

Software Reflexion Models

David Notkin

Dept. of Computer Science & Engineering

University of Washington

www.cs.washington.edu/homes/notkin

But first!

- A little about UW CSE
- And a few other things



UW

- 34,000 students (25,000 undergraduates, 9,000 graduate and professional)
- 3,500 faculty (2,900 teaching, 600 research)
 - 100+ members of the national academies, 8 MacArthur Awards, 4 Nobel Prizes, ...
- For more than 20 years, UW ranked among the top five institutions in Federal research obligations (currently second behind Johns Hopkins)
- Third nationally in industrial R&D support, fifth in licensing revenues, ninth in private giving

Computer Science & Engineering

- Intercollege graduate program in 1967
- Departmentalized and added undergraduate program in 1976
- Moved to College of Engineering and added computer engineering program in 1989
- Added a Professional Masters Program (www.cs.washington.edu/masters) in 1996
- 30+ faculty, 30+ staff, ~200 graduate students, ~300 undergraduate students

Top 10 Department

- 15 current faculty hold NSF awards for young faculty (PVI, NYI, Careers)
 - 3 also hold Presidential Faculty Fellow Awards
 - 3 also hold ONR Young Investigator Awards
- 2 Sloans, 1 Packard, 8 Fulbrights, 2, Guggenheims, 10 ACM, 6 IEEE Fellows
- 3 UW COE Faculty Achievement Awards, 3 UW Distinguished Teaching Awards, 1 UW Annual Faculty Lecturer, 1 UW Outstanding Public Service Award
- Dick Karp
 - Turing Award, National Medal of Science, Harvard's Centennial Medal, Harvey Prize

Research areas include

- VLSI, Embedded Systems & CAD
- Computer Architecture
- Operating Systems, Networks & Communication
- Programming Systems
- Information Retrieval & Database Systems
- Software Engineering, Safety & HCI
- Computer Graphics and Computer Vision
- Artificial Intelligence
- Theory of Computation
- Computational Biology

2 undergraduate programs

- Computer Science (BS)
 - About 80 students admitted each year
 - Very competitive
- Computer Engineering (BS)
 - About 40 students admitted each year
 - We're proposing to double the size of the program
 - Highly competitive
- Both
 - 40 or fewer in most junior/senior classes
 - Capstone design courses (animation, digital design, etc.)
 - Lots of coops and interns
 - <http://www.cs.washington.edu/homes/lazowska/press/index.html#video>

Industrial affiliates program

- Support mutual needs of business, industry, and academia in computer research, development, and education
 - Accomplished by providing appropriate mechanisms for technical exchange and collaboration and employment of students
- Benefits to business and industry include
 - Influence research and education, participate in long-range technical assessments of problems and directions
 - Contacts with prospective employees
 - Early access to student resumes; special seminars and short courses can be arranged; etc.

More

- Over 70 member companies
- Benefits to UW include
 - Learning about current problems in industry
 - Students become acquainted with industrial needs; the coop program and summer internships provide students with a complementary element to their education
 - Results are greater excellence in both our research and teaching missions
- <http://www.cs.washington.edu/affiliates/>

Graduate programs

- Research-oriented PhD/MS program
 - ~120 Ph. students, ~20 MS students
 - Graduate about 15 PhD students & 20 MS/year
 - Highly competitive: 10-15% accepted
- Professional Masters Program
 - "Accessible": evening and some distance learning
 - Archived video and slides: see the UW CSE courses web page
 - Aimed at working professionals with a CS background and experience in the field
 - About 120 FTEs
 - Coursework only

ACM SIGSOFT

- One of ACM's largest special interest groups
- Sponsor of several ongoing conferences including
 - International Conference on Software Engineering (joint with IEEE TCSE)
 - Symposium on the Foundations of Software Engineering
 - International Symposium on Software Testing and Analysis
 - Symposium on Software Reuse
- Bimonthly newsletter, SEN
- Involved in "software engineering as a profession" issues
- www.acm.org/sigsoft

Software Reflexion Models

Joint with Gail C. Murphy
(University of British Columbia)

<http://www.cs.ubc.ca/spider/murphy/software/rmtool/index.html>

Software Evolution

- Change is inevitable...
 - ... and becomes increasingly more difficult over time
- "As a large program is continuously changed, its complexity, which reflects deteriorating structure, increases unless work is done to maintain or reduce it."

—The Second Law of
Program Evolution
Lehman and Belady, 1978

Changes Become Harder

- Limited predictive powers of the kinds of future changes

"There is no reason anyone would want a computer in their home."

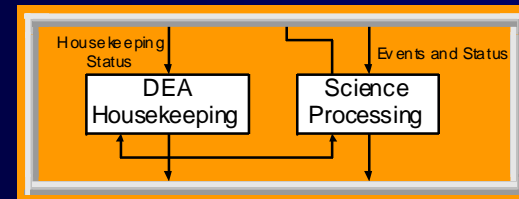
— Ken Olson, founder of Digital Equipment Corp., 1977

- (Very) limited knowledge of the current system

Software Structure

- "Any system consists of parts, such as modules, procedures, classes and methods. The *structure* of the system is the *organization* and *interactions* of those parts."

— Harold Ossher, 1987



```
#ifdef _POSIX_SOURCE
#include <sys/wait.h>
#endif

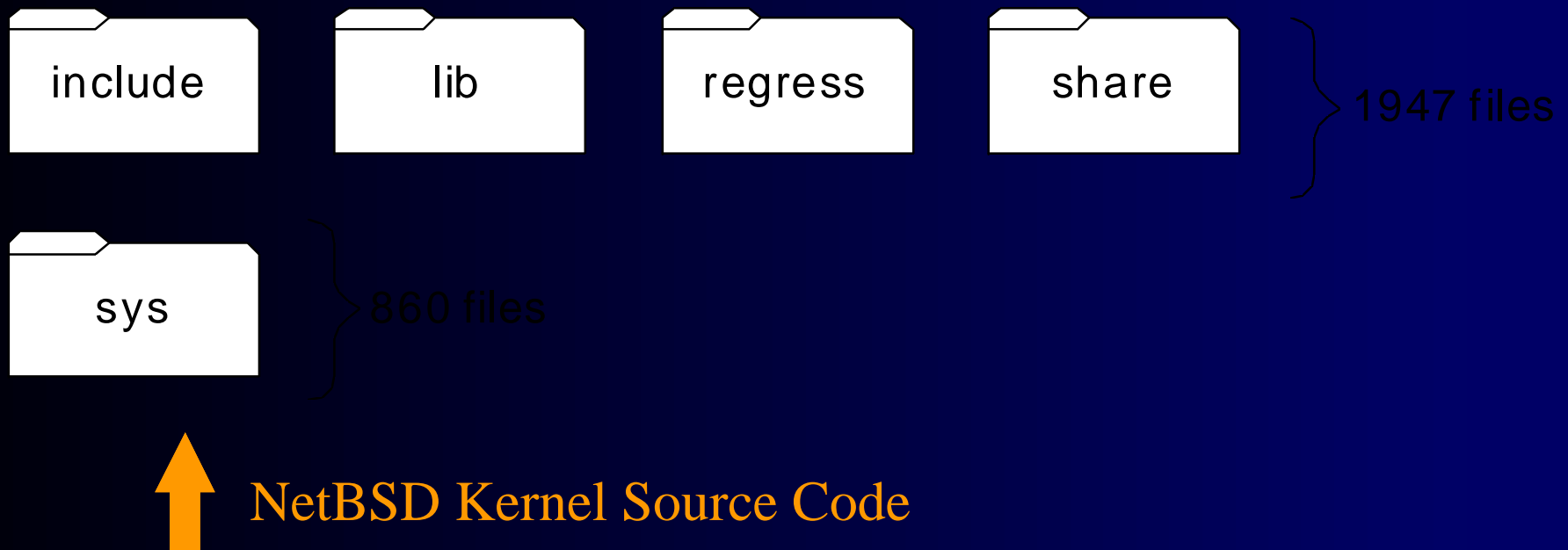
void main () {
```

Overview

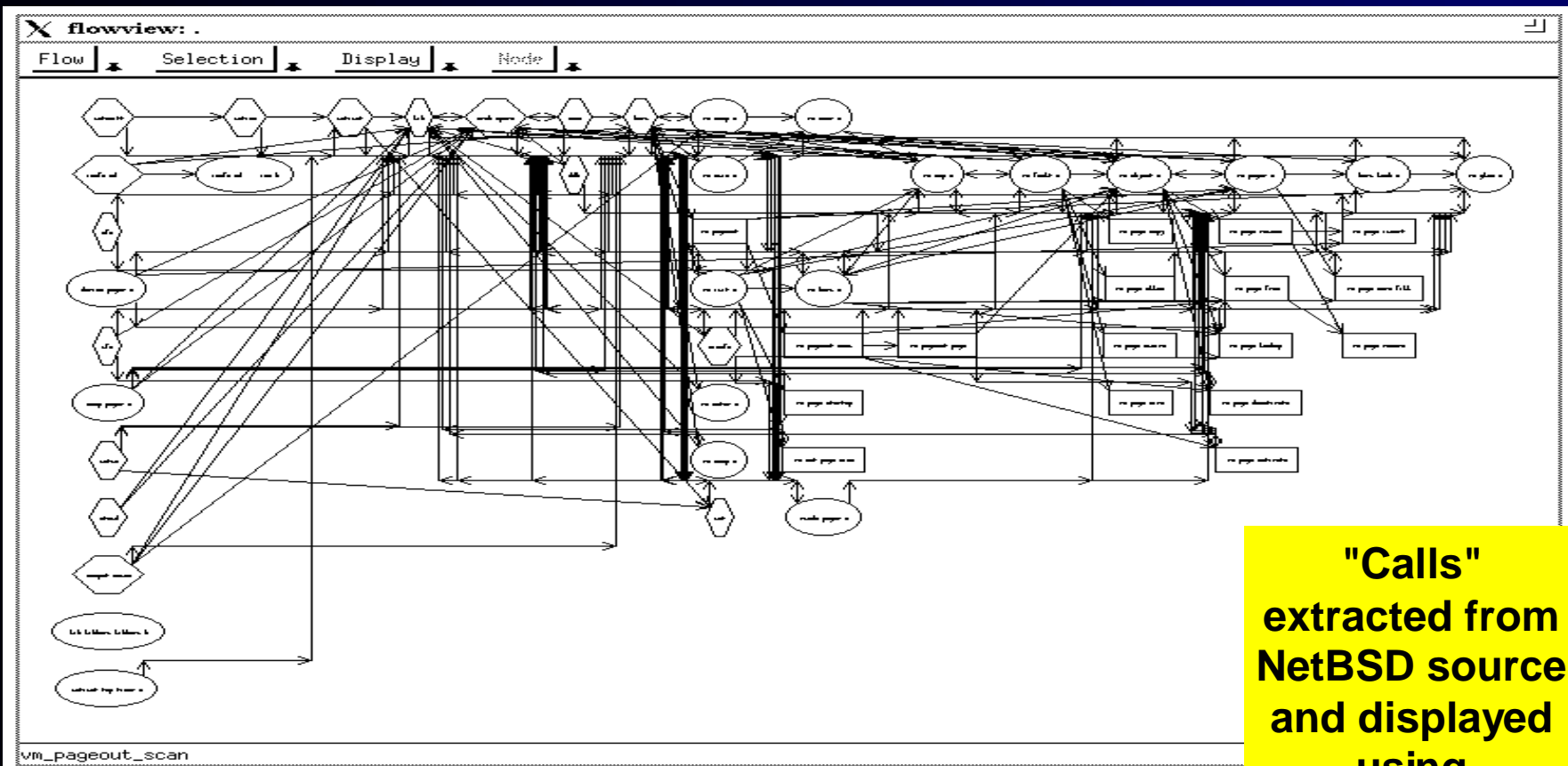
- A Typical Scenario
- Lightweight Structural Summarization
 - Software Reflexion Model Technique
 - Lexical Source Model Extraction Technique
- Summary

A Typical Scenario

- You are asked to provide, within five days, an estimate of the effort required to modify an implementation of a Unix operating system to page over a distributed network

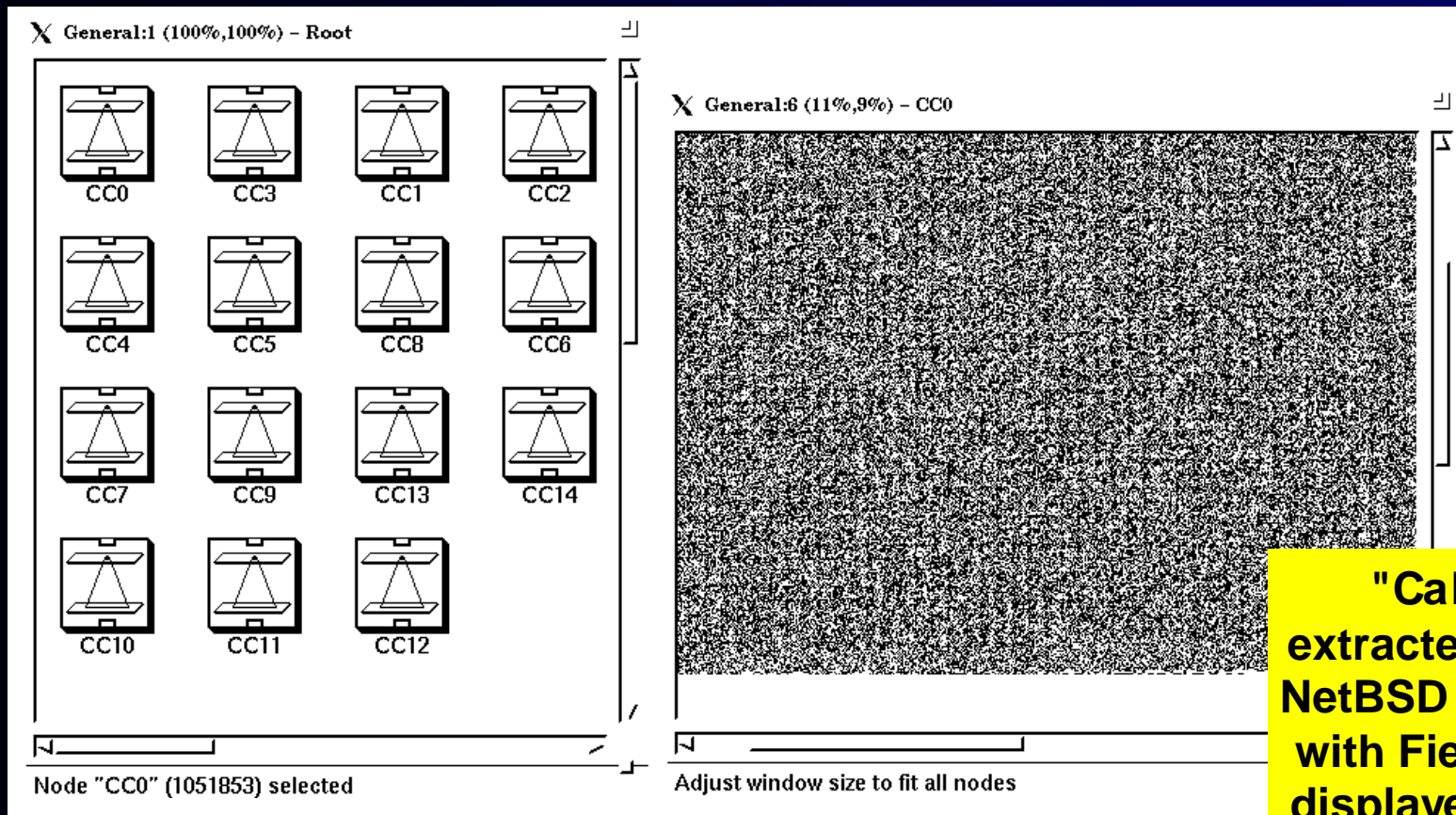


Software Visualization

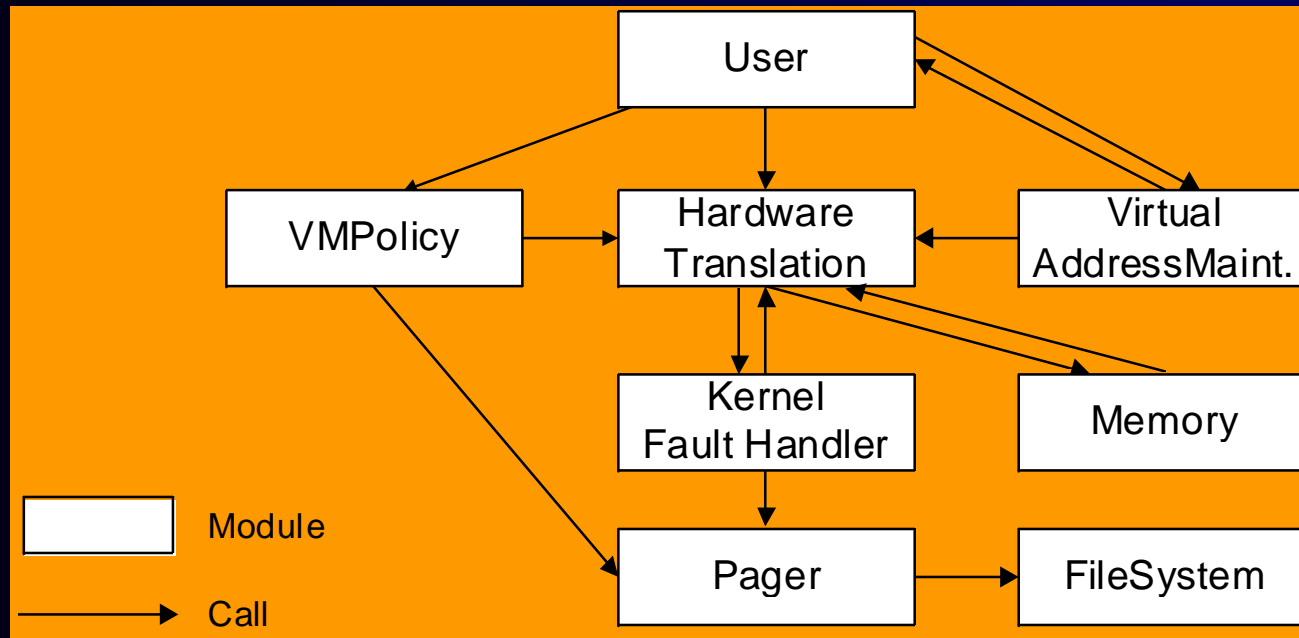


"Calls"
extracted from
NetBSD source
and displayed
using
Field [Reiss95]

Reverse Engineering



Boxology

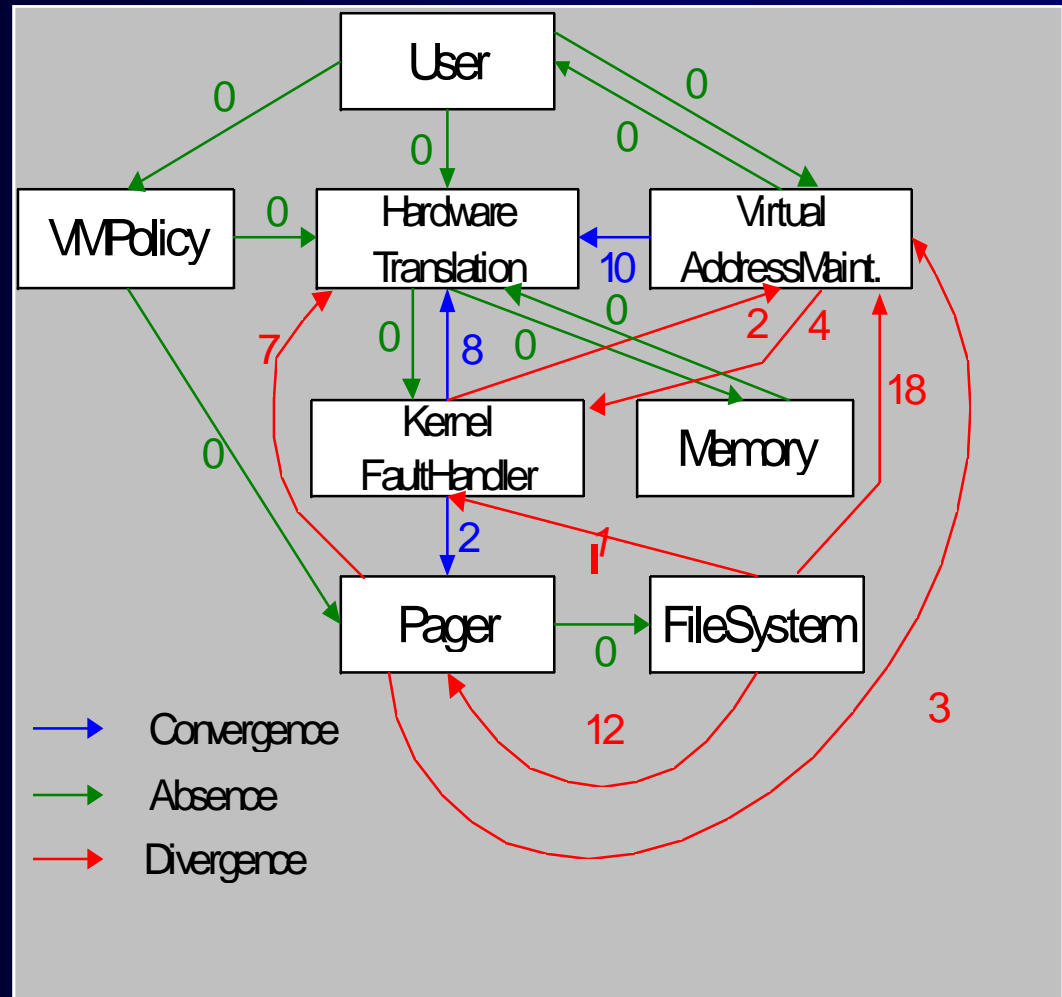
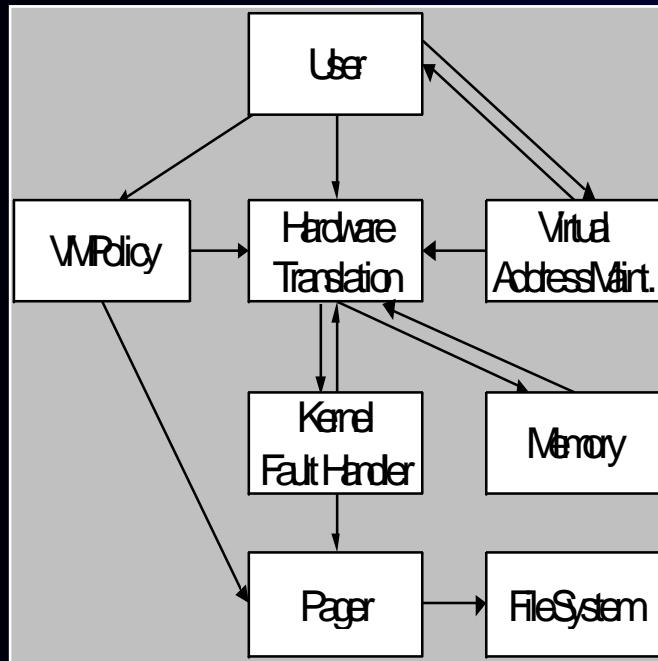


- Model of a Unix virtual memory subsystem drawn by a domain expert

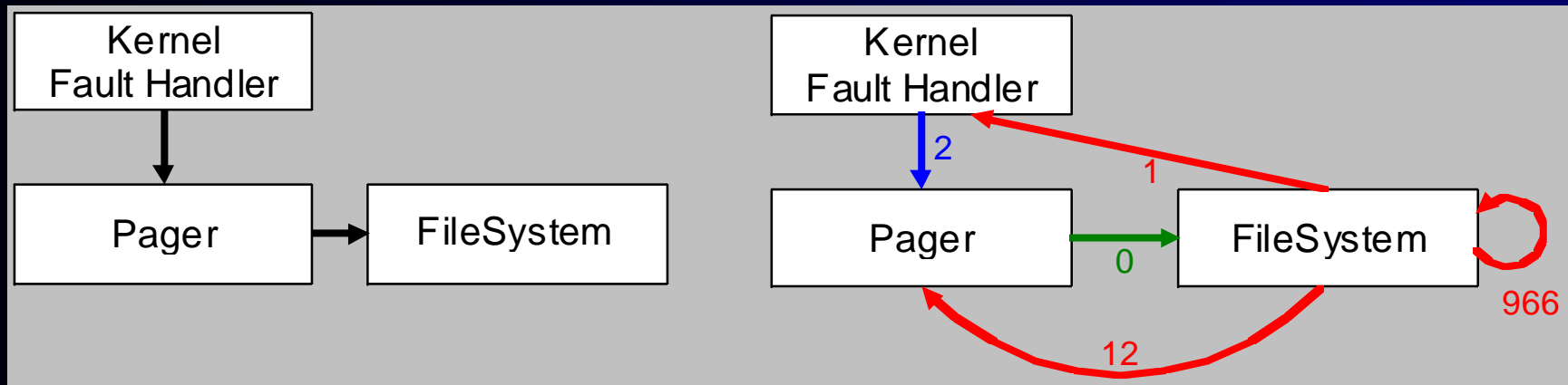
A Problem

- These models are often inaccurate with respect to the system's artifacts (including the source code)
- This may lead to budget and schedule overruns, missed opportunities for reuse, etc.

Software Reflexion Model

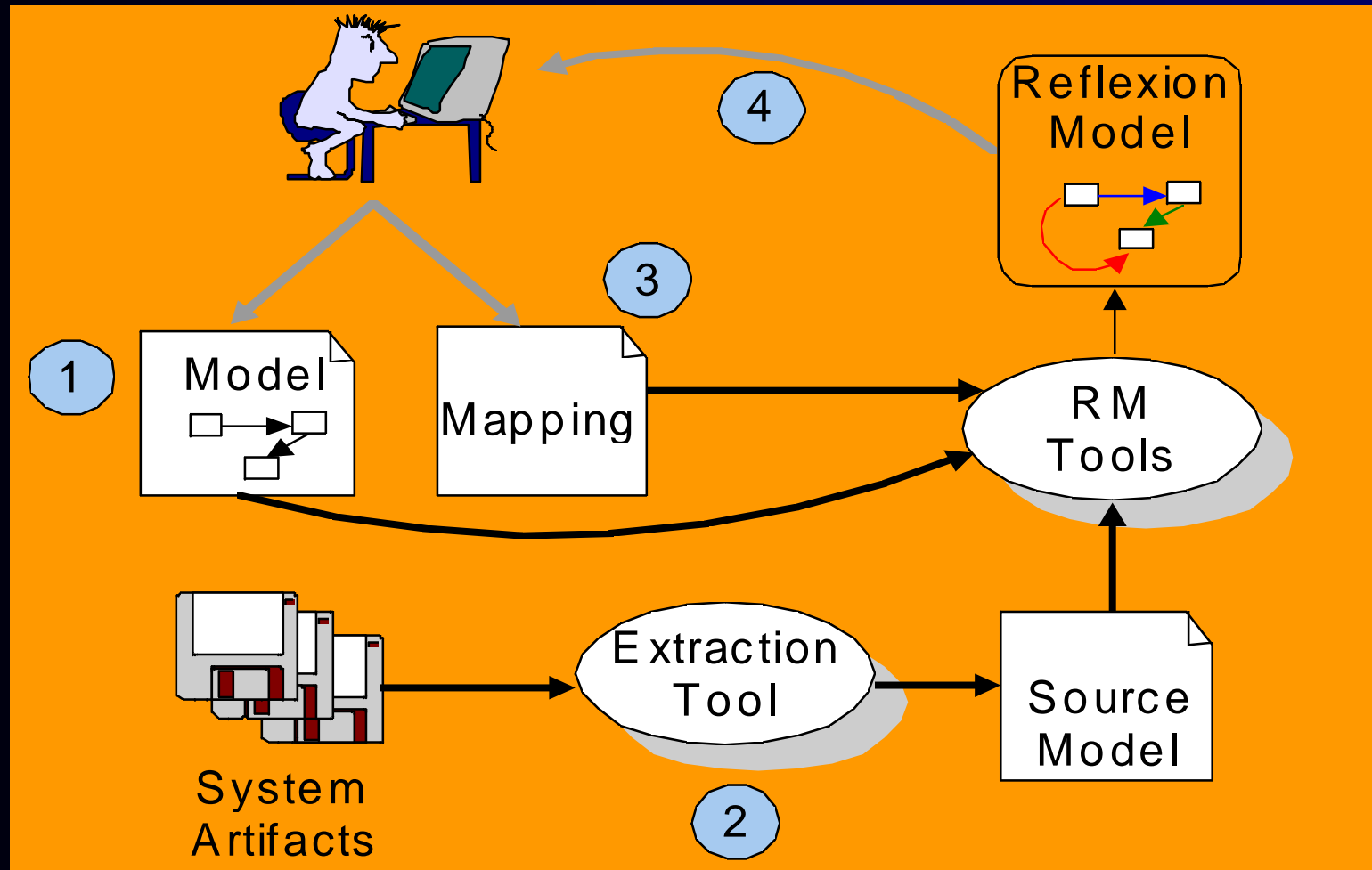


Terminology

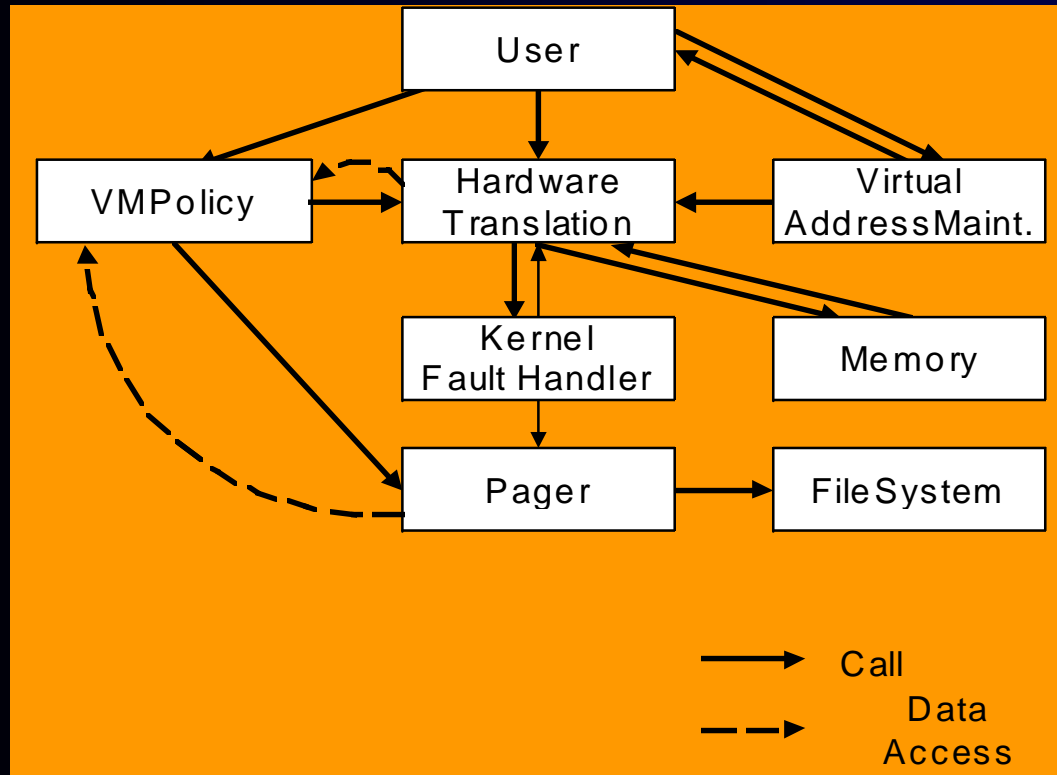


- **Convergence:** high-level model predicts source interaction
- **Divergence:** source has unpredicted interaction
- **Absence:** high-level model predicts interaction not found in source

The Technique



1. State a High-Level Model

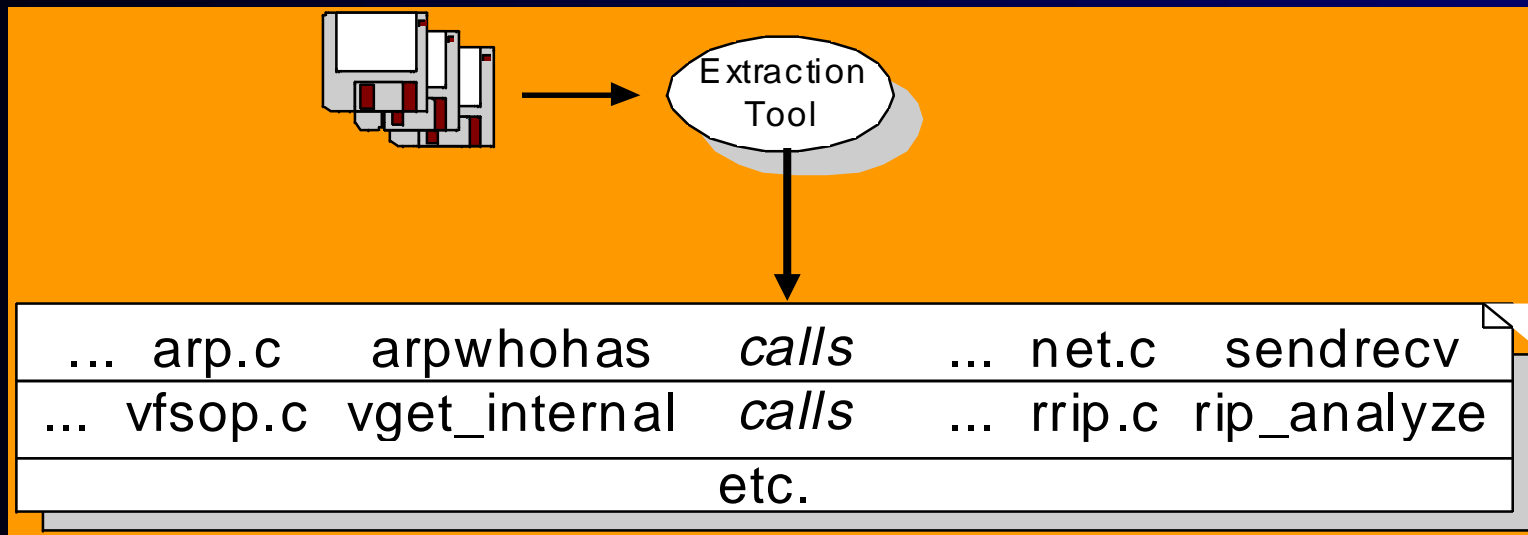


- syntactic

- multiple relations

- "everyone has one or more"

2. Extract a Source Model



- use existing tools (e.g., cflow, Field, etc.)
- may contain multiple relations

3. State a Mapping

Source Model Entities

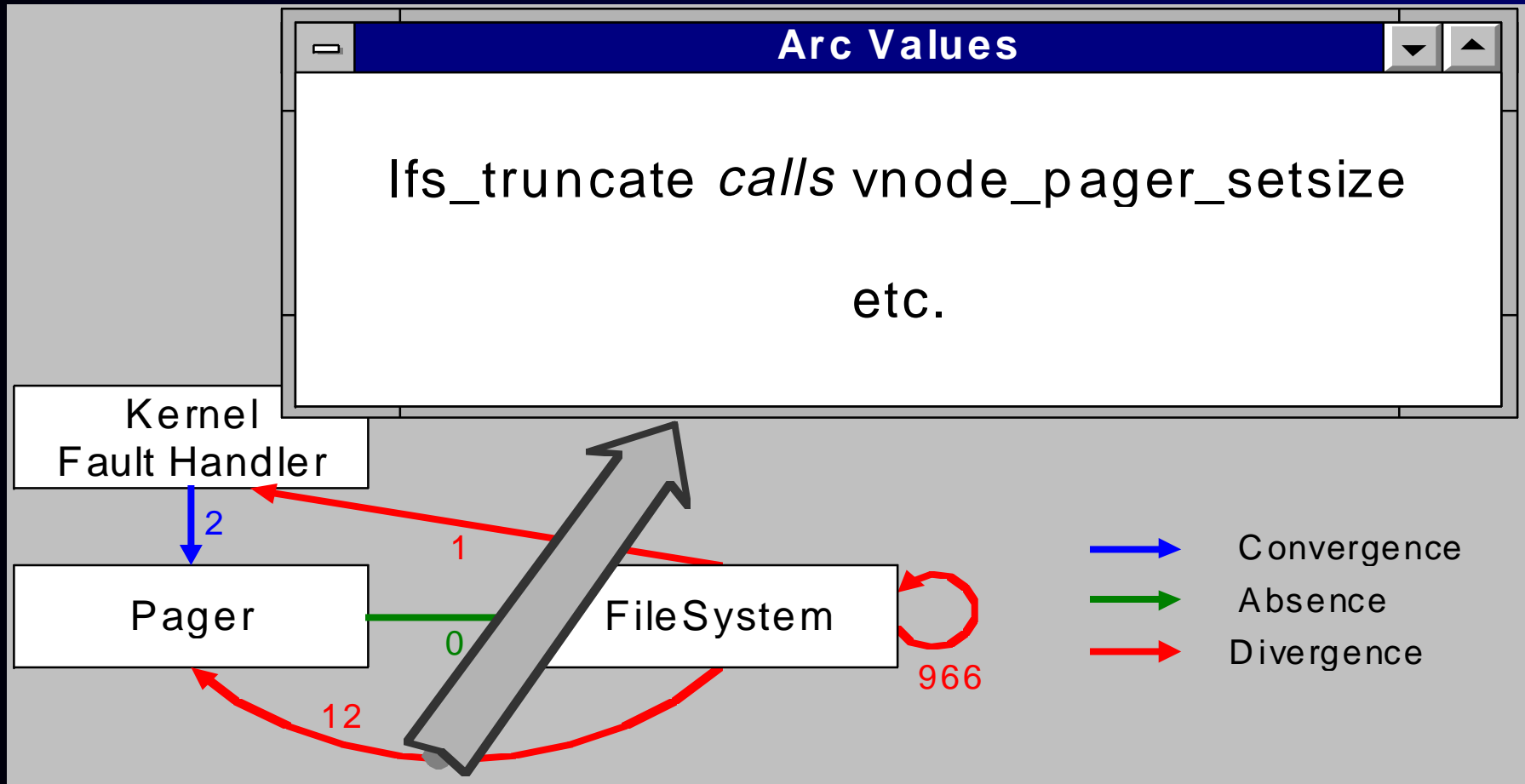
```
file=pager.c  
file=vm_map.*  
dir=vm func=active
```

High-Level Model Entities

```
Pager  
VirtualAddressMaint.  
VMPolicy
```

- name source model entities using:
 - physical and logical software structure
 - regular expressions
- many-to-many mapping

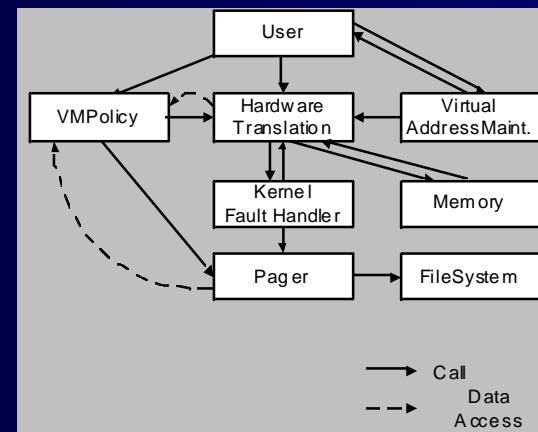
4. Investigate RM



Iteration

- want to investigate the data relationships?

- augment the source model
- update the mapping:
`var=queue.*active`
`VMPolicy`
- recompute...



Experience

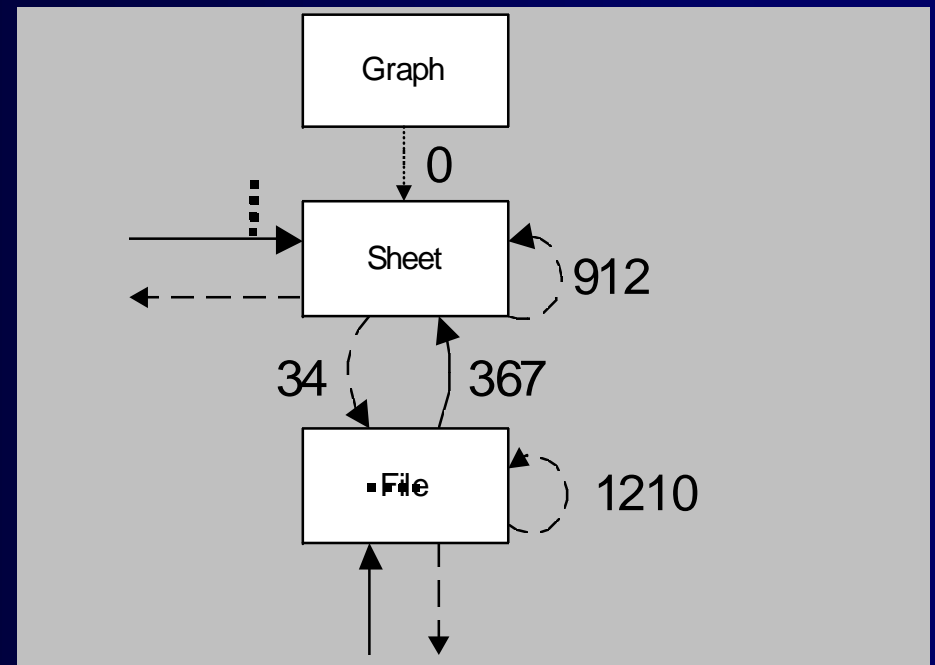
compiler	4,000	lines of C++
	4,000	lines of Ada
restructuring tool	30,000	lines of CLOS
<i>SPIN</i>	65,000	lines of Modula-3
NetBSD	250,000	lines of C
industrial audibles	6,000	lines of C++
Excel	>1,000,000	lines of C

Excel

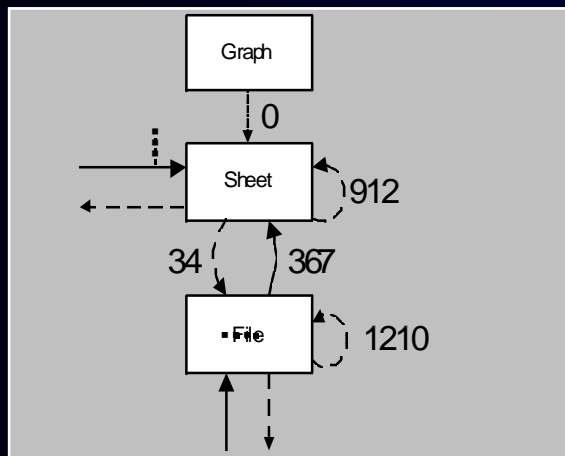
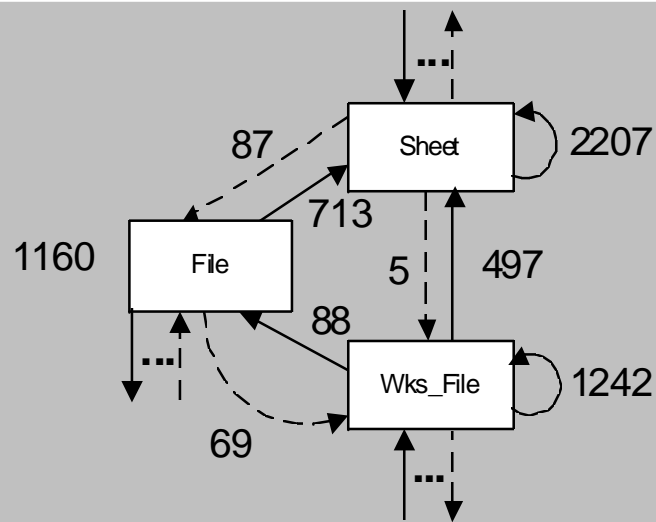
- Experimental reengineering to identify and extract component
- Microsoft engineer computed Reflexion Models several times a day for four weeks
 - 120,000 calls and global variable references
 - map file with over 1000 entries
 - high-level model with 15 entities and 96 interactions
 - 4 minutes to compute on a 486
- Some lessons learned:
 - map files evolved to be larger than expected
 - scale places pressure on managing the information

An initial Reflexion Model

- The initial Reflexion Model computed had 15 convergences, 83, divergences, and 4 absences
- It summarized 61% of calls in source model



A refined Reflexion Model

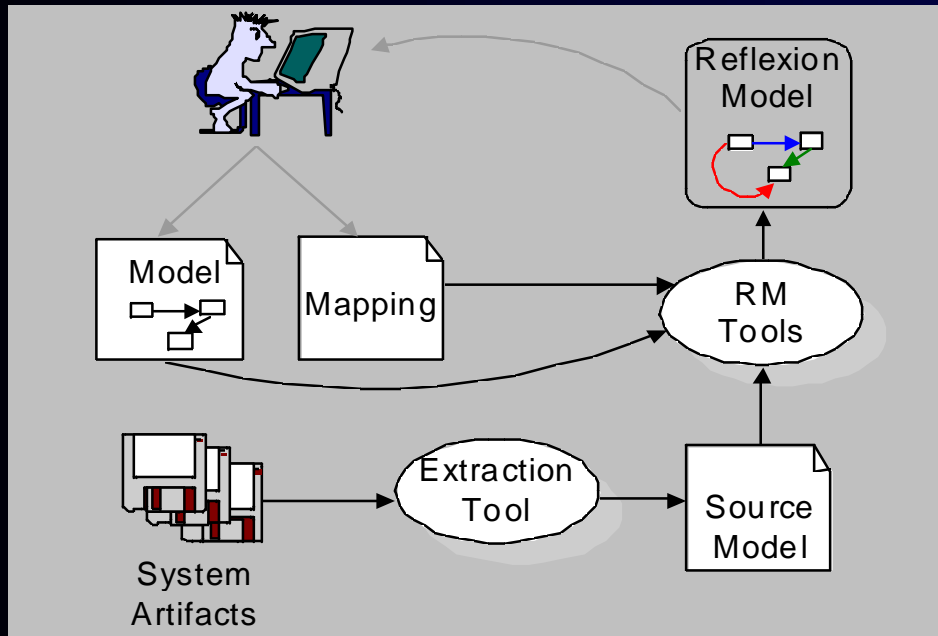


- A later Reflexion Model summarized 99% of 131,042 call and data interactions
- This approximate view of approximate information was used to reason about, plan and automate portions of the task
- Estimated two years using other tools

Other Features...

- family of reflexion model systems
- parameterized by structural descriptions
- incremental computation algorithms
- tagging and annotations to manage investigation

Summary



"Definitely confirmed suspicions about the structure of Excel. Further, it allowed me to pinpoint the deviations. ... very easy to ignore stuff that is not interesting and thereby focus on the part of Excel I want to know more about."

— Microsoft Engineer

Two other topics

- Symbolic model checking of large software specifications
 - Verification technique for improving confidence in the correctness of finite state specifications (such as those written in Statecharts)
 - Applied to two industrial strength specifications
 - TCAS and a research prototype of a Boeing avionics fault-tolerant electrical power distribution system
- Dynamic extraction of program invariants

Questions?

- Reflexion models
- UW CSE
- Anything else

