

CSSE 490 Model-Based Software Engineering: More Automatic Programming



Shawn Bohner

Office: Moench Room F212

Phone: (812) 877-8685

Email: bohner@rose-hulman.edu



ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

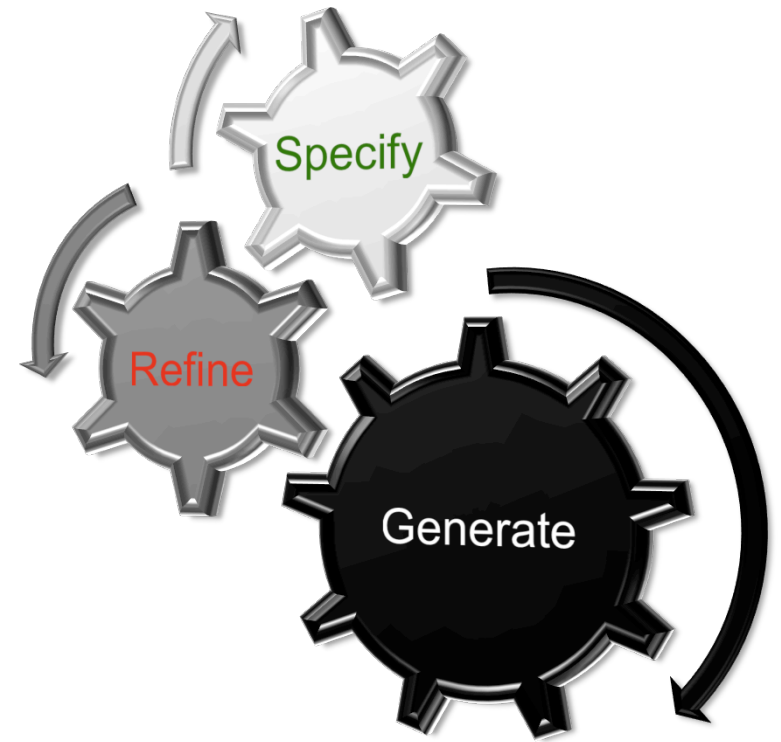
Sort of like this class... ☺



Learning Outcomes: MBE Discipline

Relate Model-Based Engineering as an engineering discipline.

- Discussion of Milestone 3
- Origins of Automatic Programming
- Exercise with Project
- Example: DRACO
- Automatic Programming in the Future (if time)



When do you think that computers will be intelligent enough to program themselves? Why?

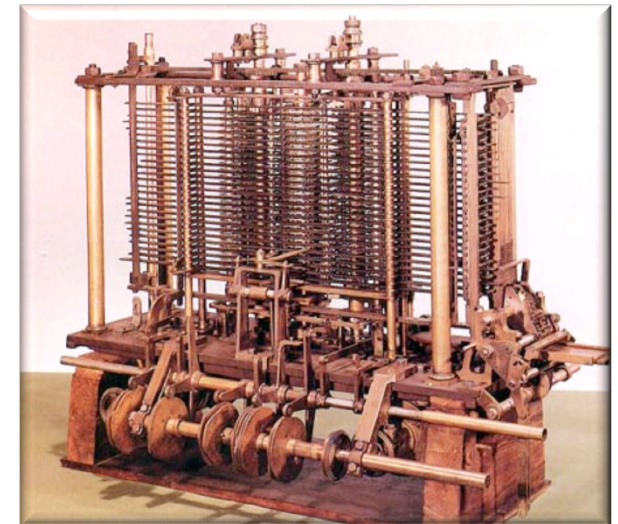
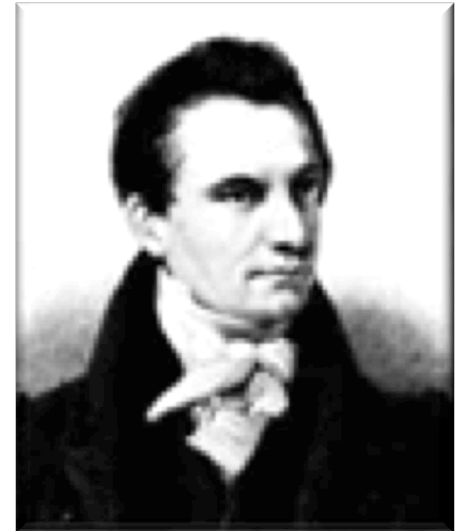
- Think for 15 seconds...
- Let's talk...



Charles Babbage (1791-1871) 1/2

- Babbage's two computer designs:
 1. Difference engine, that did computations well
 2. Programmable analytical engine, with
 - Punched card programs,
 - A three-address instruction set, and
 - A memory to hold variables

- Babbage's computing goals largely realized
 - Still need better algorithms and faster machines



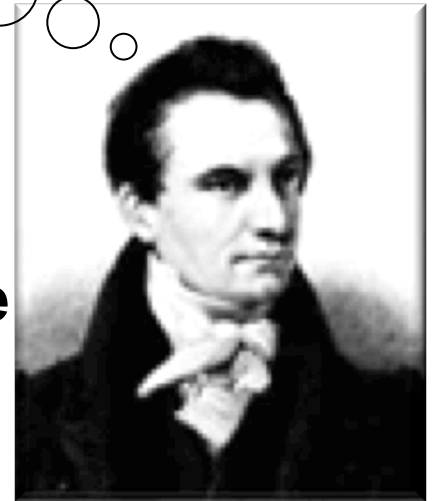
Charles Babbage (1791-1871) 2/2

What will happen when:

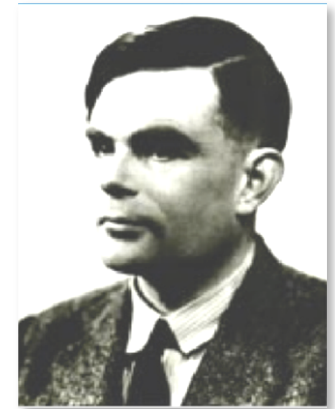
- Computers are free and infinitely powerful?**
- Bandwidth and storage is free and infinite?**

■ **Remaining limits:**

- Content (core asset of the web)**
- Software: Bugs: >\$100 per line of code**
- Operations: > \$1,000/node/year**



Alan M. Turing (1912-1954) 1/2



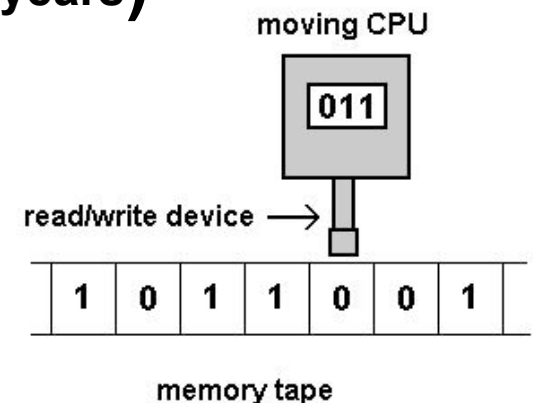
■ Turing Machine

- Theoretical device that manipulates symbols on a strip of tape according to a table of rules
- Turing machine can be adapted to simulate the logic of any computer algorithm, and is useful in explaining the functions of a CPU inside a computer

■ Computers will be intelligent (in 50 years)

■ Will this just be a symbiotic relationship (computer as tool)?

- Or will computers be “conscious”?



Alan M. Turing (1912-1954) 2/2



■ Imitation Game

- Judge, man, and a woman
- All talk over communications line (email or chat)
- Man impersonates a woman
- Woman tries to help judge
- Judge must identify man after 5 minutes



■ Turing Test

- Replace man or woman with a computer
- Fool judge 30% of the time



Vannevar Bush (1890-1974)

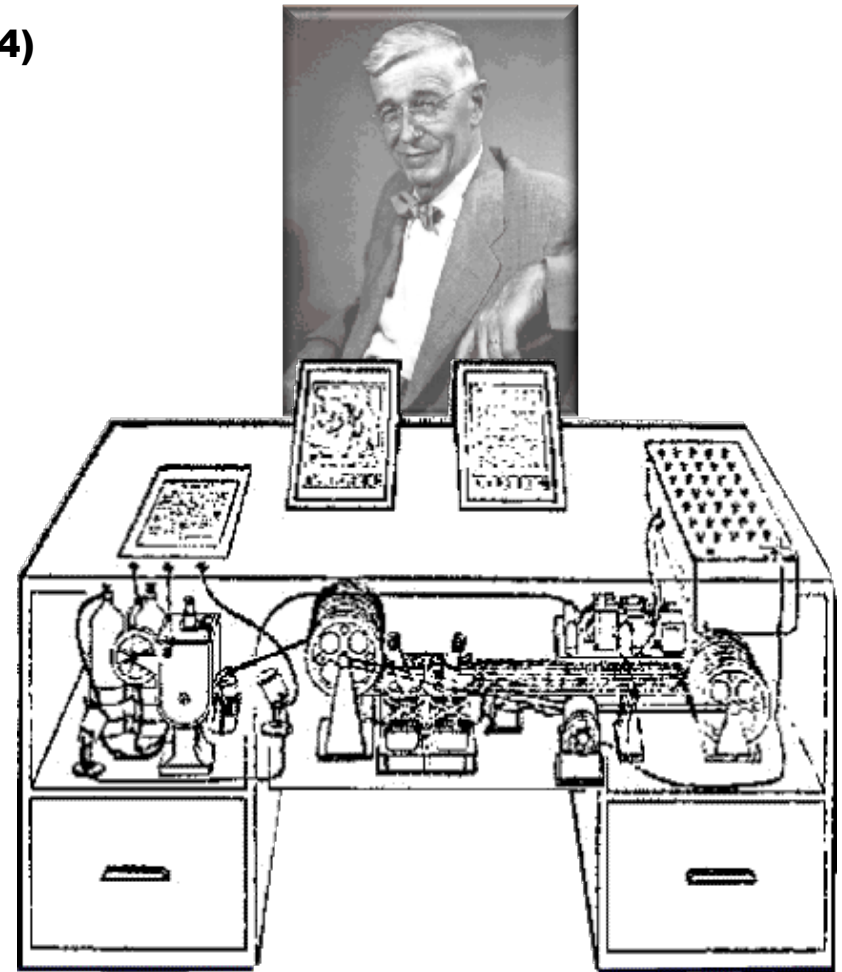
■ Memex

- All human knowledge in Memex “a billion books” hyper-linked together

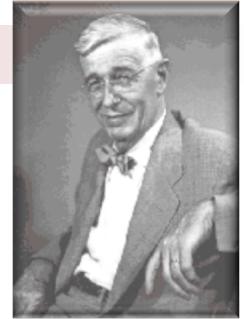
■ Record everything you see

- Camera glasses
- “A machine which types when talked to”

■ Navigate by text search following links associations



Memex is Nearly Here!



■ Internet growing quickly

- Most scientific literature is online
- Most literature is online (but copyrighted)
- Most Library of Congress visitors: web

■ Problems Bush anticipated:

- Finding answers is hard
- Collection is hard
- Veracity is an issue

Everything!
Recorded

All Books
MultiMedia

All LiboC
books (text)

.Movie

A Photo

A Book

Yotta

Zetta

Exa

Peta

Tera

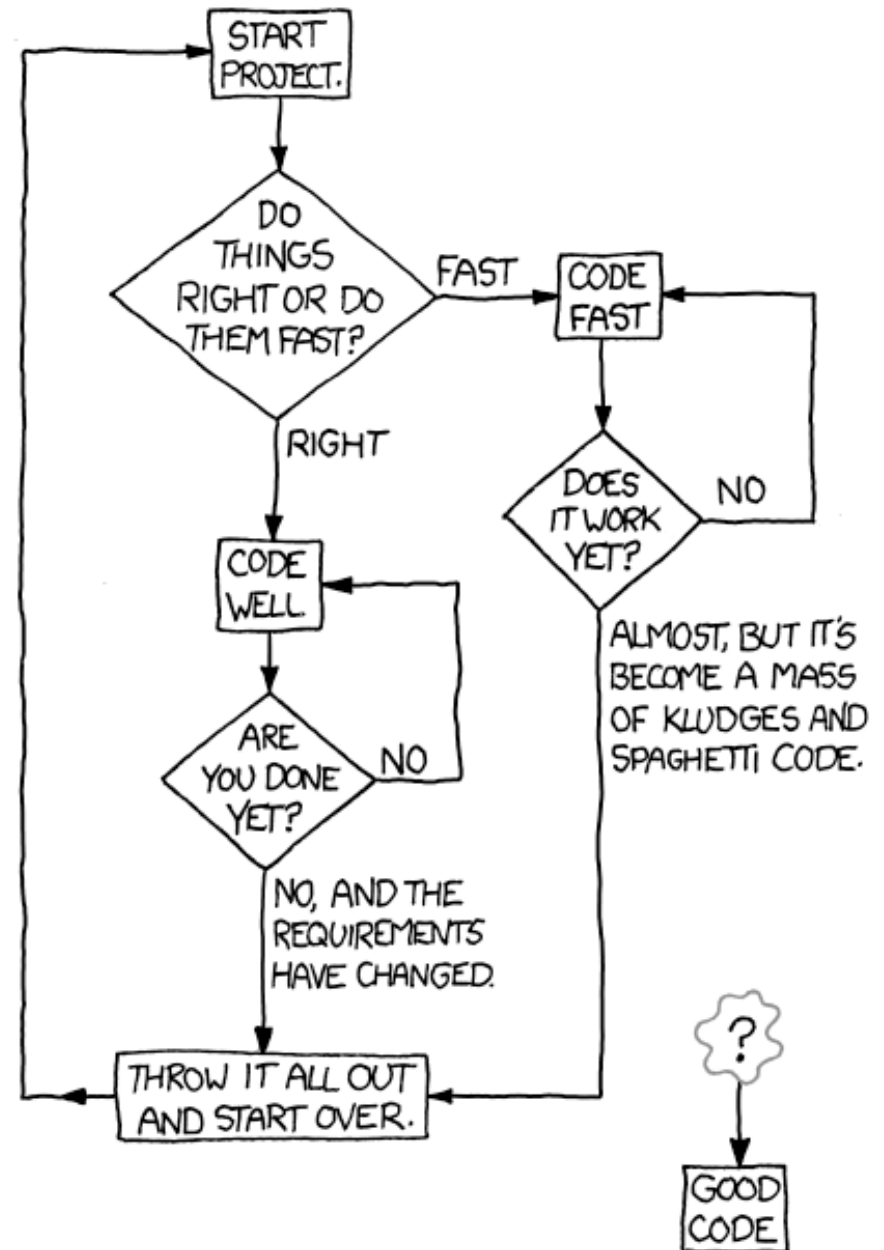
Giga

Mega

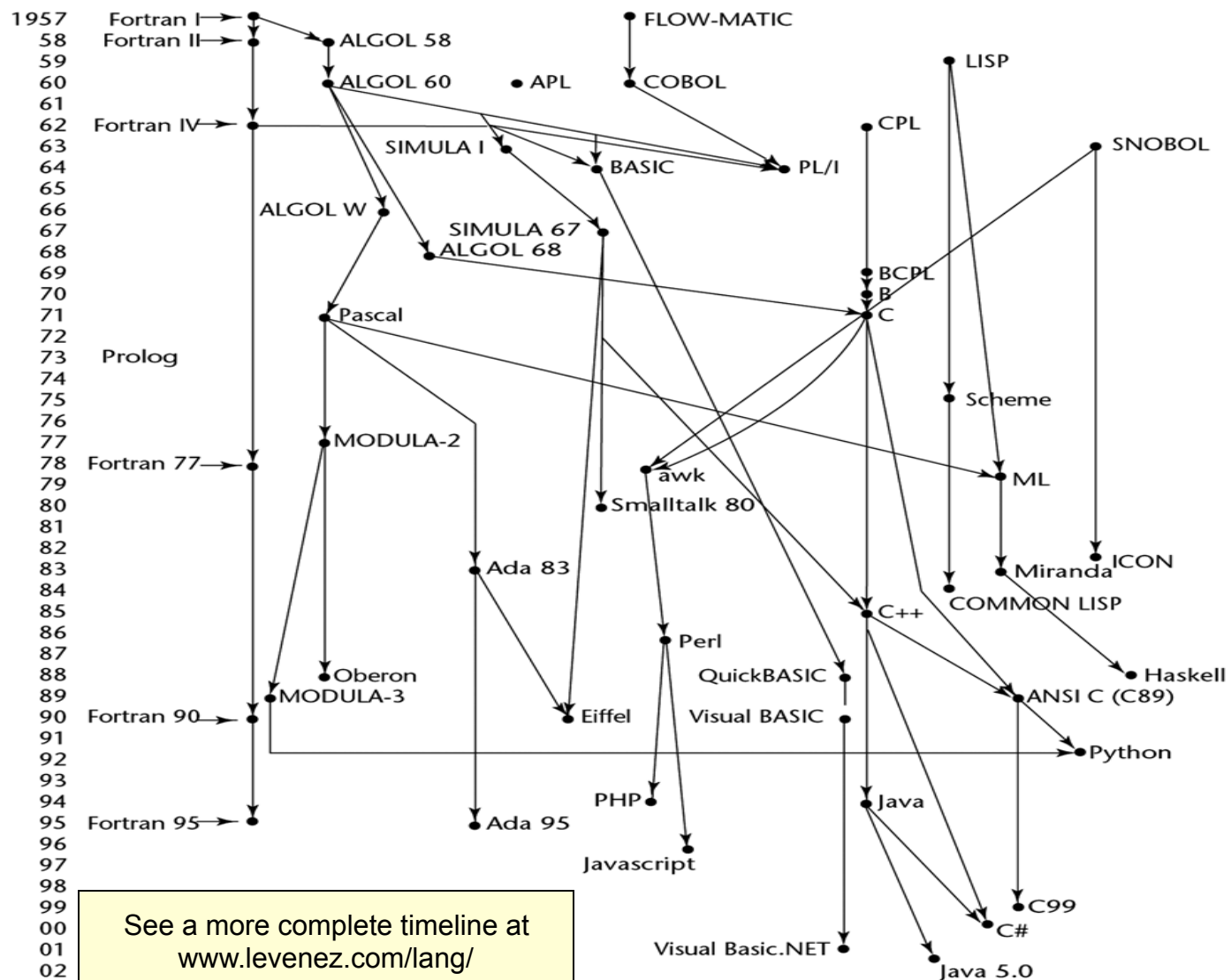
Kilo

Pessimists view: So, how does this get automated?

HOW TO WRITE GOOD CODE:



Genealogy of Common Languages

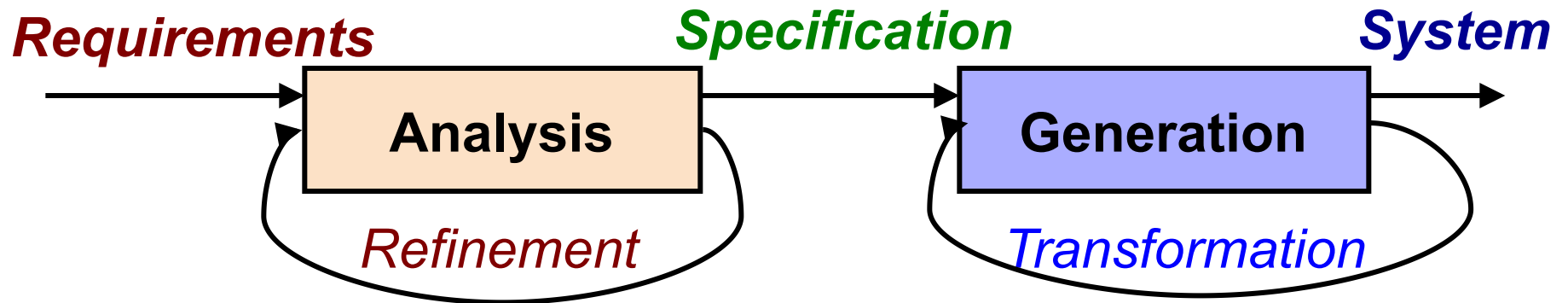


Advancing with Automatic Programming

- **Given a specification, build a system that implements it**
 - Prove implementation matches the spec.
 - Do it better than a team of programmers
- **Devise a specification language or user interface:**
 - That is easy for people to express designs (say about 1,000X easier)
 - That computers can compile
 - That can describe all applications (is complete)



Recall: Automatic Programming

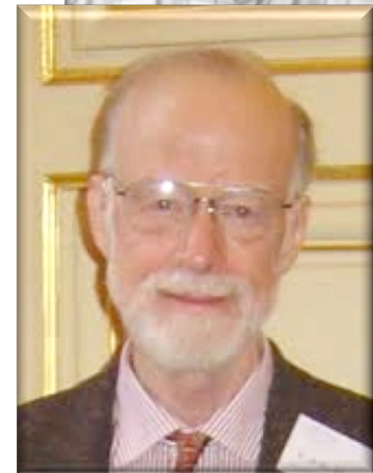
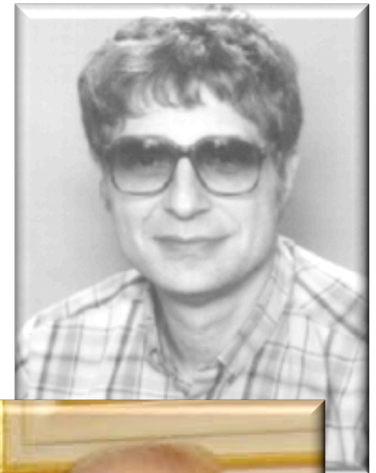


Elaborate from abstract to Concrete

Automatic Programming System Should Reason about Application

- Determine best fit designs and tradeoffs
- Ask about exception cases
- Ask about incomplete specification

But not be onerous





Put these together for MBSE

- **Turing test:** win impersonation game 30% of the time
 - Read and understand as well as a human
 - Think and write as well as a human
- **Memex:** Build a system that, given a text corpus, can answer questions about the text and summarize it as quickly and precisely as a human expert
 - Then add sounds: conversations, music...
 - Then add images, pictures, art, movies...
- **Automatic Programming:** Given a specification, build a system that implements the it
 - Prove that the implementation matches the spec.
 - Do it better than a team of programmers

Exercise: Reflect on Class Project

Describe how Milestone 3 Environment Automates Software Development.

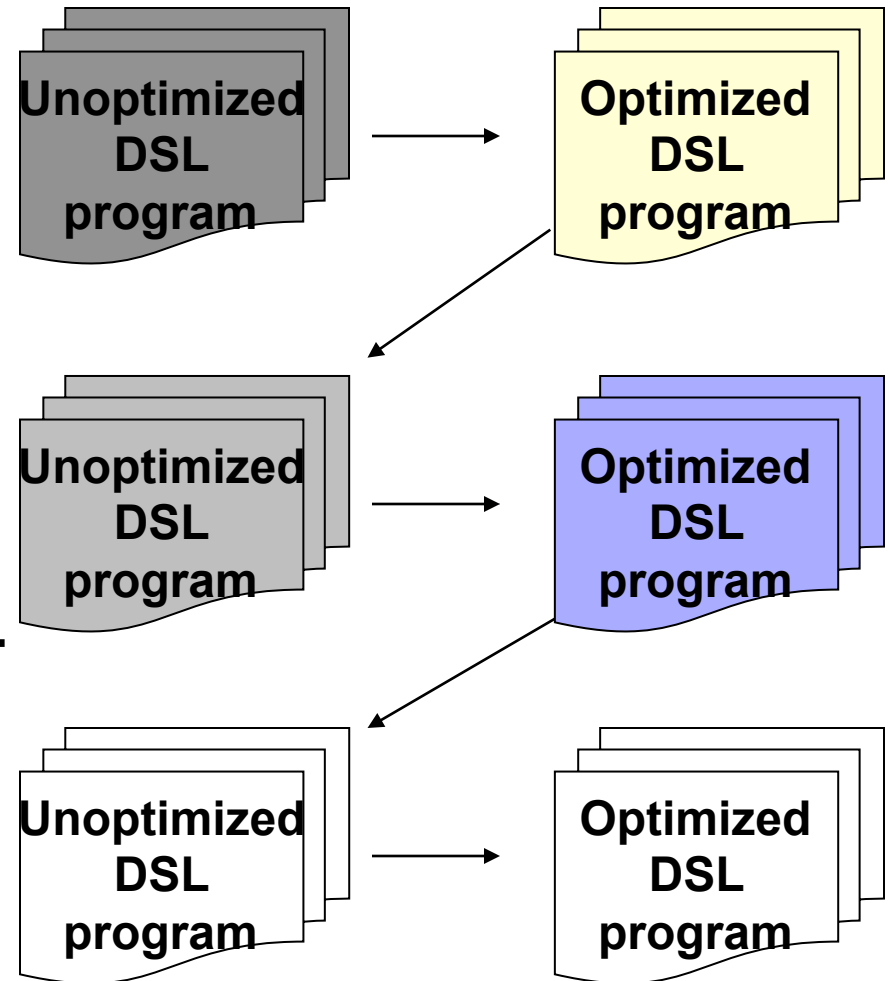
- What are the assembling steps?
- What are the transformational steps?
- Is the domain language a separate concern?
- Draw a diagram of how it works?



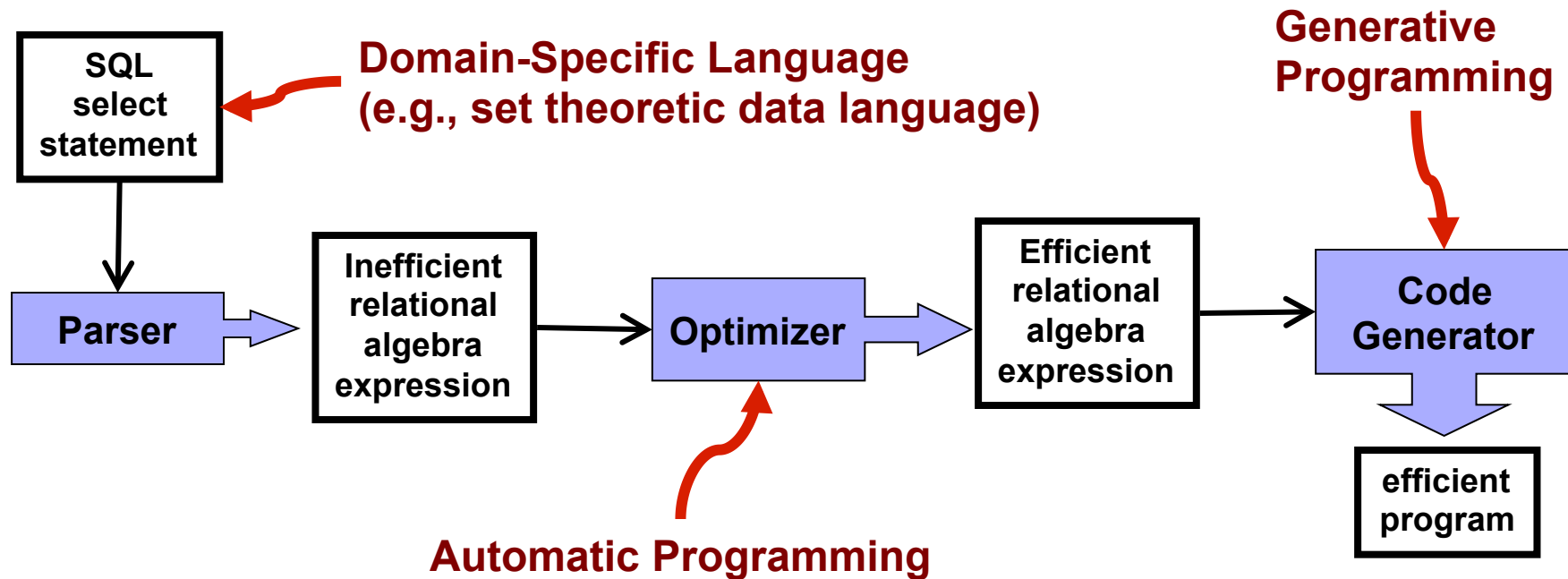
Early Generative Reuse System

DRACO – Jim Neighbors (1984)

- Programs are written in domain-specific languages
- Optimize DSL programs (because the domain abstractions are visible)
- Translate DSL to another, lower-level abstraction DSL and do the same
- Process repeats until you get to machine code

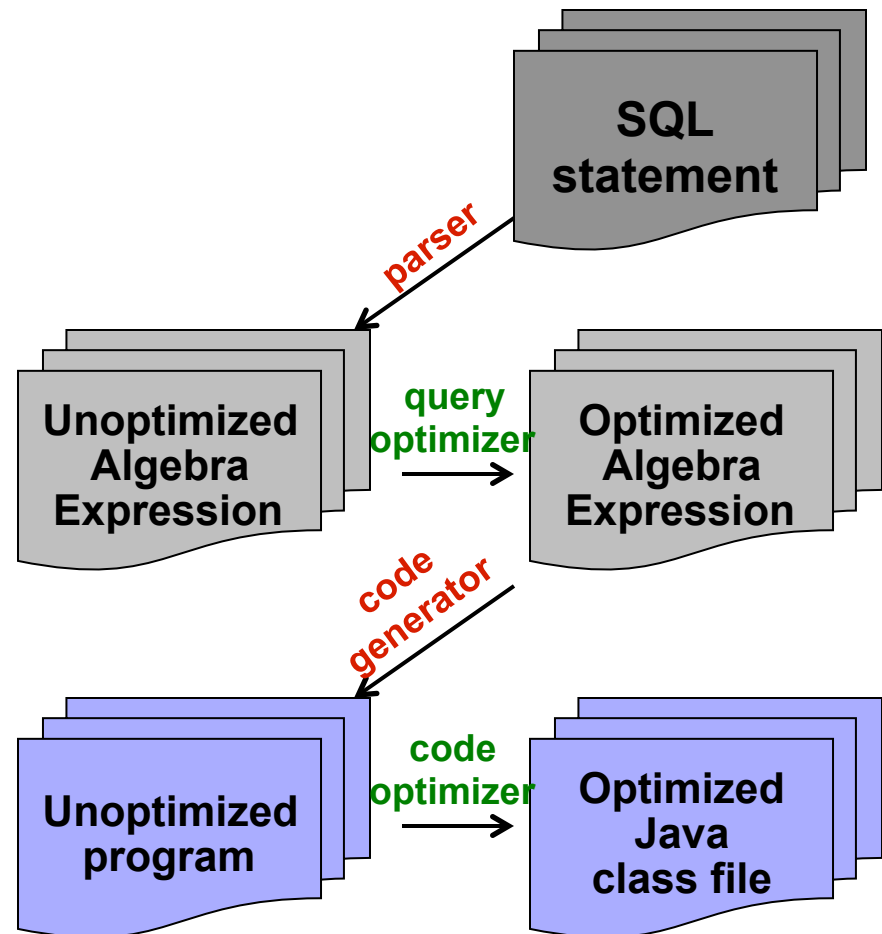


DRACO Perspective of Programming



DRACO View of Query Optimization

- **Generative Programming** occurs when mapping between levels of abstraction
- **Automatic Programming** occurs when optimizing within a level of abstraction is done automatically





MBSE Requires Advances in:

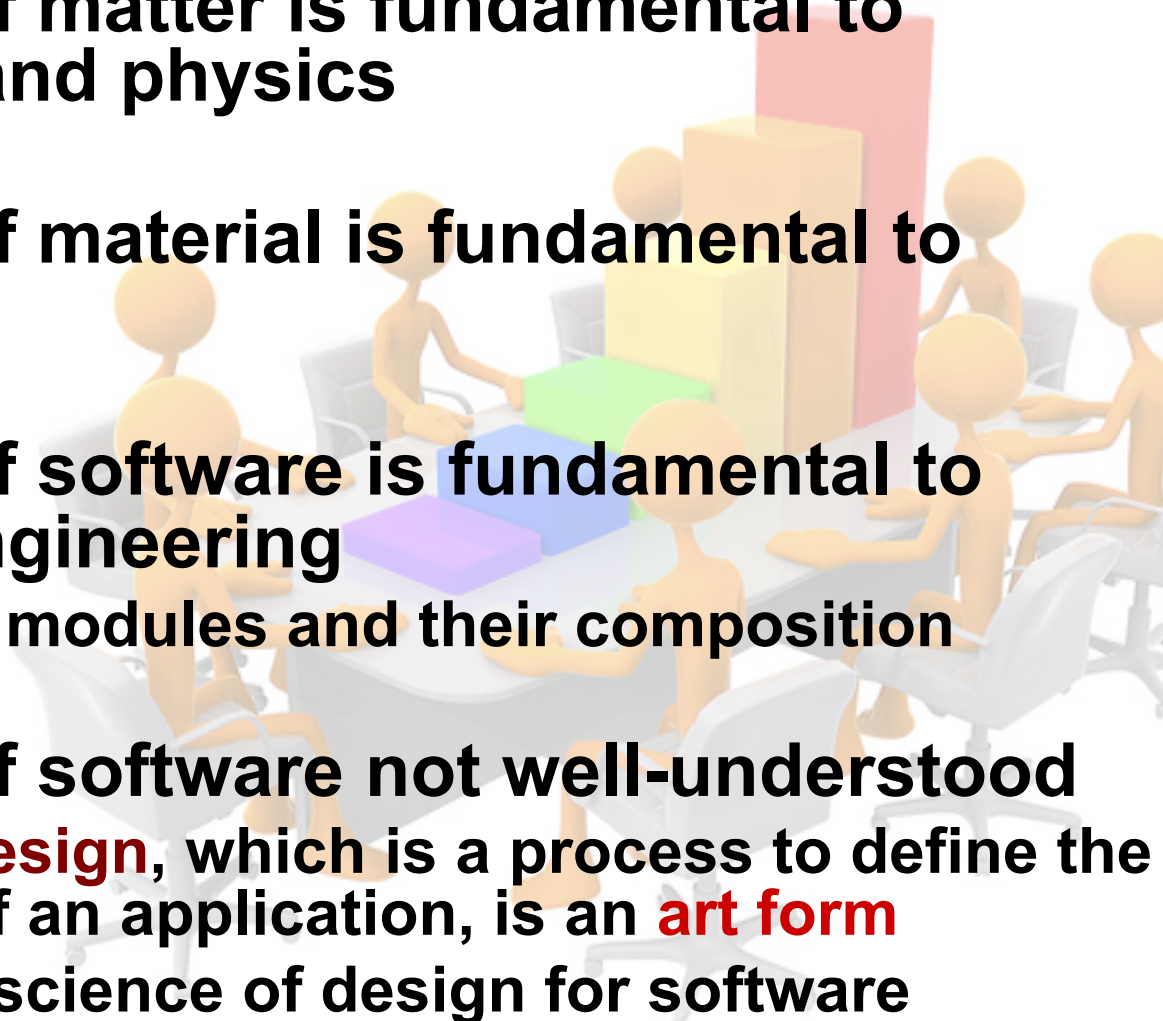
- **Generative Programming (GP)**
 - Understand domain well enough to generate software
 - Programs that synthesize other programs
 - Metaprogramming, skeleton communities

- **Domain-Specific Languages (DSLs)**
 - Elevate program specifications to compact domain-specific notations that are easier to write & maintain

- **Automatic Programming**
 - Culmination of GP and DSLs
 - Generate efficient programs from declarative specs
 - Precursor to Model-Based Software Engineering



A Case for Automated Programming

- Structure of matter is fundamental to chemistry and physics
 - Structure of material is fundamental to engineers
 - Structure of software is fundamental to software engineering
 - structure = modules and their composition
 - Structure of software not well-understood
 - **Software design**, which is a process to define the structure of an application, is an **art form**
 - We need a science of design for software
- 

So Did the Database People Get it?

CODD, E. F.

A relational model of data for large shared data banks. *Comm. ACM* 13, 6 (June 1970), 377-387.

In this article, the author discusses the advantages of using a relational rather than a network model for representing data in a data management system. It is his contention that such a model would provide greater independence of the data from application programs and would provide a means of making consistency checks on facts in the database.

The author describes a convention under which most data structures can be represented as sets of relations. He then describes how some of the common set operations correspond to database operations.

While the relational model is not a new one, this paper probably contains its most rigorous and elegant statement and, as such, represents a valuable contribution to the theory of data processing.

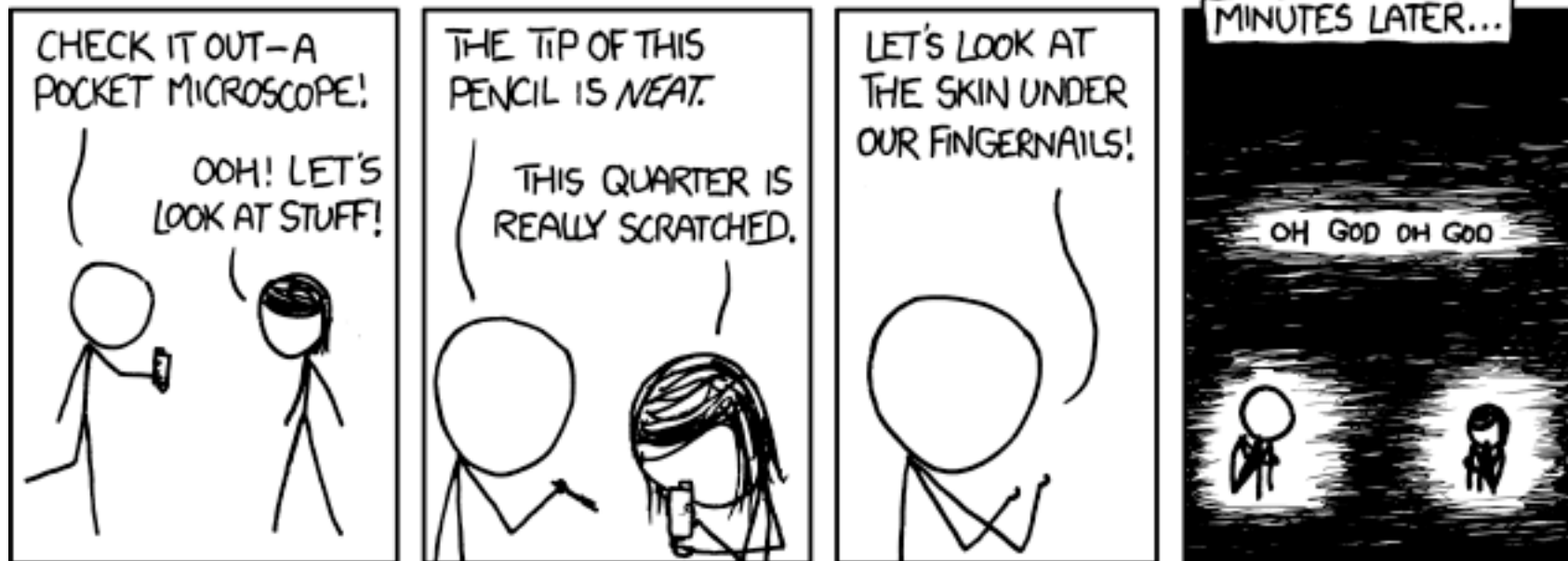
However, it would appear to this reviewer that the model is of more theoretical than practical interest.

While the relational model does indeed eliminate some data dependencies (e.g., ordering dependence) it does so by eliminating the practical advantages of ordering. While it does provide a mechanism for automatically making some data checks, these checks would often be so computationally painful that they would be of little practical value.

R. W. Elliott, College Station, Texas
#20,780 March 1971

- Nope...Look at the Computing Surveys Review of Codd's 1970 seminal paper on the Relational Model...
- Do you get it?
- This is a race and my leg is running now...
 - Your leg is coming up!

Looking at the details





Key Papers

■ Automatic Programming

Robert Balzer. **A 15-year Perspective on Automatic Programming.** *IEEE Transactions on Software Engineering*, 11(11):1257–1268, November 1985.

David R. Barstow. **Domain-Specific Automatic Programming.** *IEEE Transactions on Software Engineering*, 11(11):1321–1336, November 1985.

Charles Rich and Richard C. Waters. **Automatic Programming: Myths and Prospects.** *IEEE Computer*, 21(8):40–51, August 1988.

■ Automated Planning

Keith Golden. **A Domain Description Language for Data Processing.** Proc. of the International Conf. on Automated Planning and Scheduling, 2003.

M. Stickel et al. **Deductive Composition of Astronomical Software from Subroutine Libraries.** Proc. of the International Conf. on Automated Deduction, 1994.



Homework and Milestone Reminders

- **Milestone 3: Light-Weight Transformation Environment (see Milestone 3 assignment)**
 - Due by 11:55pm, Thursday, May 5th, 2011.

- **Milestone 4: Final MBSE Environment (see Milestone 4 assignment)**
 - Due by 11:55pm, Friday, May 13th, 2011.