

Low-Code: How Low Can You Go?

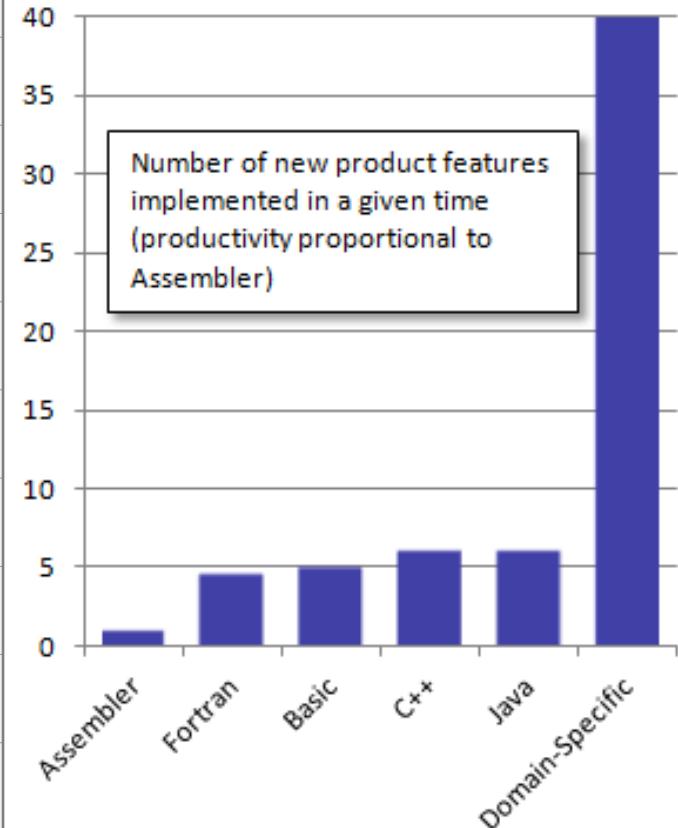
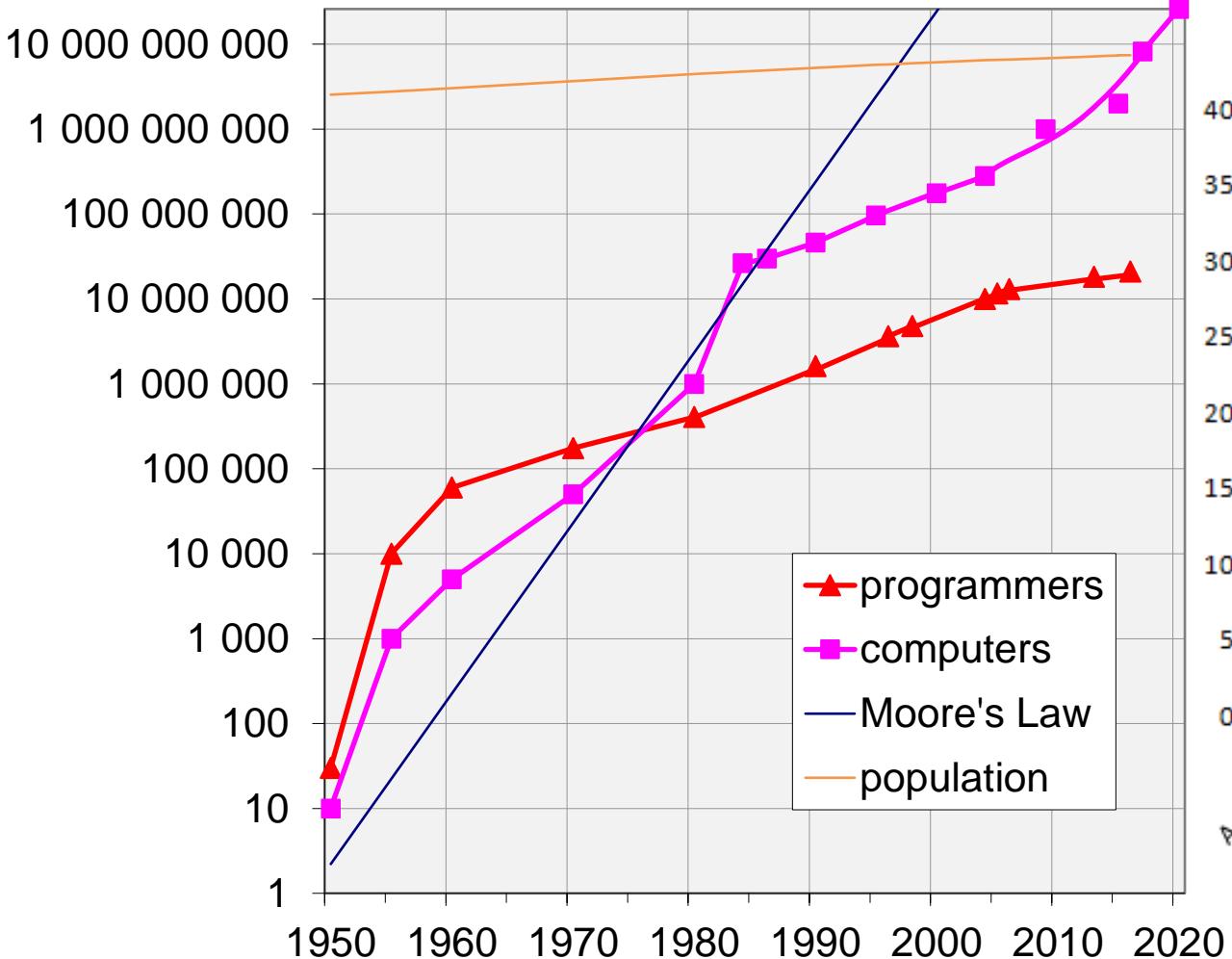
Steven Kelly, MetaCase
Low-Code 2023, MODELS,
3 October 2023

Low-Code Platforms: Huge & growing

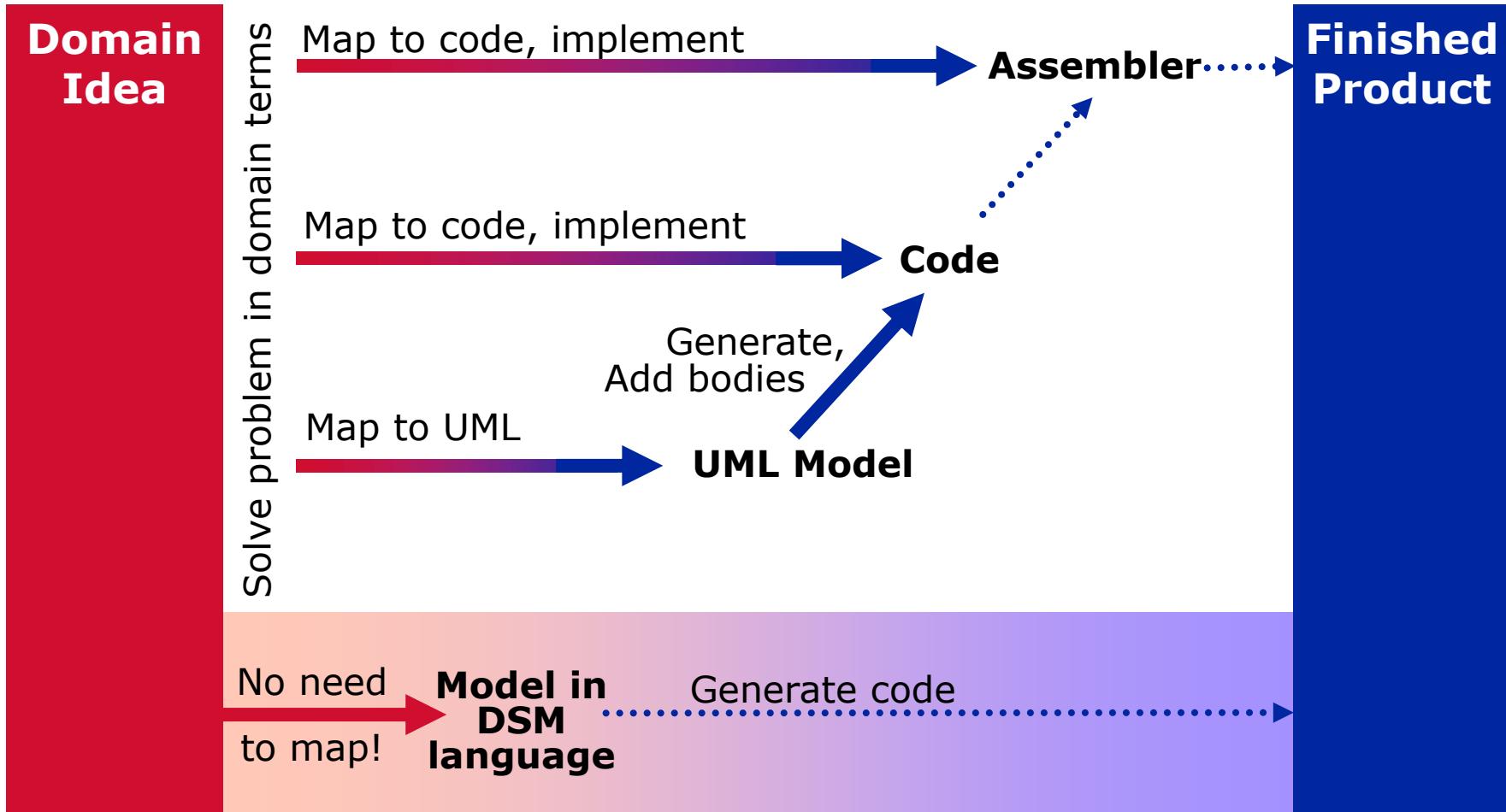
	2021	2022	2023	2024
Low-Code Application Platforms	6 324	7 968	9 960	12 351
Citizen Automat.+Dev Platforms	554	732	953	1 232
Total	6 878	8 700	10 913	13 583
All low code	18 497	22 462	26 869	31 949

Low-Code Technologies Revenue (millions of USD), Gartner, December 2022

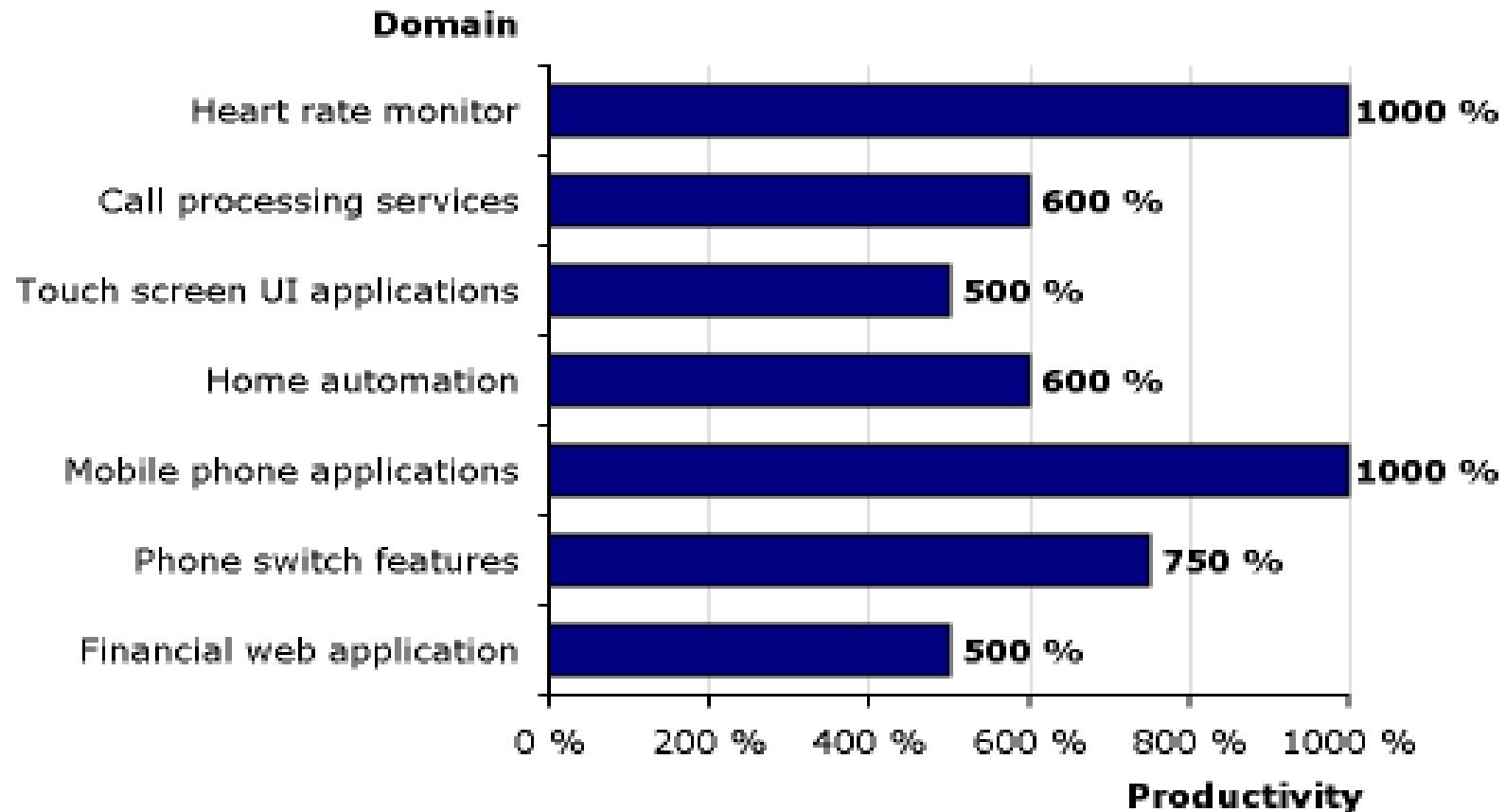
- Exponential growth: the single clearest marker of success
- ...and nearly synonymous with our whole industry history



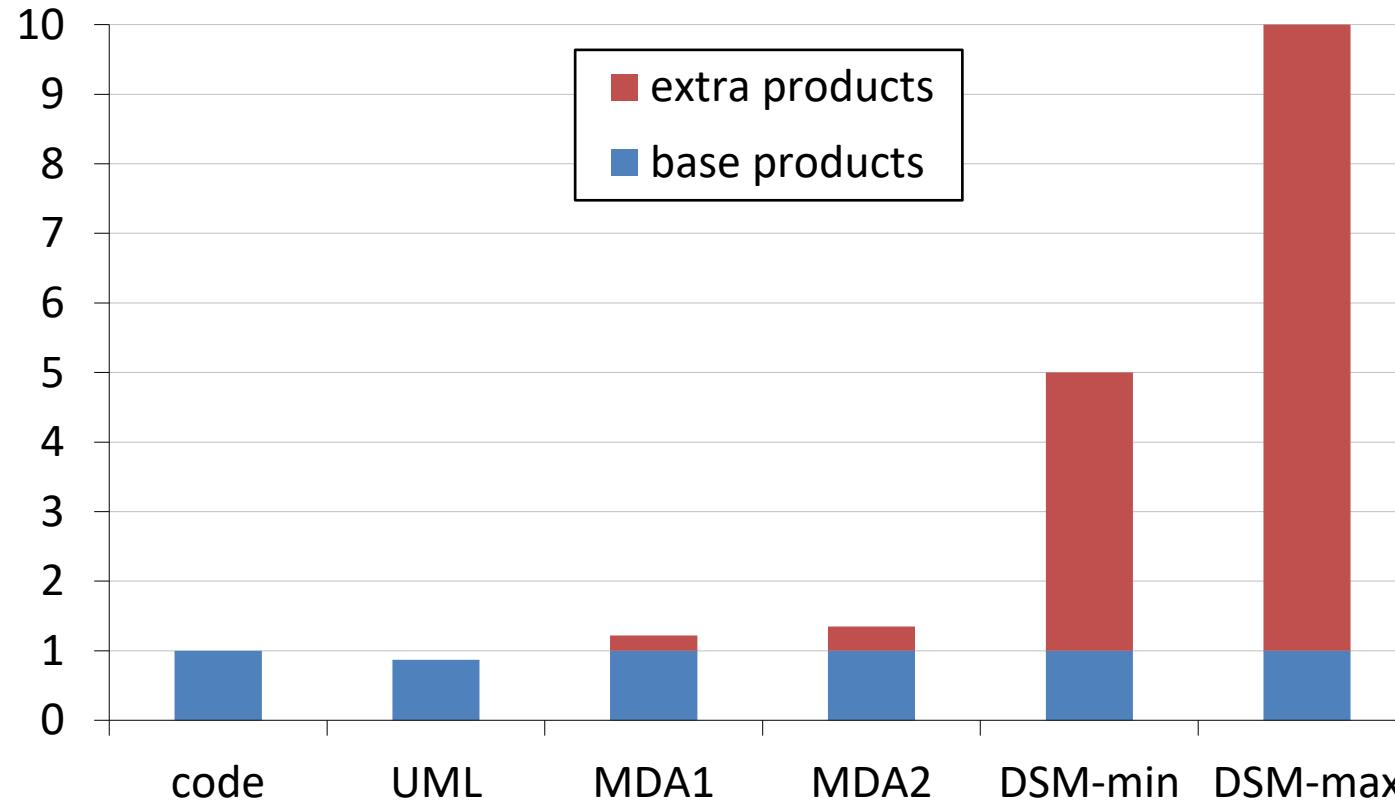
Modelling functionality vs. modelling code

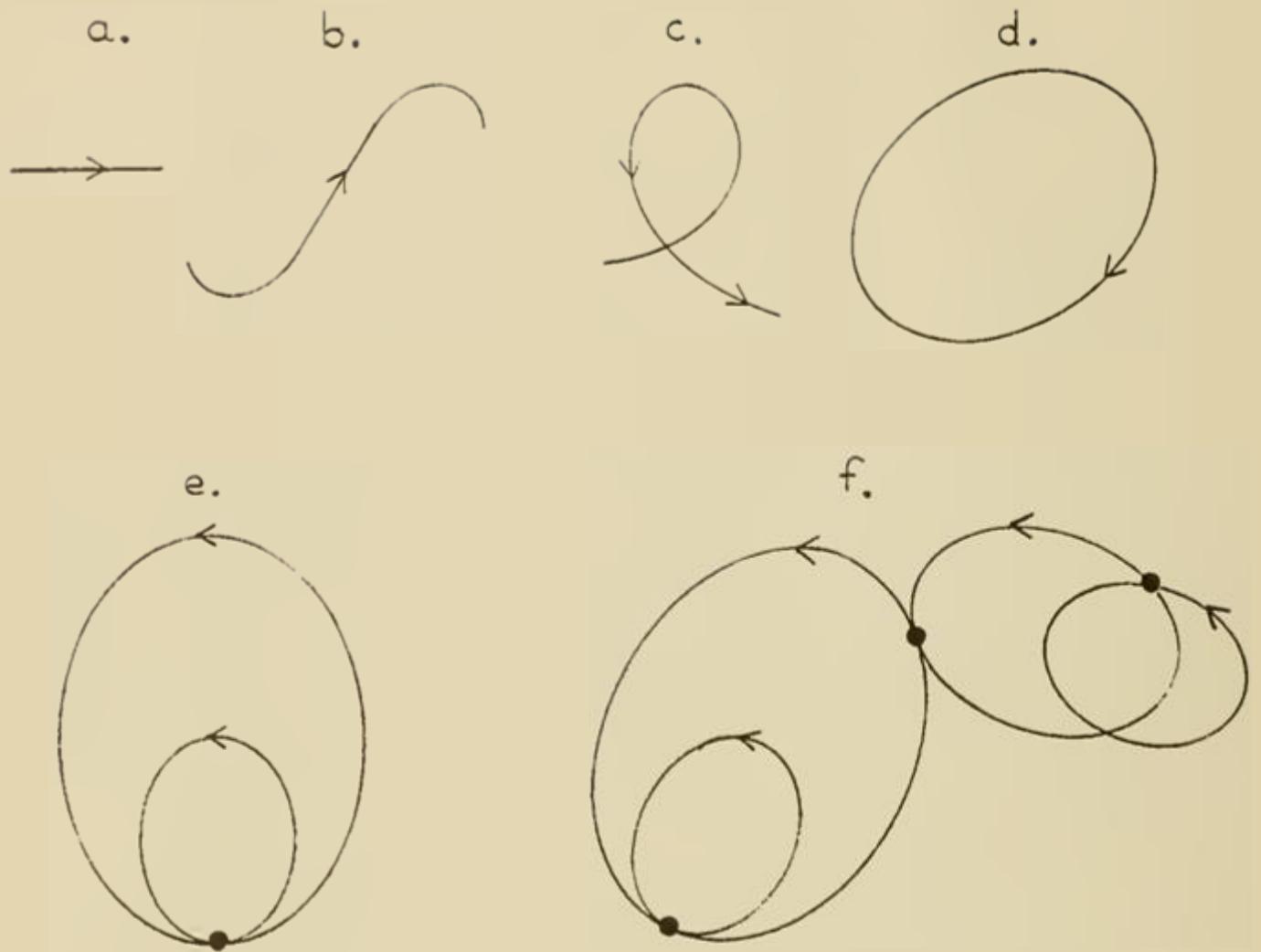


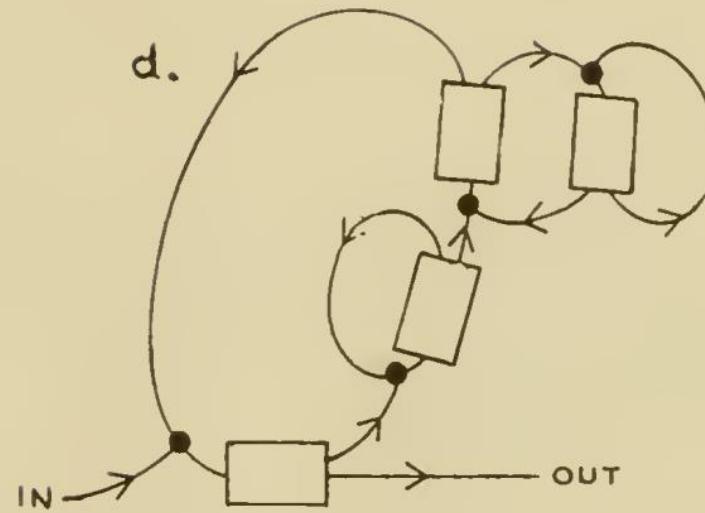
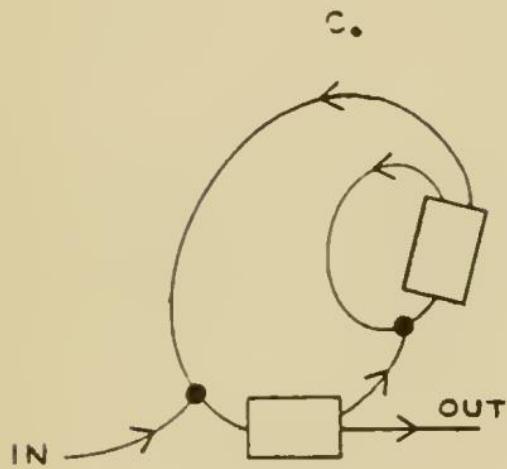
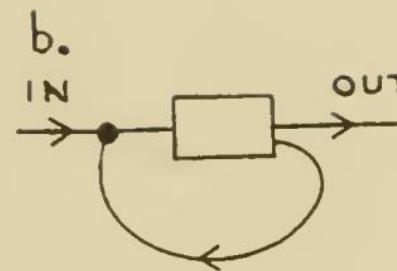
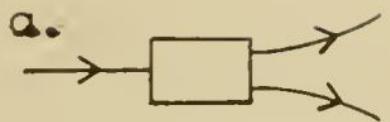
Domain-Specific Modeling: 5-10x faster than coding or UML

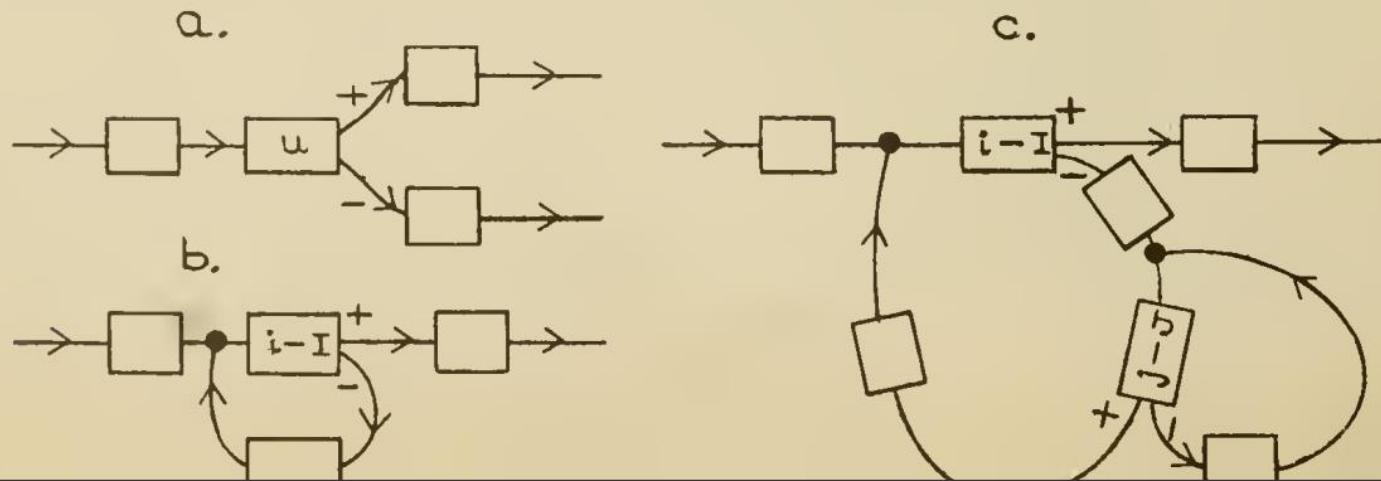
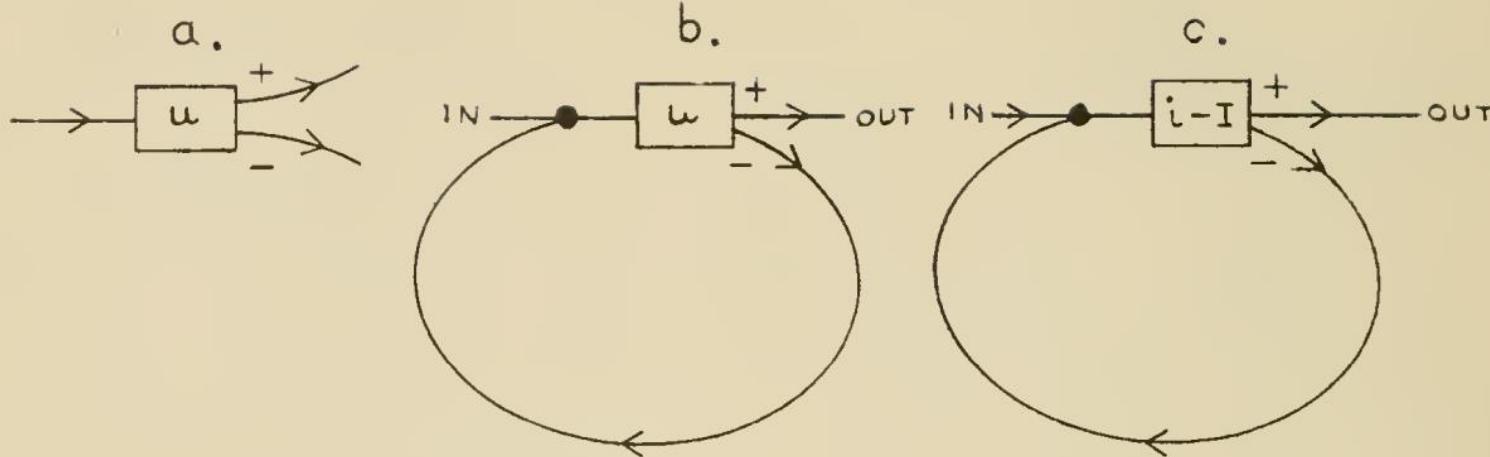


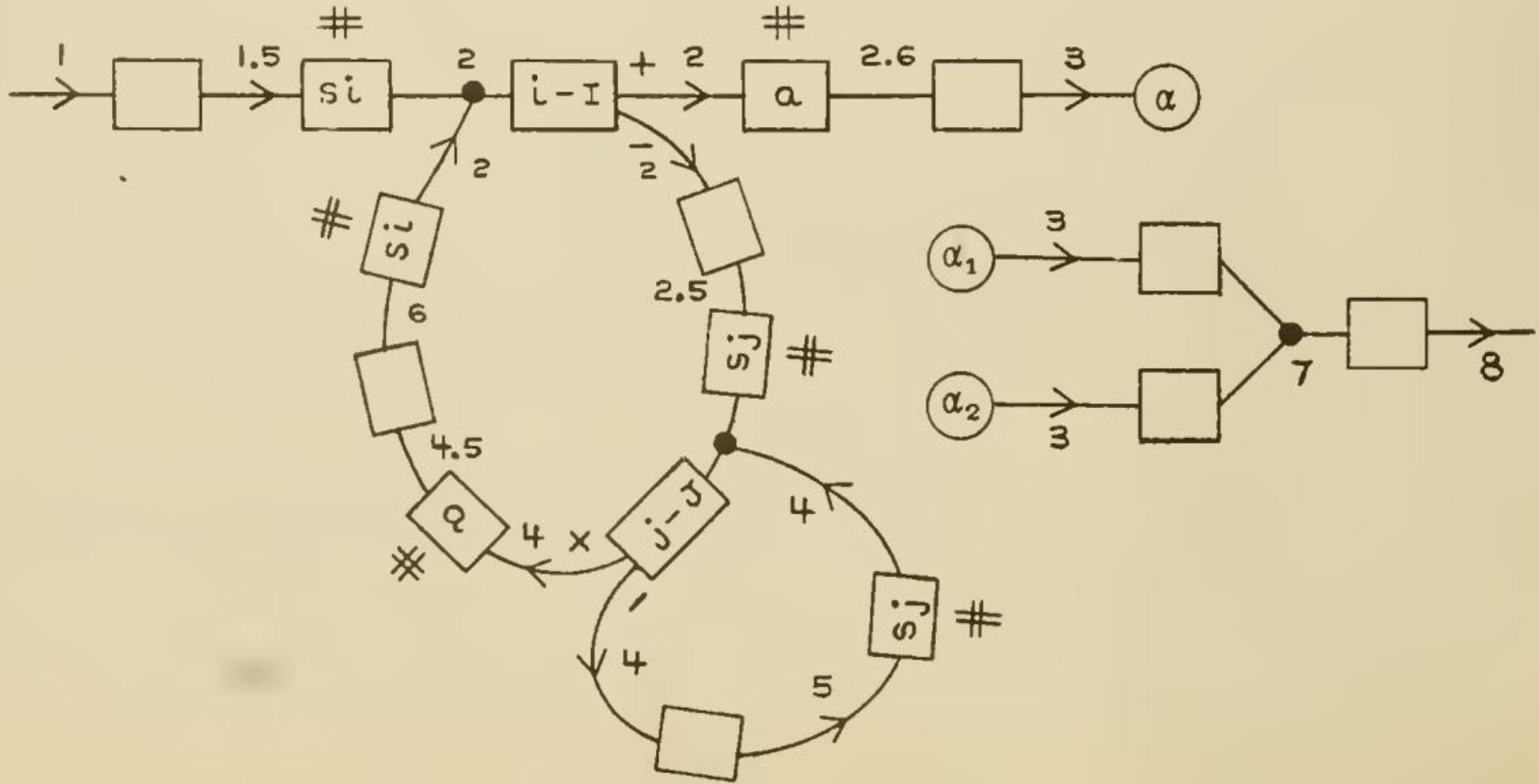
Productivity of development compared to hand-coding











		1	1.5	2	2.5	2.6	3
I,J	A. I	-				I	-
i,j	2	-		g(i)		g(I)	p
B. I	-						-
2	-				$\frac{1}{2}$		-

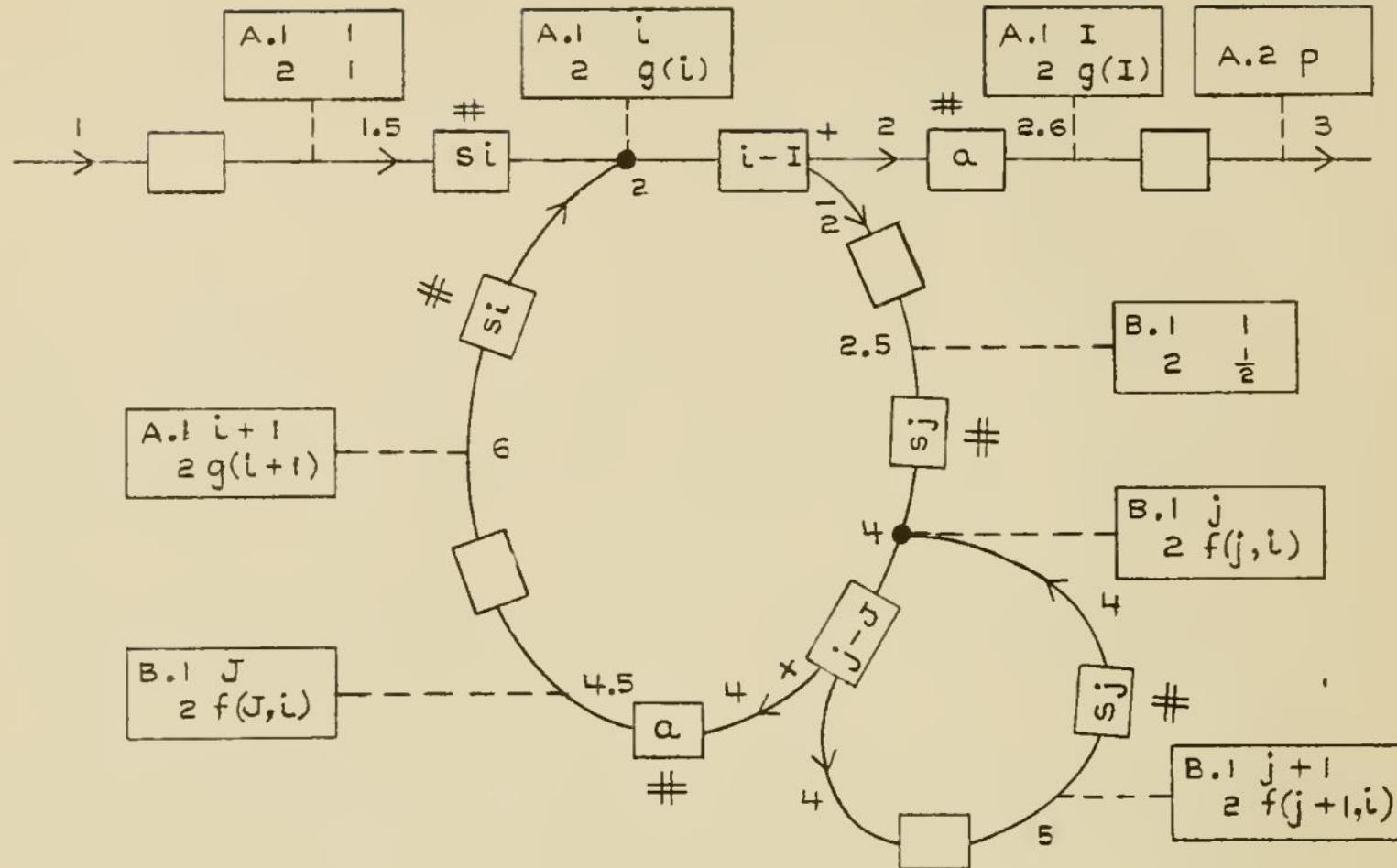
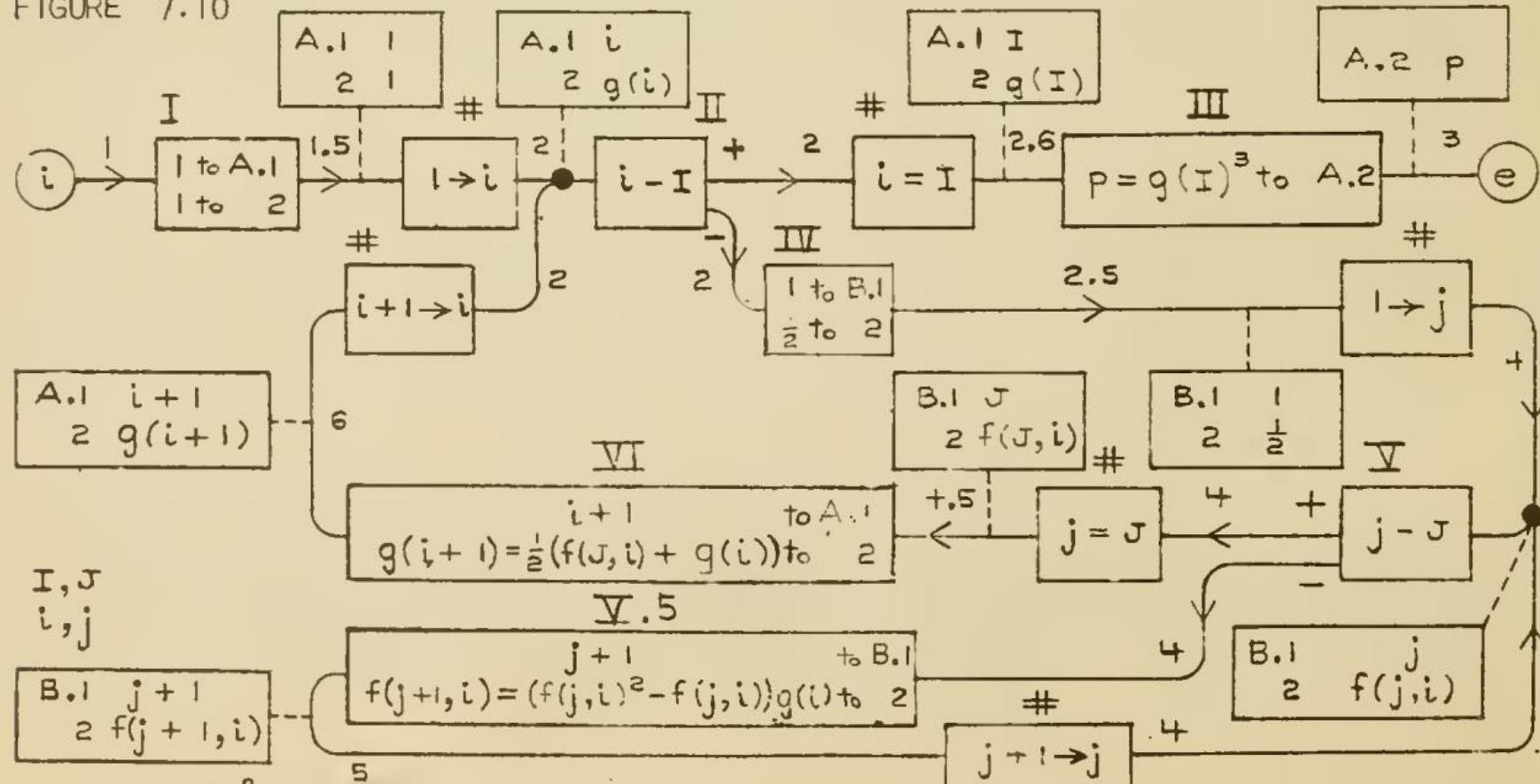


FIGURE 7.10



$$p = g(I)^3,$$

$$g(I) = 1, \quad g(i+1) = \frac{1}{2} (f(j,i) + g(i)),$$

$$f(I,i) = \frac{1}{2}, \quad f(j+1,i) = (f(j,i))^2 - f(j,i) g(i).$$

Code like John von Neumann!

- 1. Understand the problem**, assumptions and simplifications
- 2. Draw the flow diagram**

"It has been our invariable experience, that once the problem has been understood and prepared as in 1., the drawing of the flow diagram presents little difficulty. Every moderately mathematically trained person should be able to do this in a routine manner [by reading this report or similar training]."

- 3. Write the code for each box**

"We feel certain that a moderate amount of experience with this stage of coding suffices to remove from it all difficulties, and to make it a perfectly routine operation.

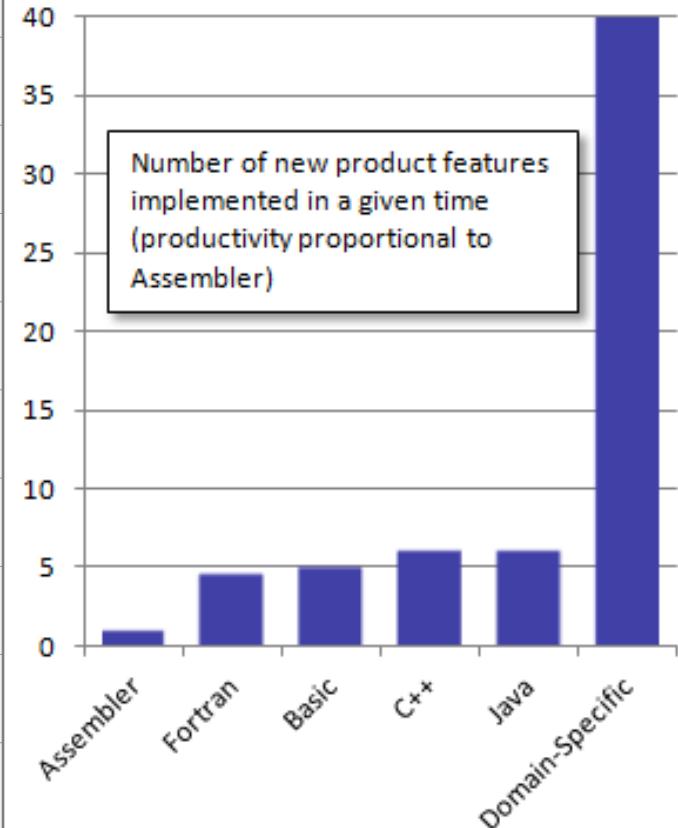
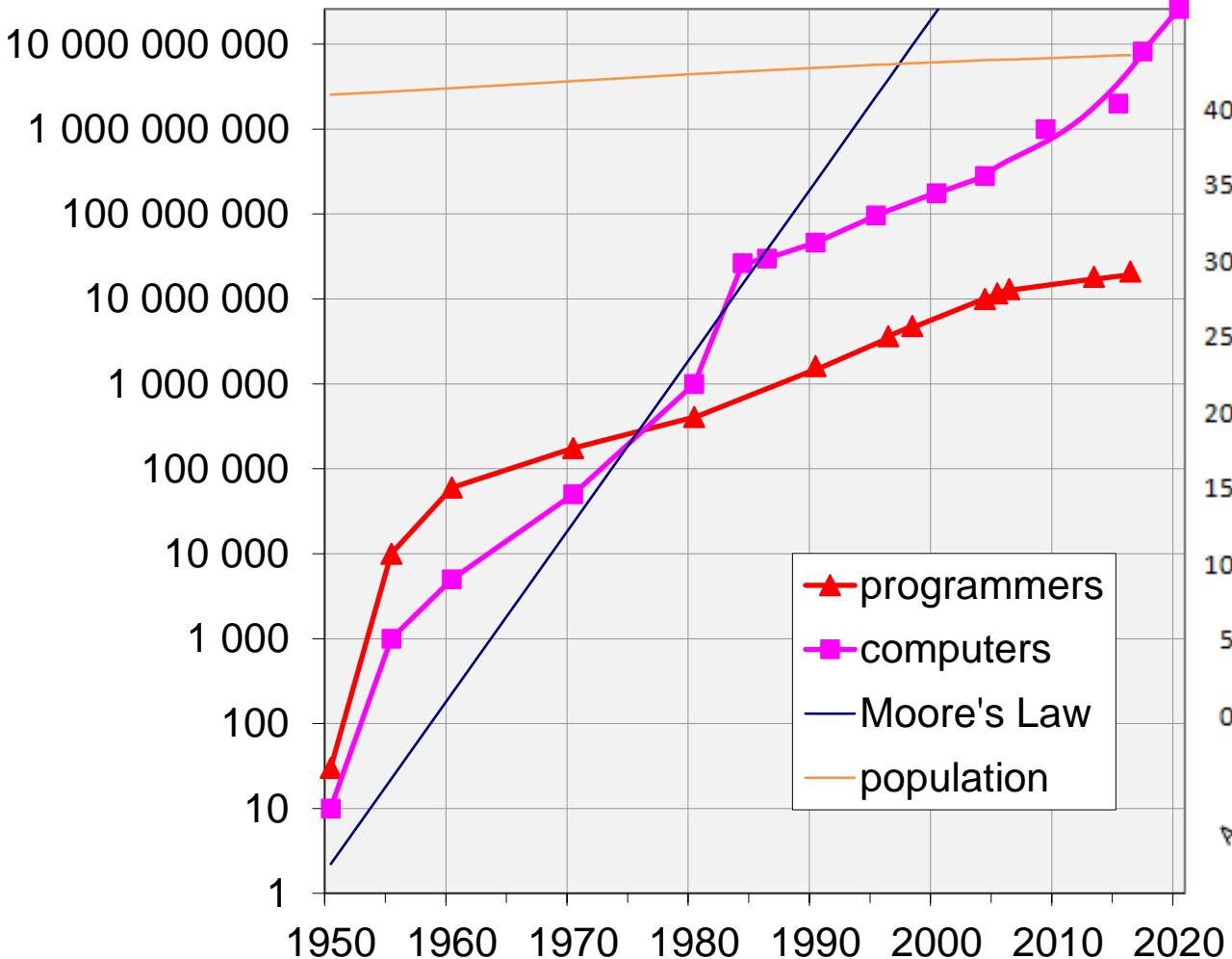
- 4. Number the variables and routines**

History of electronic stored program **computers**

- 1940s Colossus, ABC, Harvard Mark 1, Z3, ...
- 1948 January SSEC IBM, electromechanical
- 1948 May 12 ARC 2 U.London, relays
- 1948 April 12 ENIAC U.Pennsylvania, getting close...
- 1948 June 21 Baby U.Manchester
- 1949 April Mark 1 U.Manchester
- 1949 May 6 EDSAC U.Cambridge
- 1951 Mark 1 Ferranti, Manchester: commercially available

What does it mean that von Neumann could figure all this out in 1946-7, even before the first computer existed?!

Development time
increases to infinity as the
task approaches the limit
of the programmer's skill

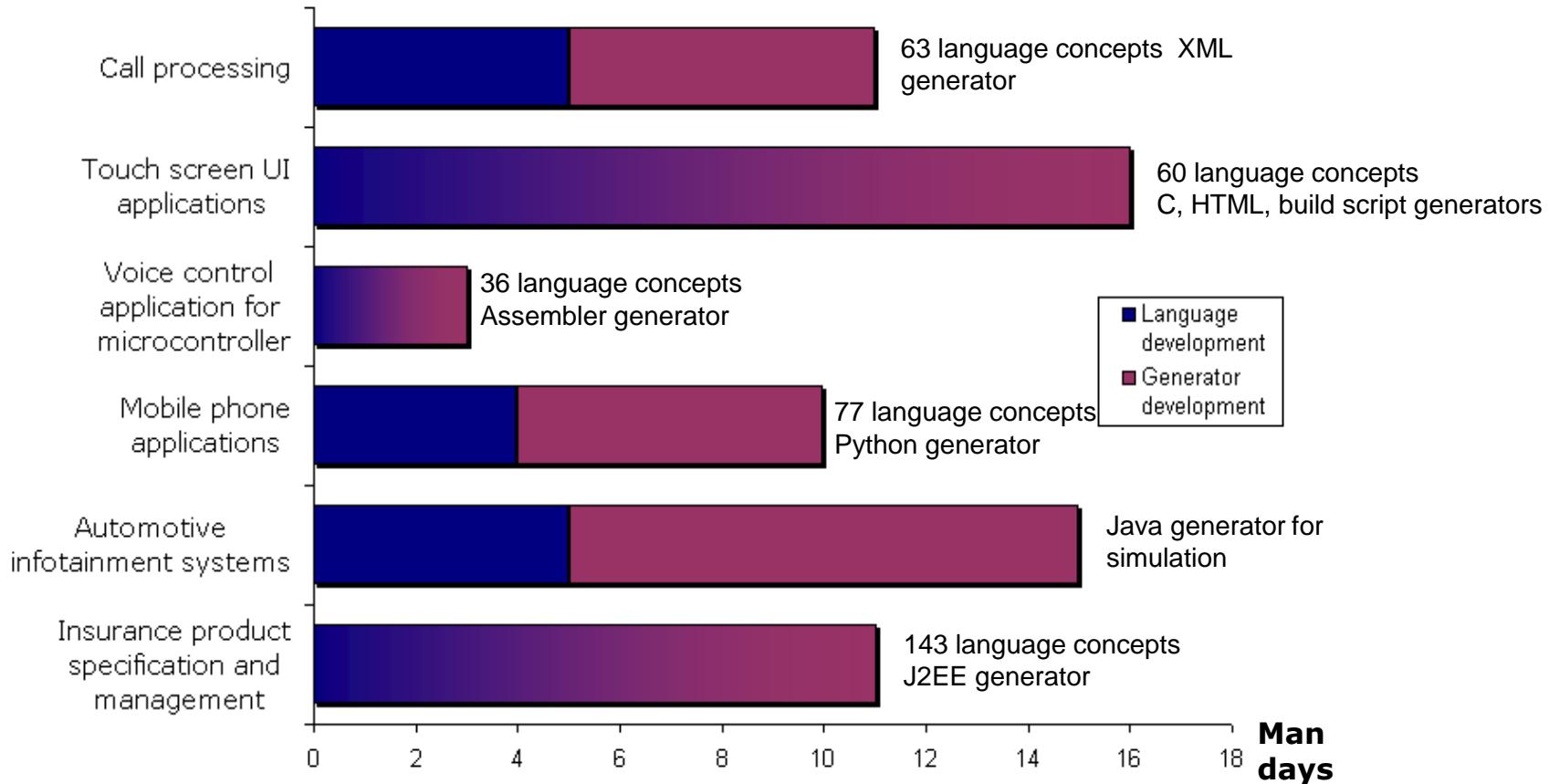


N generations of languages + tools, N/2 generations of meta lang. tools

Fixed, ready-made	date	date	Build your own, meta
Machine code			
Assembler	1948	1960	compiler-compiler
Compiler	1955		compiler-compiler
CASE tool (graphical)	1986	1986	language workbench
Low-code tool	2011		language workbench

**But Low-Code tools today only exist
for a very small subset of domains.
So how to make more? And fast enough?**

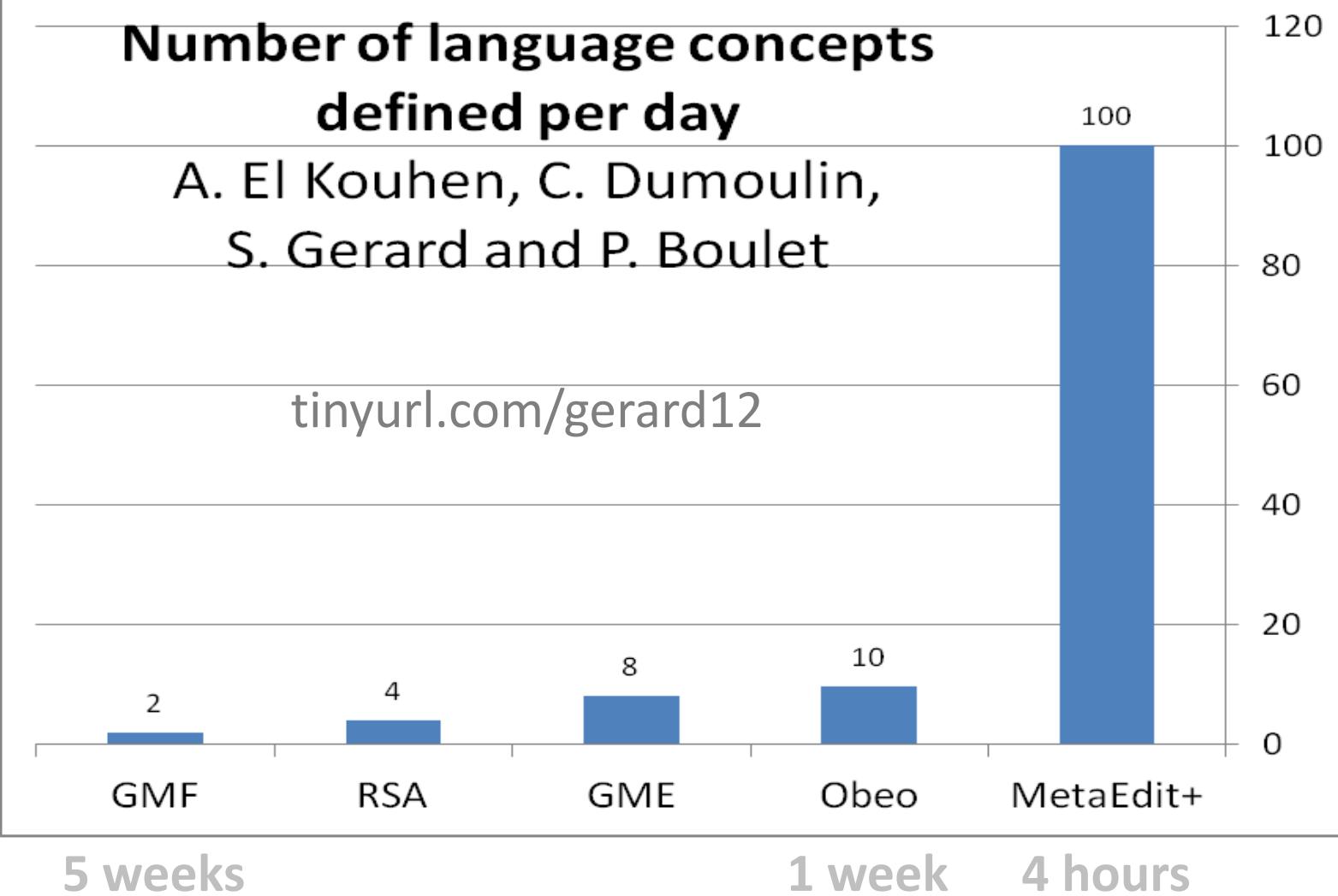
DSM Solution Development Time



Number of language concepts defined per day

A. El Kouhen, C. Dumoulin,
S. Gerard and P. Boulet

tinyurl.com/gerard12



1843 Ada Lovelace, Note G, on Babbage's Analytical Engine

Number of Operation	Nature of Operation	acted upon.	receiving results.	the value on any Variable.	Statement of Results.	Data							
						V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8
1	\times	$^1V_2 \times ^1V_3$	$^1V_4, ^1V_5, ^1V_6$	$\begin{cases} ^1V_2 = ^1V_2 \\ ^1V_3 = ^1V_3 \end{cases}$	$= 2n$...	2	n	$2n$	$2n$	$2n$
2	$-$	$^1V_4 - ^1V_1$	1V_4	$\begin{cases} ^1V_4 = ^2V_4 \\ ^1V_1 = ^1V_1 \end{cases}$	$= 2n-1$	1	$2n-1$
3	$+$	$^1V_5 + ^1V_1$	2V_5	$\begin{cases} ^1V_5 = ^2V_5 \\ ^1V_1 = ^1V_1 \end{cases}$	$= 2n+1$	1	$2n+1$
4	\div	$^2V_4 \div ^2V_5$	$^1V_{11}$	$\begin{cases} ^3V_5 = ^0V_5 \\ ^3V_4 = ^0V_4 \end{cases}$	$= \frac{2n-1}{2n+1}$	0	0
5	\div	$^1V_{11} \div ^1V_2$	$^2V_{11}$	$\begin{cases} ^1V_{11} = ^2V_{11} \\ ^1V_2 = ^1V_2 \end{cases}$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$...	2
6	$-$	$^0V_{13} - ^2V_{11}$	$^1V_{13}$	$\begin{cases} ^2V_{11} = ^0V_{11} \\ ^0V_{13} = ^1V_{13} \end{cases}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1}$
7	$-$	$^1V_3 - ^1V_1$	$^1V_{10}$	$\begin{cases} ^1V_2 = ^1V_3 \\ ^1V_3 = ^2V_2 \end{cases}$	$= n-1 (= 3)$	1	...	n
8	$+$	$^1V_2 + ^0V_7$	1V_7	$\begin{cases} ^1V_2 = ^1V_2 \\ ^0V_7 = ^1V_7 \end{cases}$	$= 2+U=2$...	2	2	...
9	\div	$^1V_6 \div ^1V_7$	$^3V_{11}$	$\begin{cases} ^1V_6 = ^1V_6 \\ ^0V_{11} = ^3V_{11} \end{cases}$	$= \frac{2n}{2} = A_1$	$2n$	2	...
10	\times	$^1V_{12} \times ^3V_{11}$	$^1V_{12}$	$\begin{cases} ^1V_{21} = ^1V_{21} \\ ^3V_{11} = ^3V_{11} \end{cases}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$
11	$+$	$^1V_{12} + ^1V_{13}$	$^2V_{12}$	$\begin{cases} ^1V_{12} = ^0V_{12} \\ ^1V_{13} = ^2V_{13} \end{cases}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$
12	$-$	$^1V_{10} - ^1V_1$	$^2V_{10}$	$\begin{cases} ^1V_{10} = ^2V_{10} \\ ^1V_1 = ^1V_1 \end{cases}$	$= n-2=2$	1
13	$-$	$^1V_6 - ^1V_1$	2V_6	$\begin{cases} ^1V_6 = ^2V_6 \\ ^1V_1 = ^1V_1 \end{cases}$	$= 2n-1$	1	$2n-1$

Alt Text

How would you describe this object and its context to someone who is blind or low vision?

- The subject(s) in detail
- The setting
- The actions or interactions
- Other relevant information

(1-2 detailed sentences recommended)

A screenshot of a computer

Description automatically generated



Thank you!

Le langage GRAPHTALK : illustration

Plusieurs types de méta-objets et plusieurs types de méta-liens

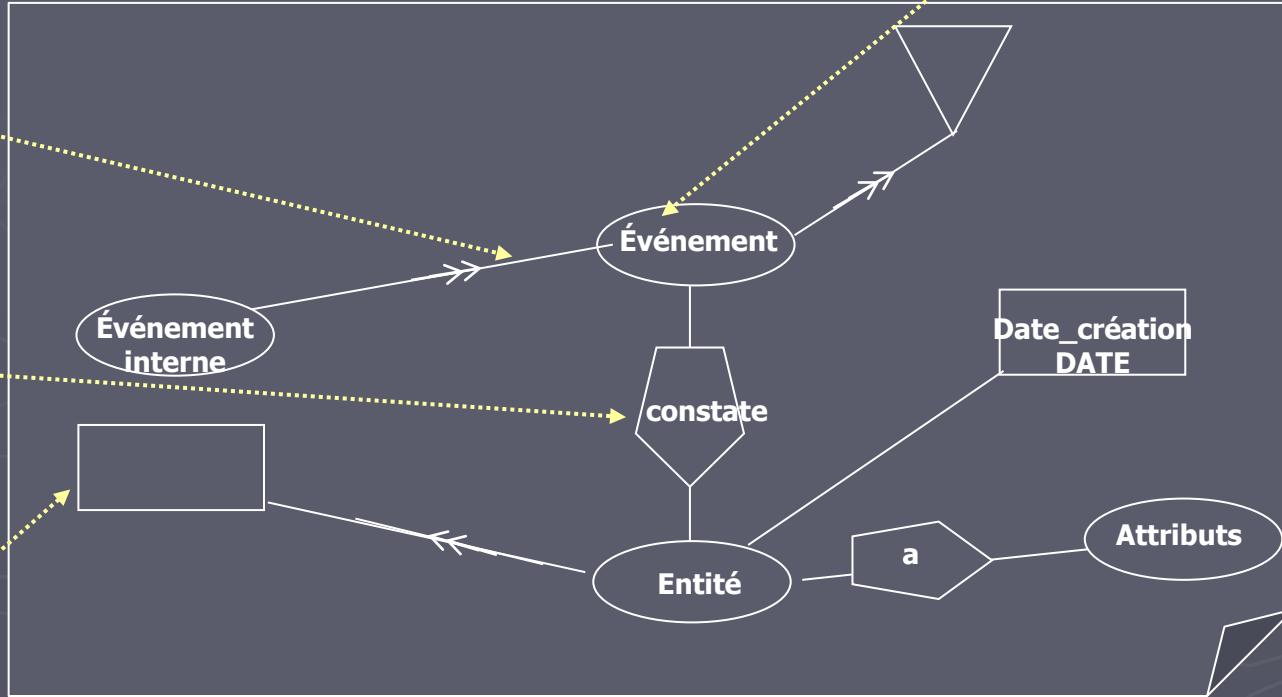
Un méta-objet de type **objet**

Un méta-lien de type **is-A**

Un méta-objet de type **lien**

Un méta-objet de type **forme**

Un méta-modèle
Sous forme de **graphe**



GRAPHTALK language: example

Several types of meta-objects and several types of meta-relationships

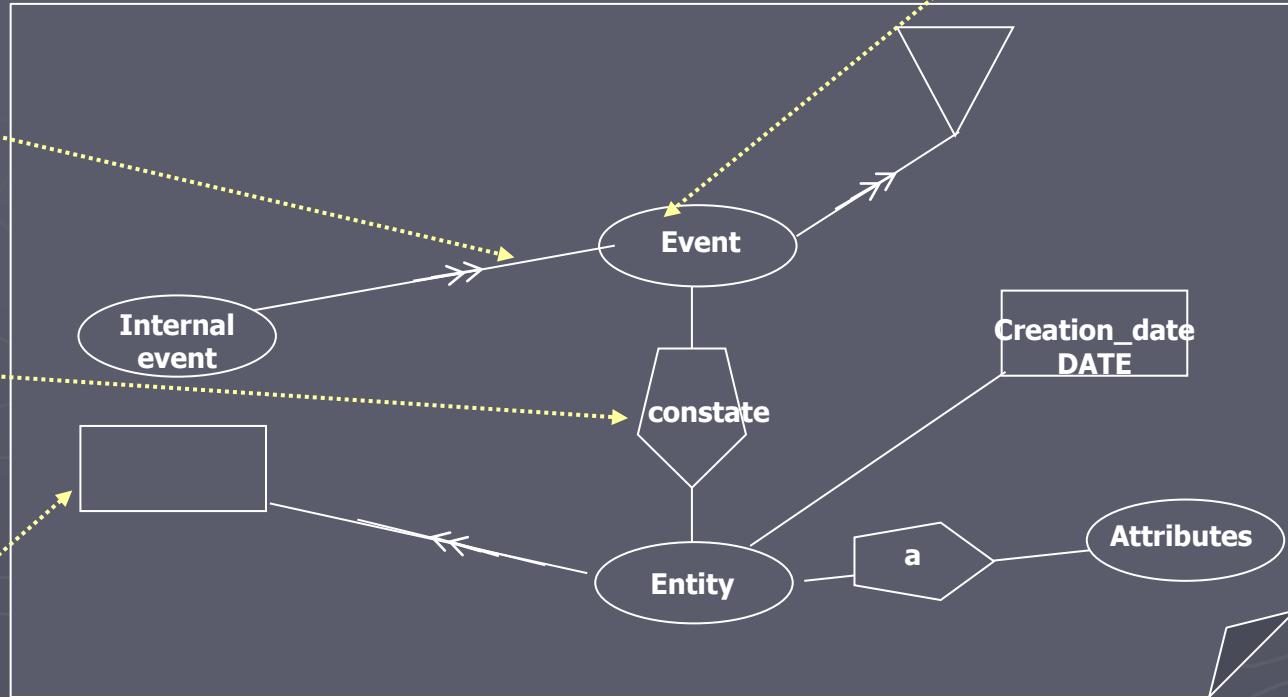
A meta-relationship
of type **is-A**

A meta-object
of type
relationship

A meta-object
of type **symbol**

A metamodel in the
form of a **graph**

A meta-object
of type **object**



Referenced DSM Cases

(for more see www.dsmforum.org)

Heart rate monitor	Polar	Kärnä et al., Evaluating the use of DSM in embedded UI application development , Procs of DSM'09 at OOPSLA, 2009.
Call processing services		Kelly, S., Tolvanen, J.-P., Chapter 5, Domain-Specific Modeling: Enabling Full Code Generation , Wiley, 2008. [CPL project in MetaEdit+]
Touch screen UI applications	Panasonic	Safa, L., The Making Of User-Interface Designer: A Proprietary DSM Tool , Procs of DSM'07 at OOPSLA, 2007.
Home automation		Kelly, S., Tolvanen, J.-P., Chapter 5, Domain-Specific Modeling: Enabling Full Code Generation , Wiley, 2008. [Home automation project in MetaEdit+]
Mobile phone applications	Nokia	MetaCase, Nokia case study , 2000
Phone switch features		Weiss, D. M., Lai, C. T. R., Software Product-line Engineering: A Family-Based Software Development Process , Addison Wesley Longman, 1999.
Financial web application	Pecunet	Kelly, S., Tolvanen, J.-P., Chapter 6, Domain-Specific Modeling: Enabling Full Code Generation , Wiley, 2008. MetaCase, Pecunet case study , 2001. IASA Architect Skills Library: Domain-Specific Modeling , 2007. [Insurance project in MetaEdit+]