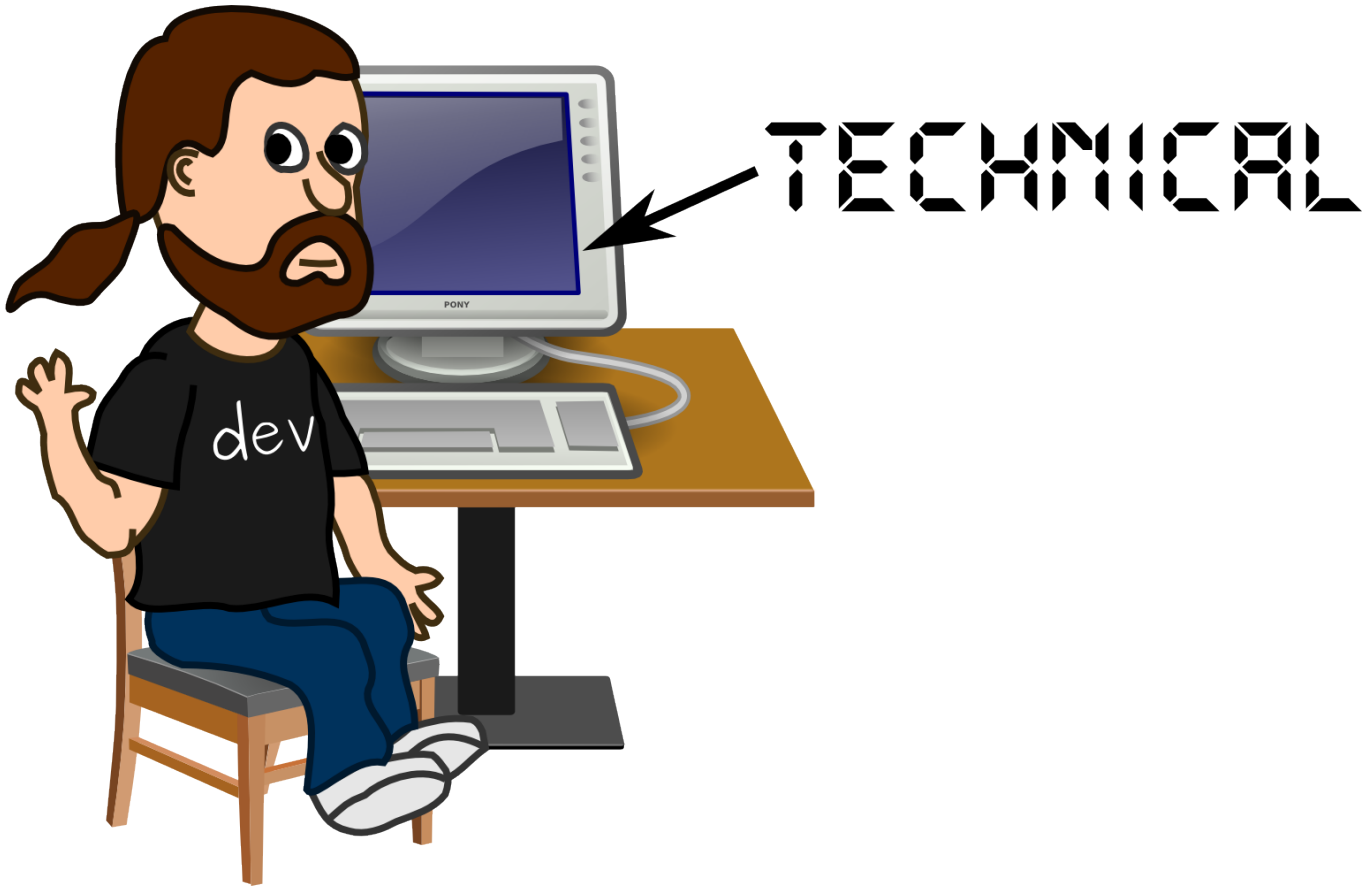
Empirical Software Engineering: Complexity Matters

Marcelo Serrano Zanetti

mzanetti@ethz.ch

Empirical Software Engineering

Empirical Software Engineering

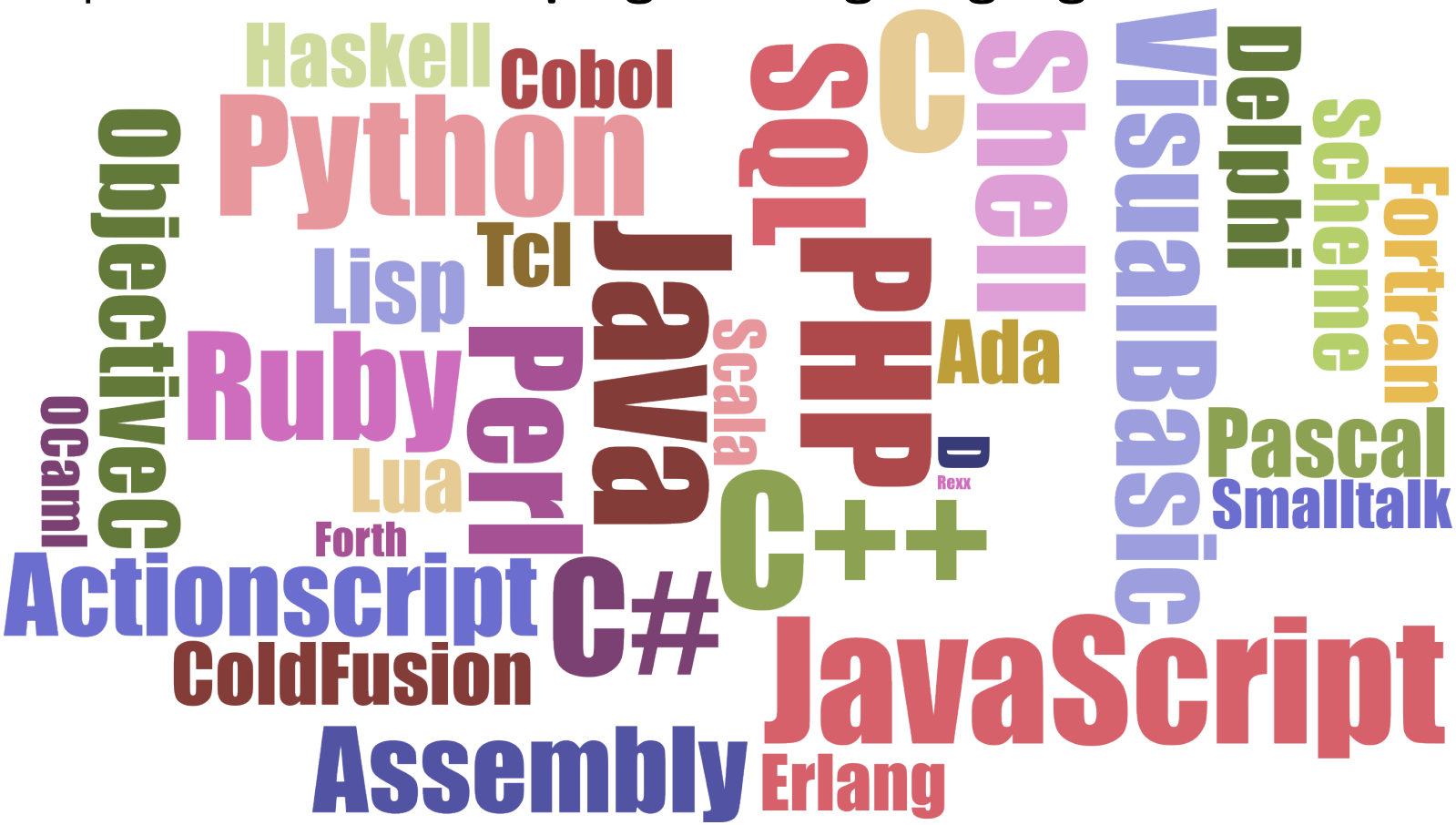


What Works Best and When?

- How do we optimize project performance?

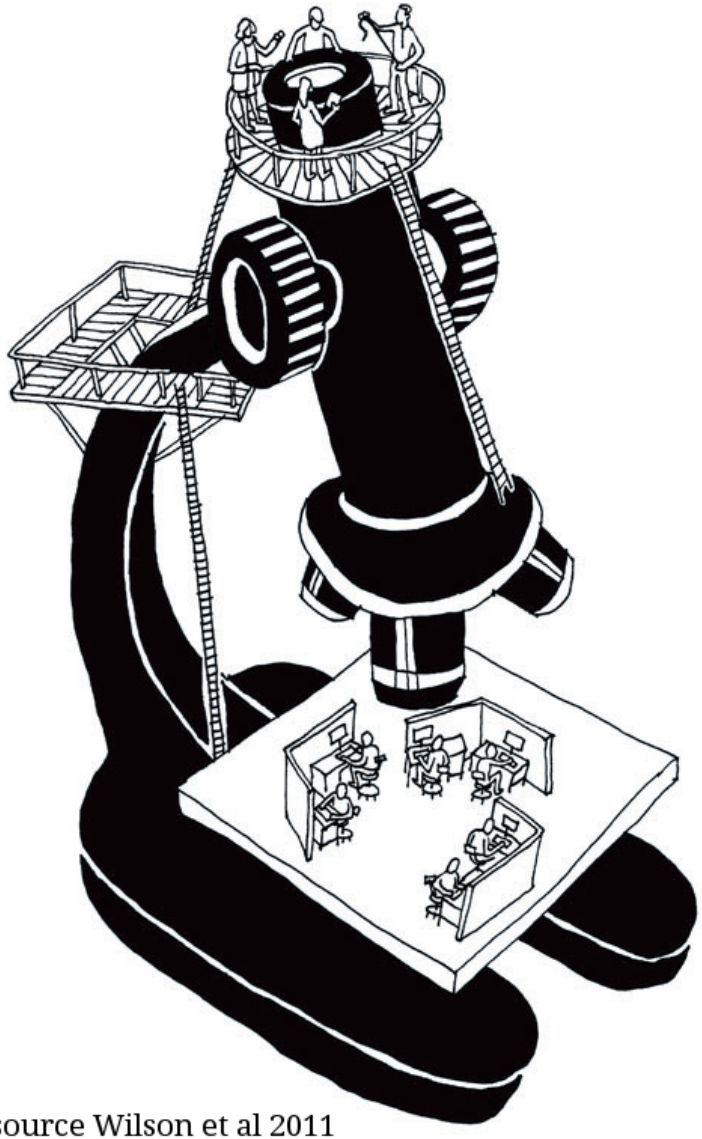
What Works Best and When?

- How do we optimize project performance?
- Example: the choice of a **programming language**



based on raw data from LangPop.com 2011

Empirical (Scientific) Software Engineering



source Wilson et al 2011

– Pardon me, but what exactly is not trivial from what you have told us so far?

Software is Written By People

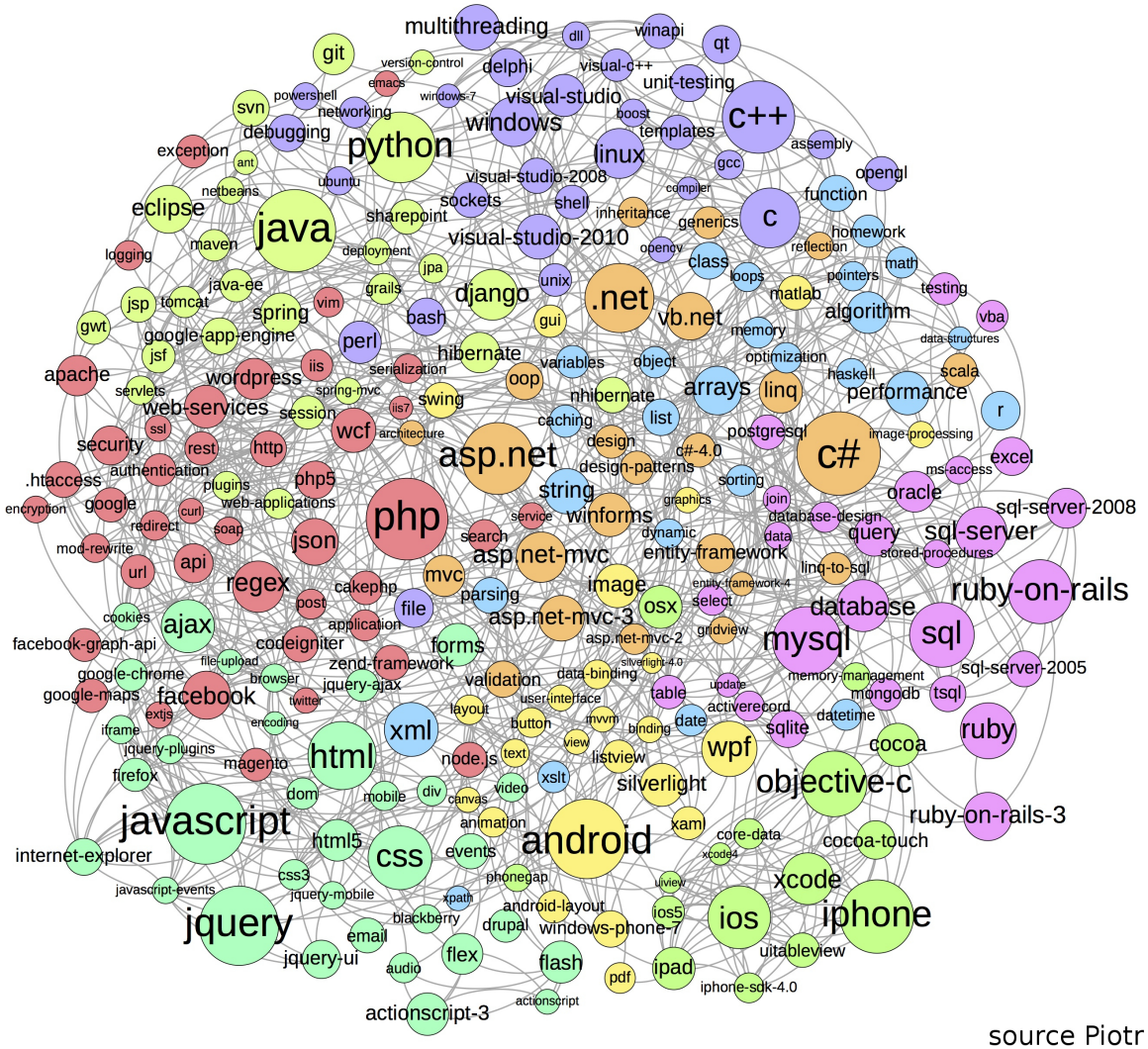
Software is Written By People

Catastrophic Complex Social Interactions

Millennium Bridge

– I knew it all along ...

How do We Handle Complexity?

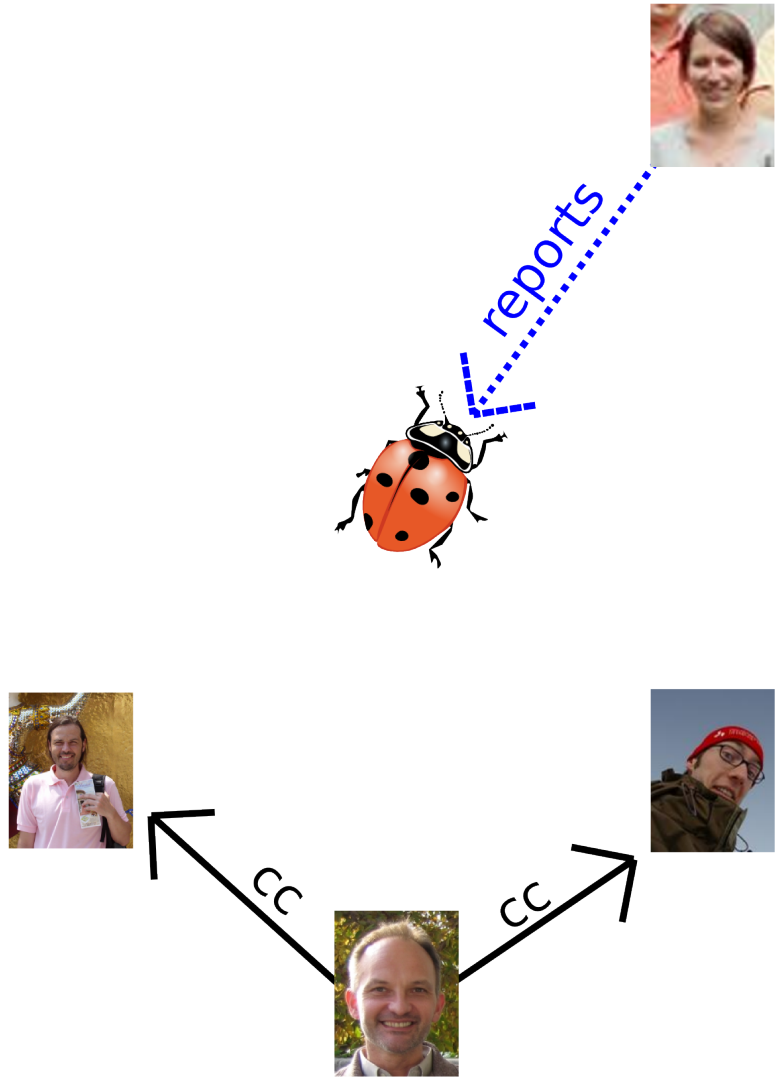


source Piotr Migdal 2012

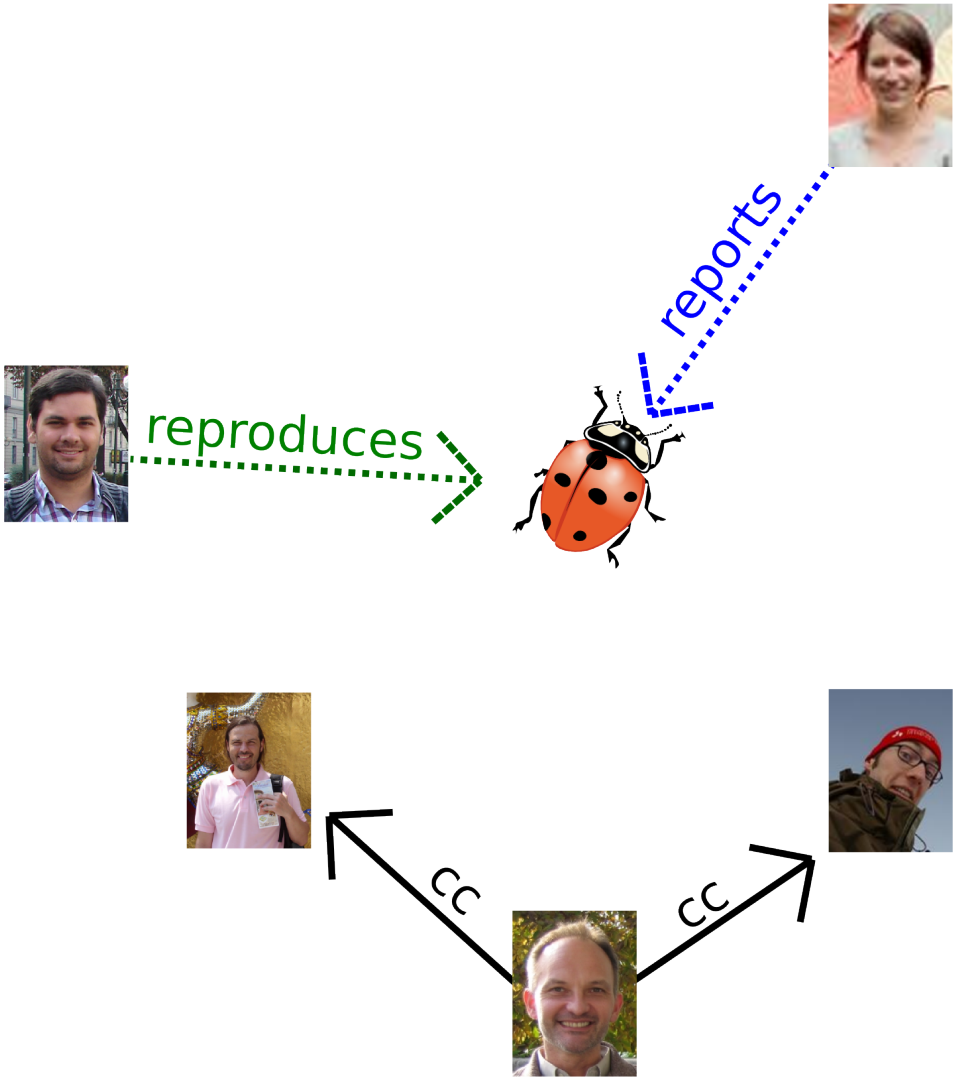
Network Science!

Networks in Collaborative Software Development

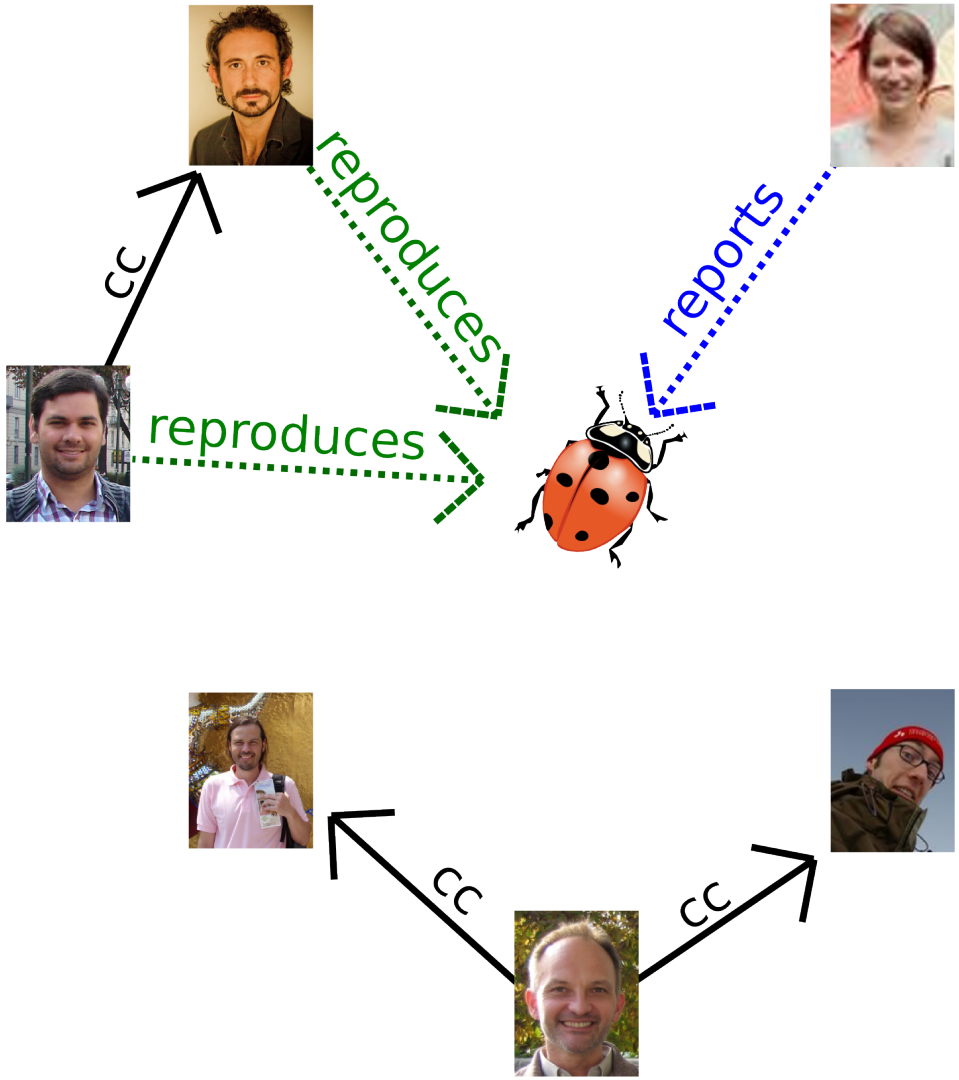
Networks in Collaborative Software Development



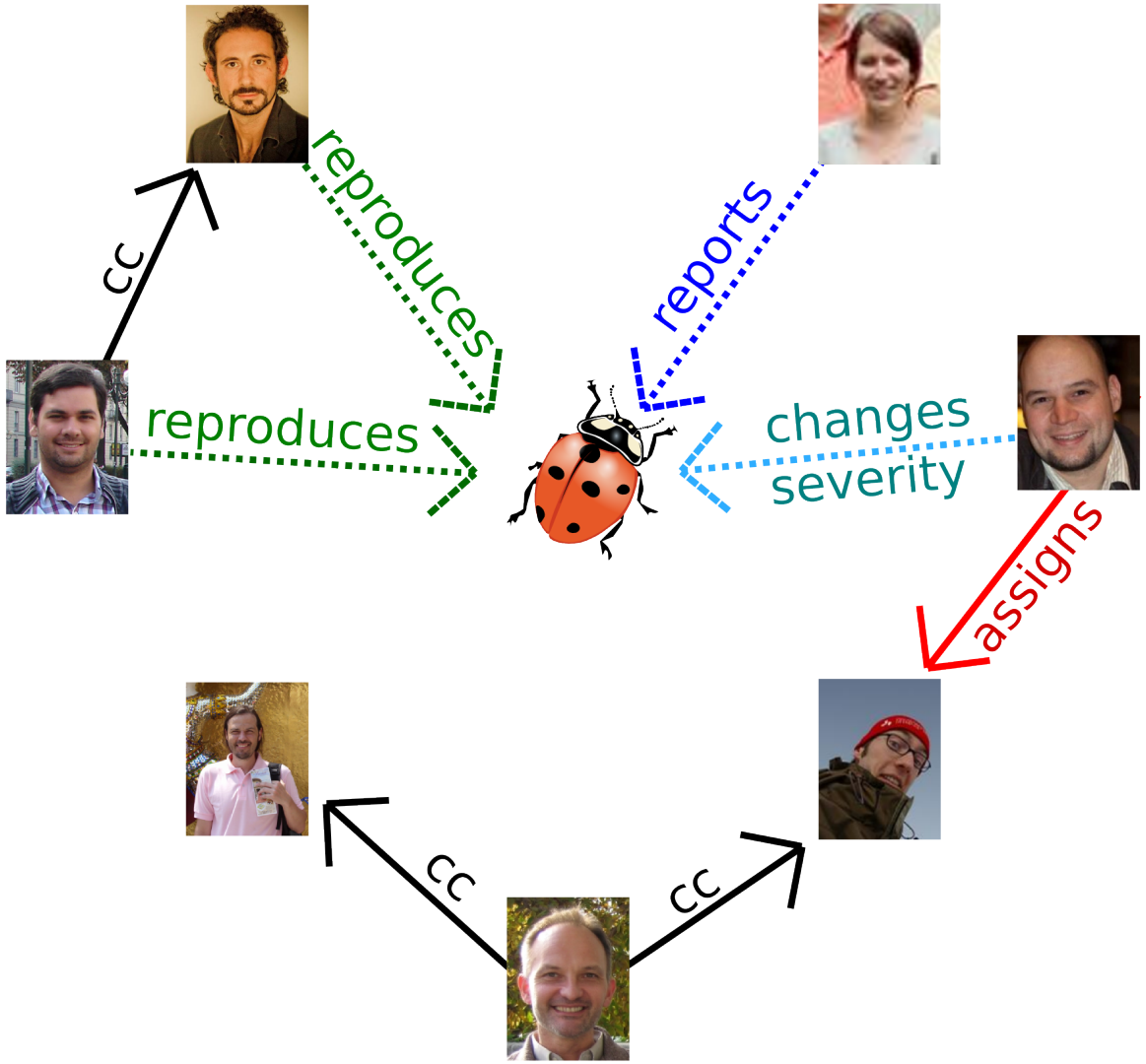
Networks in Collaborative Software Development



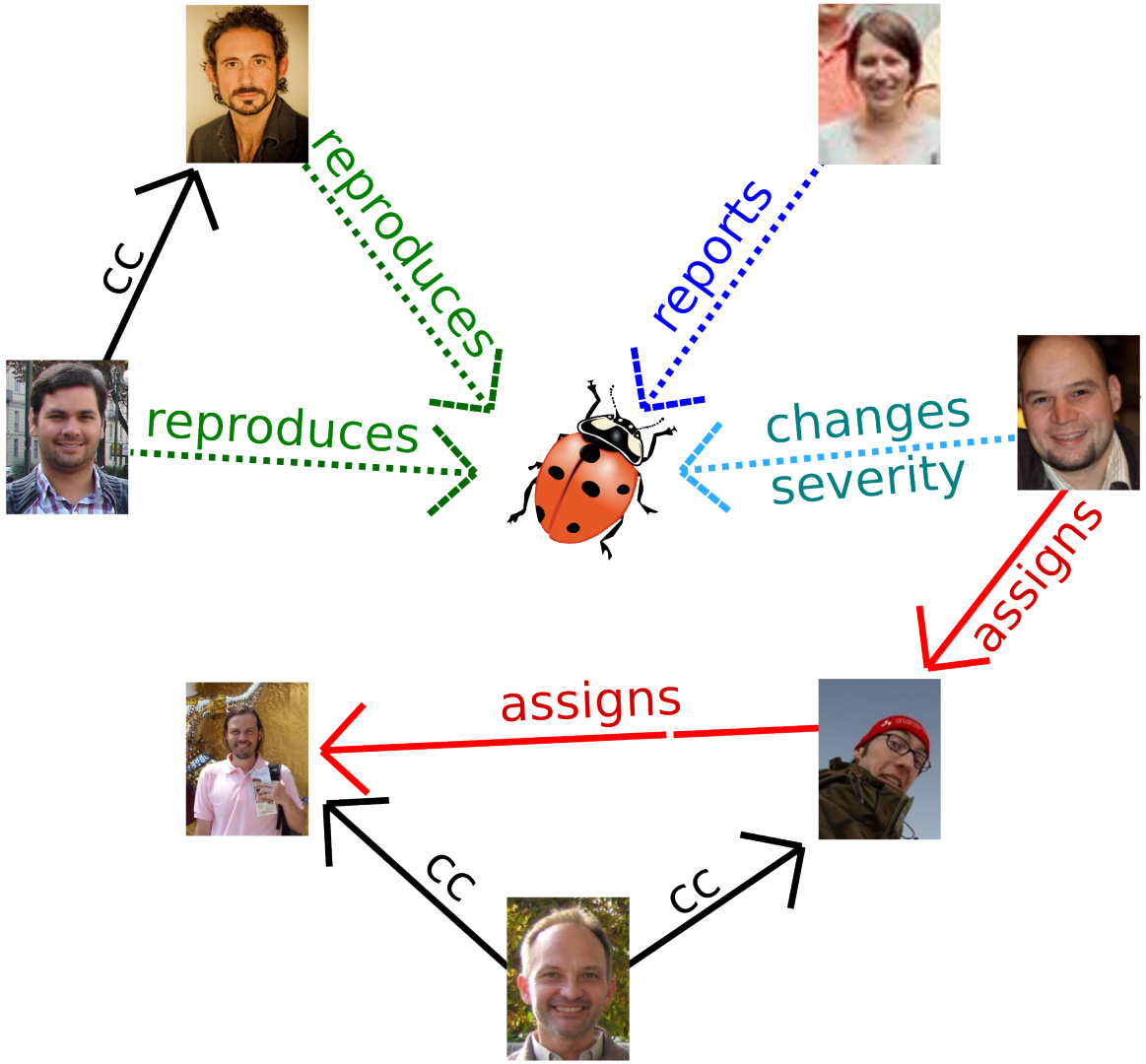
Networks in Collaborative Software Development



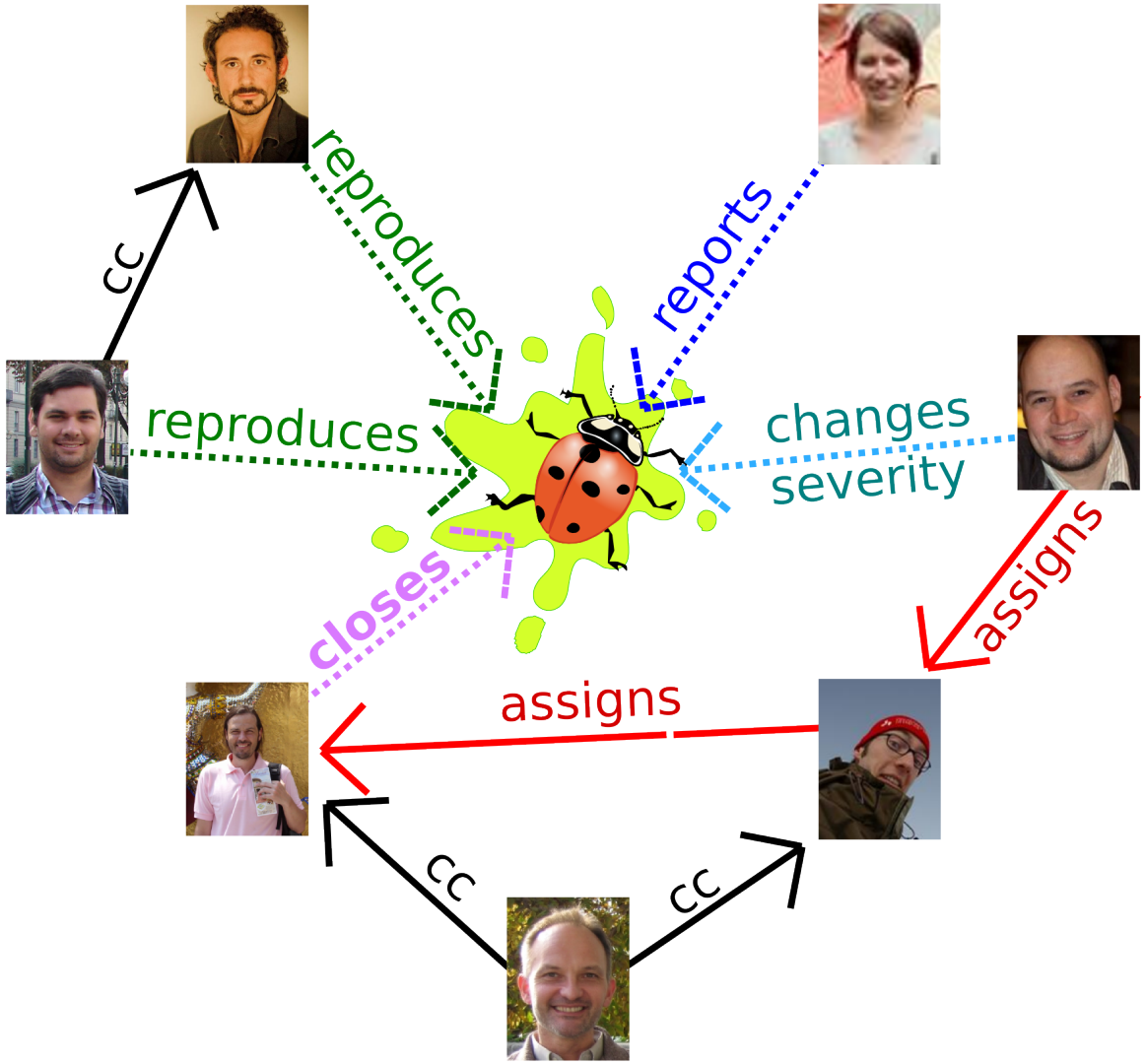
Networks in Collaborative Software Development



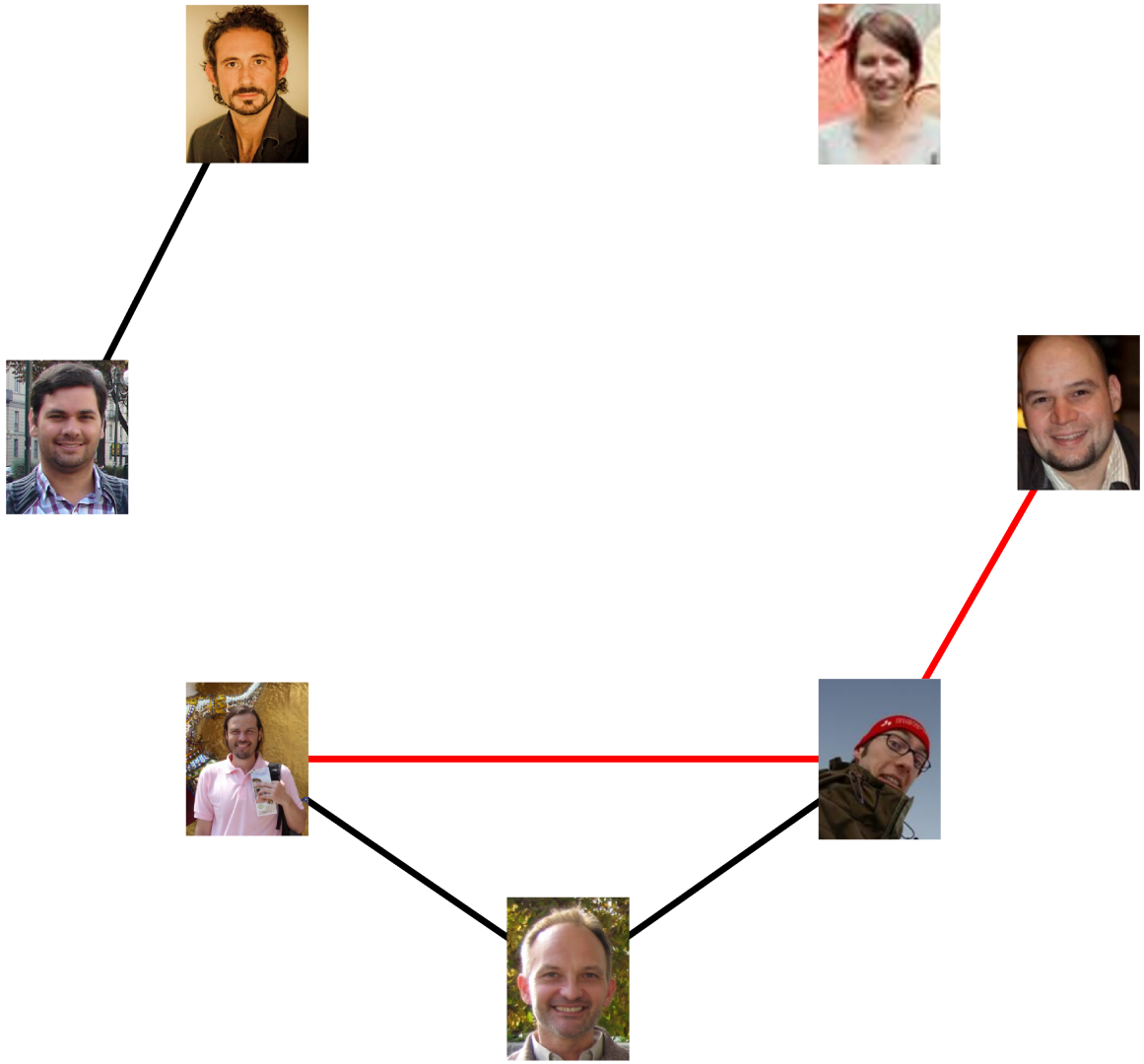
Networks in Collaborative Software Development



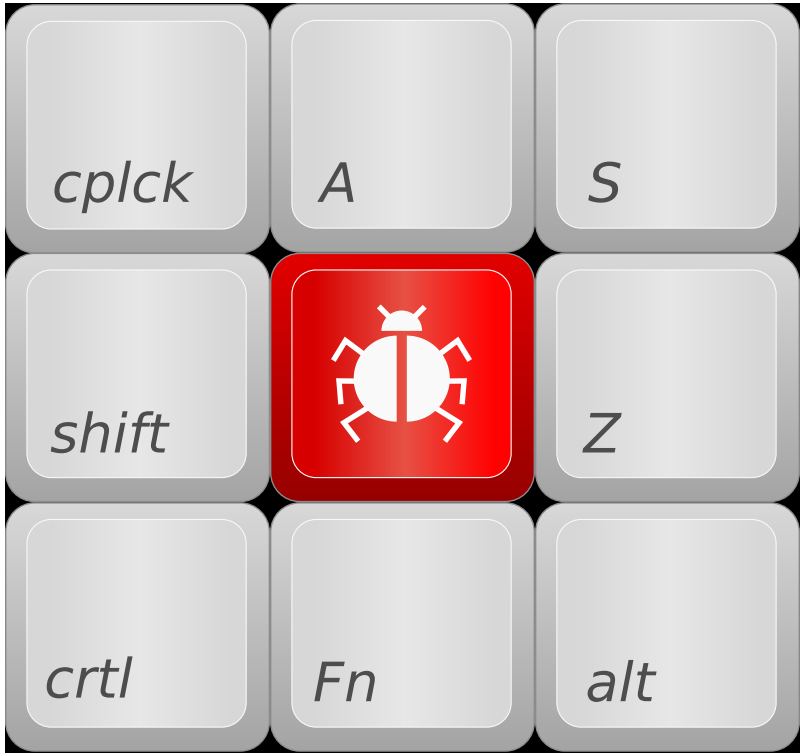
Networks in Collaborative Software Development



Networks in Collaborative Software Development

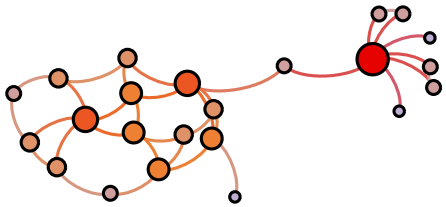


Who Writes Valid Bug Reports?

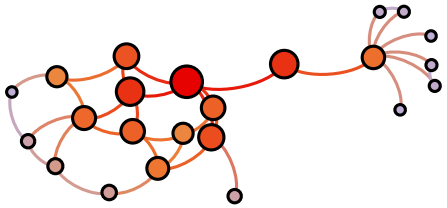


- **valid** bug report:
  resolved as FIX or WON'T FIX
- **faulty** bug report:
  resolved as DUPLICATE or INCOMPLETE or INVALID

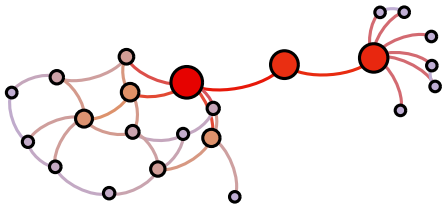
Who is Important in a Network? Depends ...



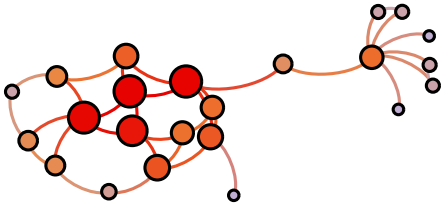
degree centrality



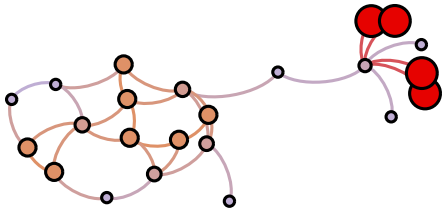
closeness centrality



betweeness centrality

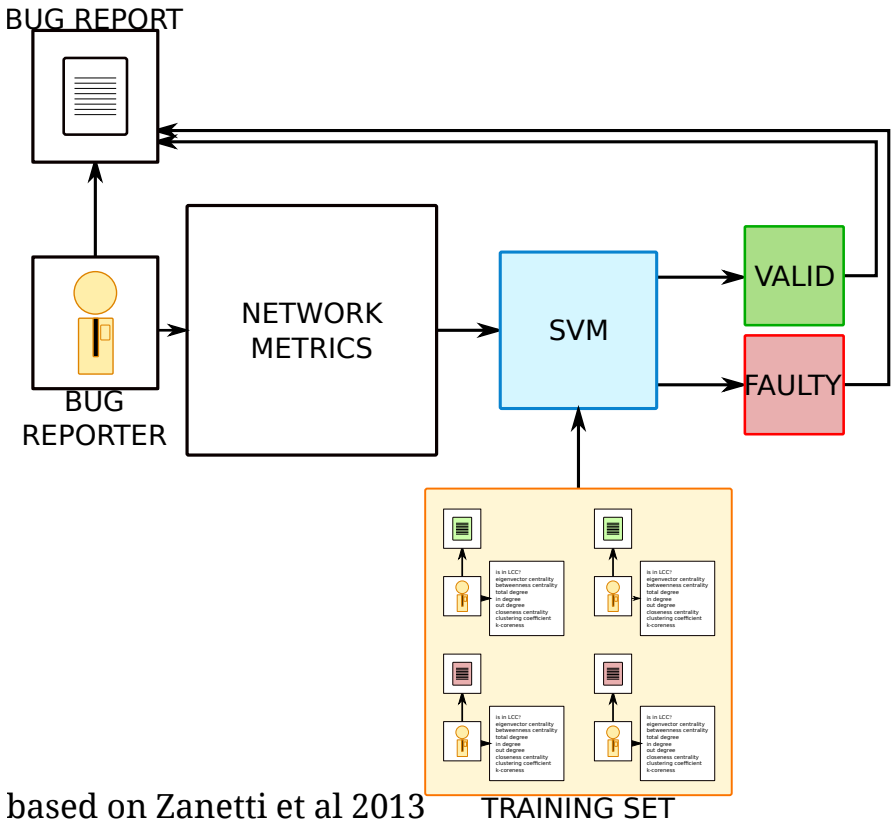


eigenvector centrality



clustering coefficient

Accurate Automatic Prioritization of Bug Reports



based on Zanetti et al 2013

max(precision)
VALID
90.3%

max(precision)
FAULTY
86.9%

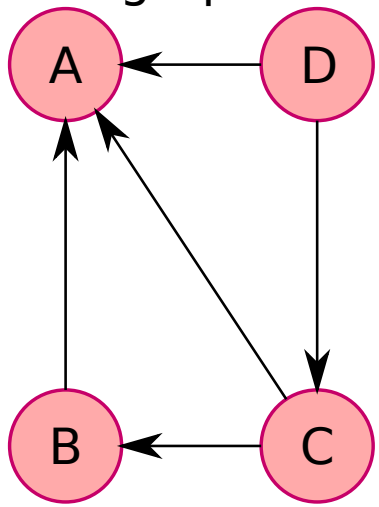
– Fascinating! Can you help me minimizing the actual number of bugs in the source code? I find humans to be very unreliable programmers ...

Source Code Can Also Be Seen as a Network

source code

```
class A {
        // definition of class A
};
class B {
        A* ab;
        // rest of definition of class B
};
class C {
        A* ac;
        B* bc;
        // rest of definition of class C
};
class D: public C {
        A* ad;
        // rest of definition of class D
};
```

graph

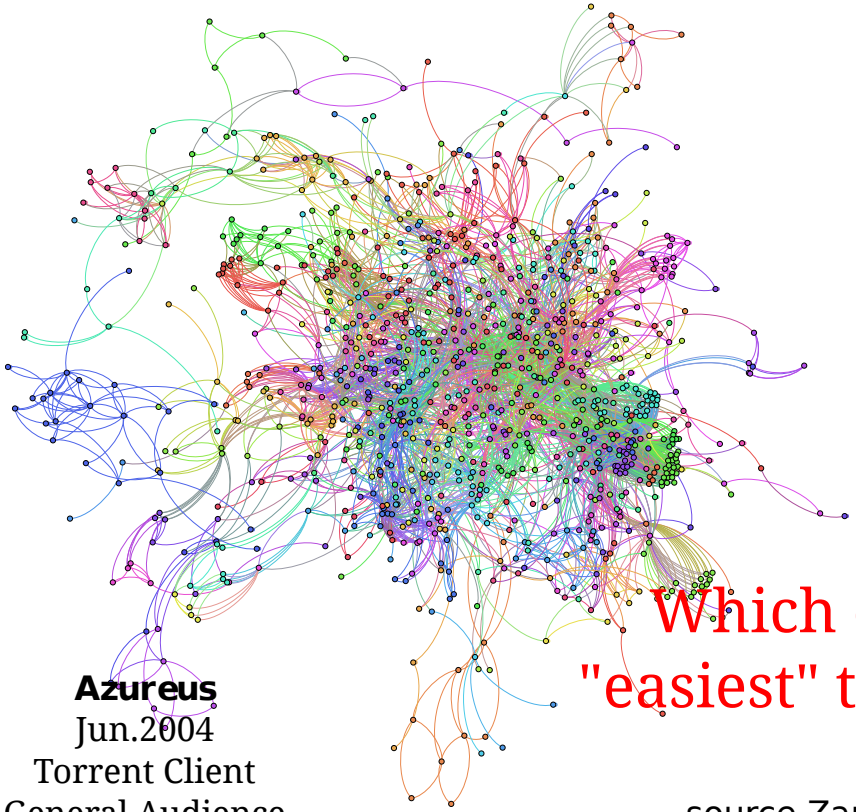


edge list:

B,A
C,A
C,B
D,C
D,A

based on Myers 2003

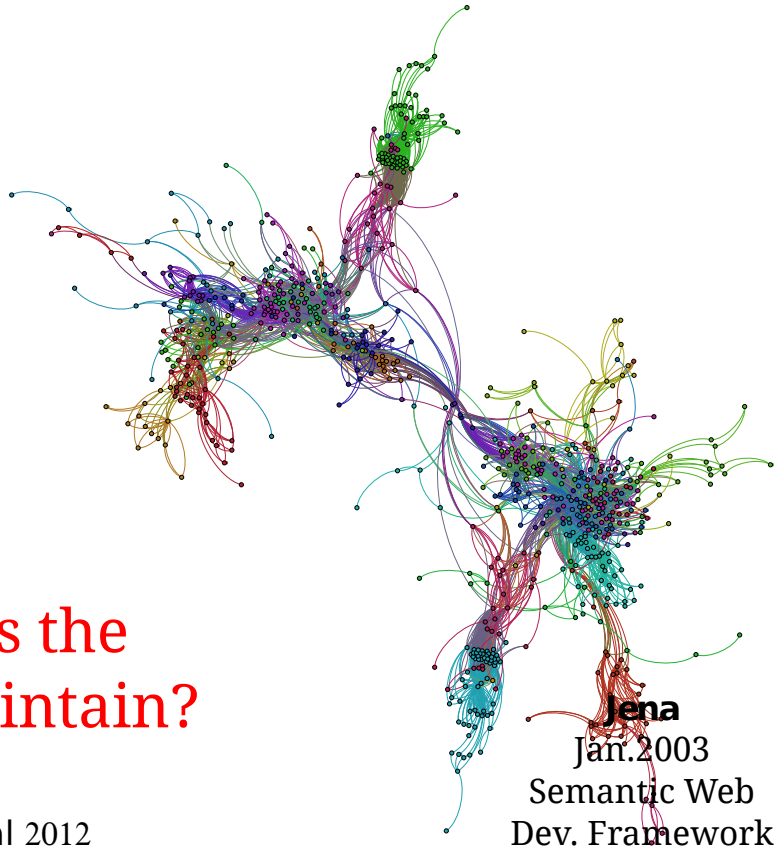
Source Code Can Also Be Seen as a Network



Azureus
Jun.2004
Torrent Client
General Audience

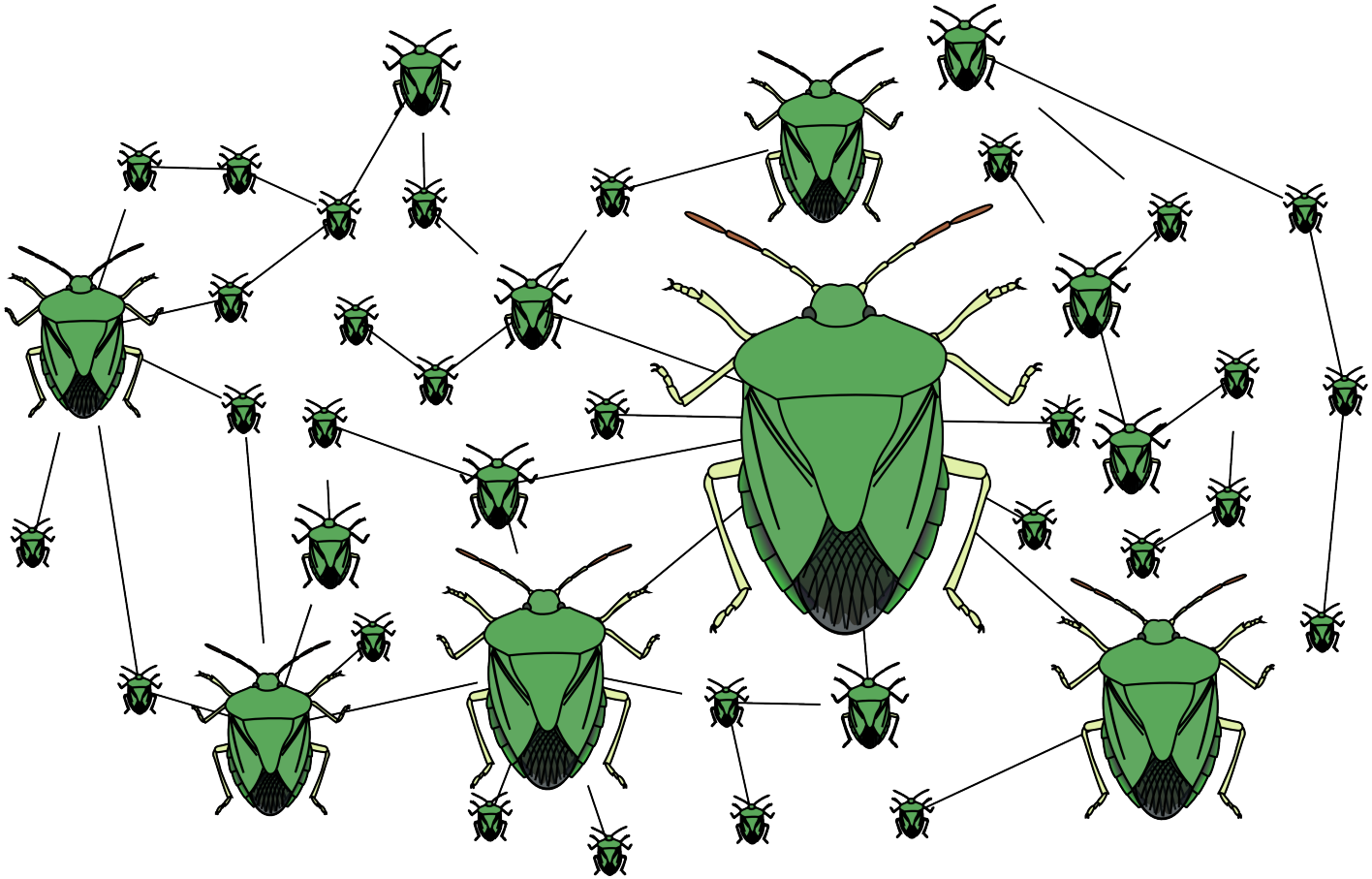
Which one is the
"easiest" to maintain?

source Zanetti et al 2012



Jena
Jan.2003
Semantic Web
Dev. Framework

Network Importance Influences Failure Severity



Bhattacharya et al (2012) show that the **importance of nodes** in a code network is indicative of the **severity of possible failures**

New Opportunities Ahead ... Go for It!

source StuckinCustoms 2007

Open Source Software for Network Analysis

- Libraries:
  NETWORKX (PYTHON)
  IGRAPH (PYTHON, R, C, RUBY)
- Visualization tools:
  GEPHI
  CUTTLEFISH
- Source code call graph generator:
  CODEVIZ
  DOXYGEN
  CDA

References

- Bhattacharya et al, *Graph-Based Analysis and Prediction for Software Evolution*, ICSE, 2012
- Migdal, *Visualizations for the Stackexchange*, project homepage, 2012
- Myers, *Software systems as complex networks: structure, function, and evolvability of software collaboration graphs*, Phys.Rev.E 68, 046116, 2003
- Newmann, *Networks: An Introduction*, Oxford Press, 2010
- Wilson et al, *Empirical Software Engineering*, online article, 2011
- Zanetti et al, *A Network Perspective on Software Modularity*, GI-Ed.-LNI, Proceedings P-200, ARCS 2012 Workshops, pp. 175–186, 2012
- Zanetti et al, *Categorizing Bugs with Social Networks: A Case Study on Four Open Source Software Communities*, under review, ICSE, 2013
- Zelkowitz, *Techniques for Empirical Validation*, Basili et al (Eds.): Empirical Software Engineering Issues, LNCS 4336, pp. 4–9, 2007