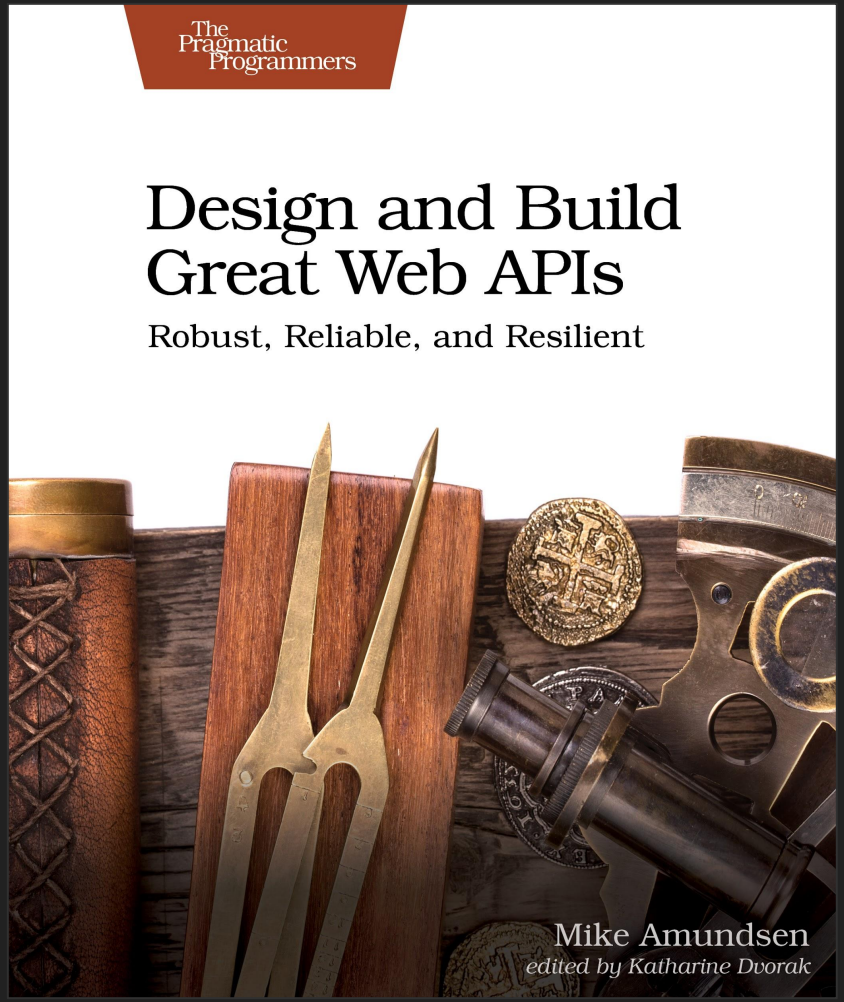
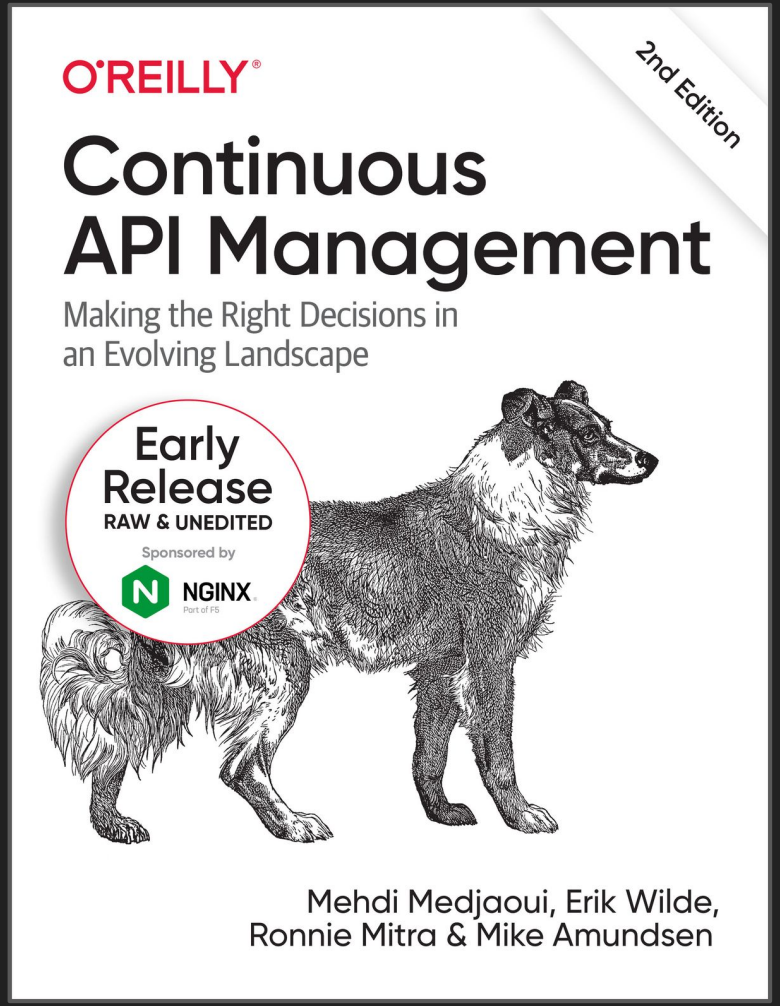

APIs of the Future:
Are You Ready?
Mike Amundsen
@mamund
http://linkedin.com/mamund
youtube.com/mamund


Mike Amundsen
@mamund


The Pragmatic Programmers
Design and Build Great Web APIs
Robust, Reliable, and Resilient
Mike Amundsen
edited by Katharine Dvorak


O'REILLY
2nd Edition
Continuous API Management
Making the Right Decisions in an Evolving Landscape
Early Release
RAW & UNEDITED
Sponsored by
NGINX
Part of F5
Mehdi Medjaoui, Erik Wilde,
Ronnie Mitra & Mike Amundsen

APIs of the Future: Overview

- The First Age of Computing : Where we were
- The Second Age of Computing : Where we are
- The Third Age of Computing: Where we're going
- And So...


Where We Were
1850-1950

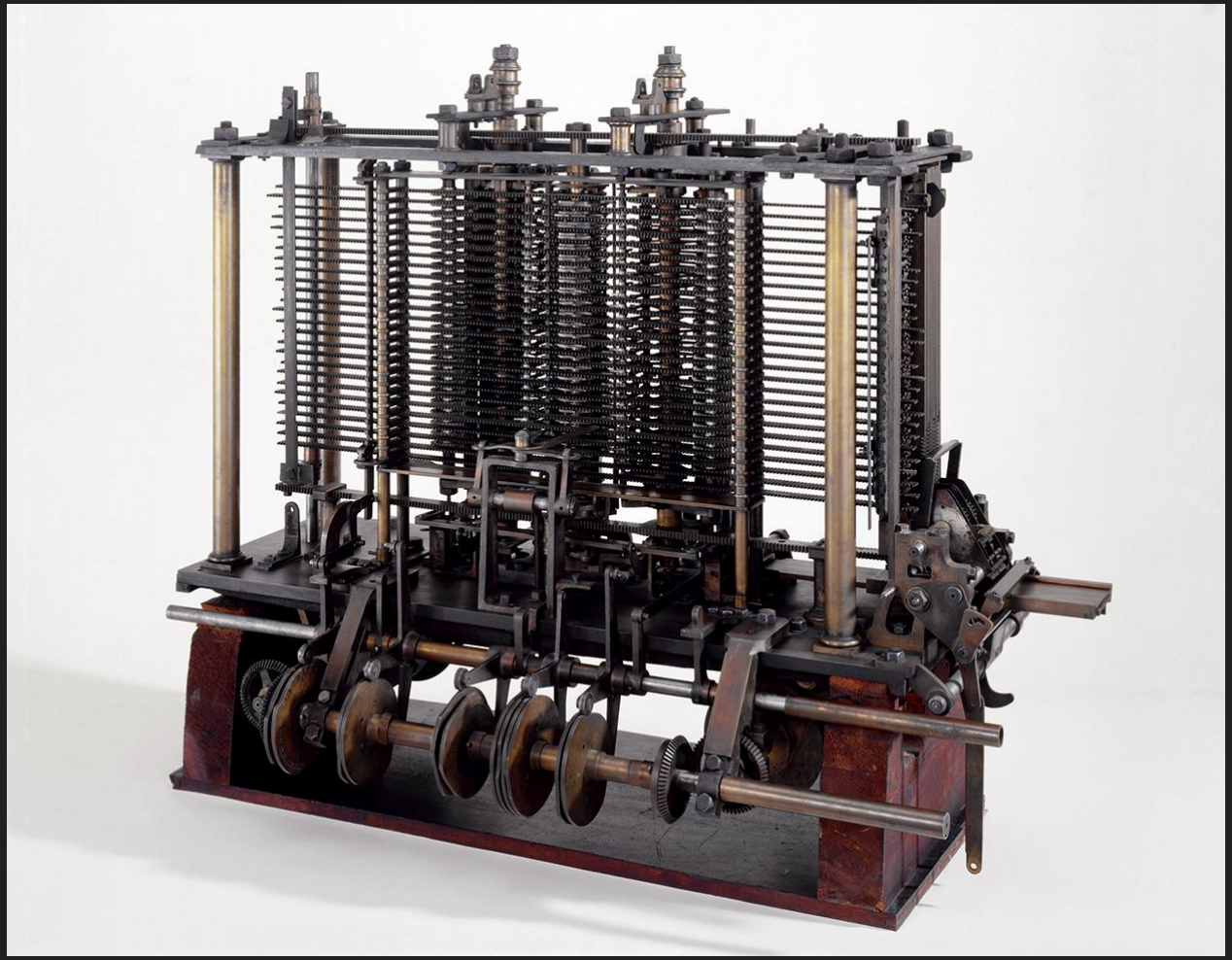


Diagram for the computation by the Engine of the Numbers of Bernoulli. See Note G. (page 722 *et seq.*)

Number of Operation.	Nature of Operation.	Variables acted upon.	Variables receiving results.	Indication of change in the value on any Variable.	Statement of Results.	Data. ${}^{1}V_1$ 1	Data. ${}^{1}V_2$ 2	Data. ${}^{1}V_3$ n	Working Variables. ${}^{0}V_4$	${}^{0}V_5$	${}^{0}V_6$	${}^{0}V_7$	${}^{0}V_8$	${}^{0}V_9$	${}^{0}V_{10}$	${}^{0}V_{11}$	${}^{0}V_{12}$	${}^{0}V_{13}$	Result Variables. ${}^{1}V_{21}$ B_1	${}^{1}V_{22}$ B_2	${}^{1}V_{23}$ B_3	${}^{0}V_{24}$... B_7
1	×	${}^{1}V_2 \times {}^{1}V_3$	${}^{1}V_4, {}^{1}V_5, {}^{1}V_6$	${}^{1}V_2 = {}^{1}V_2$, ${}^{1}V_3 = {}^{1}V_3$	$= 2n$	...	2	n	$2n$	$2n$	$2n$											
2	−	${}^{1}V_4 - {}^{1}V_1$	${}^{2}V_4$	${}^{1}V_4 = {}^{2}V_4$, ${}^{1}V_1 = {}^{1}V_1$	$= 2n-1$	1	...	...	$2n-1$													
3	+	${}^{1}V_5 + {}^{1}V_1$	${}^{2}V_5$	${}^{1}V_5 = {}^{2}V_5$, ${}^{1}V_1 = {}^{1}V_1$	$= 2n+1$	1	...	...	...	$2n+1$												
4	÷	${}^{2}V_5 \div {}^{2}V_4$	${}^{1}V_{11}$	${}^{2}V_5 = {}^{0}V_5$, ${}^{2}V_4 = {}^{0}V_4$	$= \frac{2n-1}{2n+1}$	...	...	...	0	0	...	...	...	...	...	$\frac{2n-1}{2n+1}$						
5	÷	${}^{1}V_{11} \div {}^{1}V_2$	${}^{2}V_{11}$	${}^{1}V_{11} = {}^{2}V_{11}$, ${}^{1}V_2 = {}^{1}V_2$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$	...	2	...	...	...	...	...	...	...	...	$\frac{1}{2} \cdot \frac{2n-1}{2n+1}$						
6	−	${}^{0}V_{13} - {}^{2}V_{11}$	${}^{1}V_{13}$	${}^{2}V_{11} = {}^{0}V_{11}$, ${}^{0}V_{13} = {}^{1}V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$	...	...	...	...	...	...	...	...	...	...	0	.....	$-\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$				
7	−	${}^{1}V_3 - {}^{1}V_1$	${}^{1}V_{10}$	${}^{1}V_3 = {}^{1}V_3$, ${}^{1}V_1 = {}^{1}V_1$	$= n-1\ (=3)$	1	...	n	...	...	...	...	...	...	$n-1$							
8	+	${}^{1}V_2 + {}^{0}V_7$	${}^{1}V_7$	${}^{1}V_2 = {}^{1}V_2$, ${}^{0}V_7 = {}^{1}V_7$	$= 2 + 0 = 2$	...	2	...	...	...	...	2										
9	÷	${}^{1}V_6 \div {}^{1}V_7$	${}^{3}V_{11}$	${}^{1}V_6 = {}^{1}V_6$, ${}^{0}V_{11} = {}^{3}V_{11}$	$= \frac{2n}{2} = A_1$	...	...	...	...	...	$2n$	2	...	...	...	$\frac{2n}{2} = A_1$						
10	×	${}^{1}V_{21} \times {}^{3}V_{11}$	${}^{1}V_{12}$	${}^{1}V_{21} = {}^{1}V_{21}$, ${}^{3}V_{11} = {}^{3}V_{11}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$	...	...	...	...	...	...	...	...	...	...	$\frac{2n}{2} = A_1$	$B_1 \cdot \frac{2n}{2} = B_1 A_1$	.....	B_1			
11	+	${}^{1}V_{12} + {}^{1}V_{13}$	${}^{2}V_{13}$	${}^{1}V_{12} = {}^{0}V_{12}$, ${}^{1}V_{13} = {}^{2}V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$ ...	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{-\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}\right\}$				
12	−	${}^{1}V_{10} - {}^{1}V_1$	${}^{2}V_{10}$	${}^{1}V_{10} = {}^{2}V_{10}$, ${}^{1}V_1 = {}^{1}V_1$	$= n-2\ (=2)$	1	...	...	...	...	...	...	...	...	$n-2$							
13	−	${}^{1}V_6 - {}^{1}V_1$	${}^{2}V_6$	${}^{1}V_6 = {}^{2}V_6$, ${}^{1}V_1 = {}^{1}V_1$	$= 2n-1$	1	...	...	...	...	$2n-1$											
14	+	${}^{1}V_1 + {}^{1}V_7$	${}^{2}V_7$	${}^{1}V_1 = {}^{1}V_1$, ${}^{1}V_7 = {}^{2}V_7$	$= 2 + 1 = 3$	1	...	...	...	...	...	3										
15	÷	${}^{2}V_6 \div {}^{2}V_7$	${}^{1}V_8$	${}^{2}V_6 = {}^{2}V_6$, ${}^{2}V_7 = {}^{2}V_7$	$= \frac{2n-1}{3}$	...	...	...	...	...	$2n-1$	3	$\frac{2n-1}{3}$									
16	×	${}^{1}V_8 \times {}^{3}V_{11}$	${}^{4}V_{11}$	${}^{1}V_8 = {}^{0}V_8$, ${}^{3}V_{11} = {}^{4}V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3}$	...	...	...	...	...	...	...	0	...	...	$\frac{2n}{2} \cdot \frac{2n-1}{3}$						
17	−	${}^{2}V_6 - {}^{1}V_1$	${}^{3}V_6$	${}^{2}V_6 = {}^{3}V_6$, ${}^{1}V_1 = {}^{1}V_1$	$= 2n-2$	1	...	...	...	...	$2n-2$											
18	+	${}^{1}V_1 + {}^{2}V_7$	${}^{3}V_7$	${}^{2}V_7 = {}^{3}V_7$, ${}^{1}V_1 = {}^{1}V_1$	$= 3 + 1 = 4$	1	...	...	...	...	...	4										
19	÷	${}^{3}V_6 \div {}^{3}V_7$	${}^{1}V_9$	${}^{3}V_6 = {}^{3}V_6$, ${}^{3}V_7 = {}^{3}V_7$	$= \frac{2n-2}{4}$	...	...	...	...	...	$2n-2$	4	...	$\frac{2n-2}{4}$	...	$\left\{\frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = A_3\right\}$						
20	×	${}^{1}V_9 \times {}^{4}V_{11}$	${}^{5}V_{11}$	${}^{1}V_9 = {}^{0}V_9$, ${}^{4}V_{11} = {}^{5}V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = A_3$	...	...	...	...	...	...	...	...	0								
21	×	${}^{1}V_{22} \times {}^{5}V_{11}$	${}^{0}V_{12}$	${}^{1}V_{22} = {}^{1}V_{22}$, ${}^{0}V_{12} = {}^{2}V_{12}$	$= B_3 \cdot \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = B_3 A_3$	...	...	...	...	...	...	...	...	...	...	0	$B_3 A_3$	.....	.....	B_3		
22	+	${}^{2}V_{12} + {}^{2}V_{13}$	${}^{3}V_{13}$	${}^{2}V_{12} = {}^{0}V_{12}$, ${}^{2}V_{13} = {}^{3}V_{13}$	$= A_0 + B_1 A_1 + B_3 A_3$	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{A_3 + B_1 A_1 + B_2 A_3\right\}$				
23	−	${}^{2}V_{10} - {}^{1}V_1$	${}^{3}V_{10}$	${}^{2}V_{10} = {}^{3}V_{10}$, ${}^{1}V_1 = {}^{1}V_1$	$= n-3\ (=1)$	1	...	...	...	...	...	...	...	...	$n-3$							

Here follows a repetition of Operations thirteen to twenty-three.

24	+	${}^{4}V_{13} + {}^{0}V_{24}$	${}^{1}V_{24}$	${}^{4}V_{13} = {}^{0}V_{13}$, ${}^{0}V_{24} = {}^{1}V_{24}$	$= B_7$	...	...	...	...	...	...	...	...	...	...	.....	.....	.....	.....	.....	.....	B_7
25	+	${}^{1}V_1 + {}^{1}V_3$	${}^{1}V_3$	${}^{1}V_1 = {}^{1}V_1$, ${}^{1}V_3 = {}^{1}V_3$, ${}^{5}V_6 = {}^{0}V_6$, ${}^{5}V_7 = {}^{0}V_7$	$= n+1 = 4+1 = 5$ ..... by a Variable-card. by a Variable-card.	1	...	$n+1$	...	...	0	0										

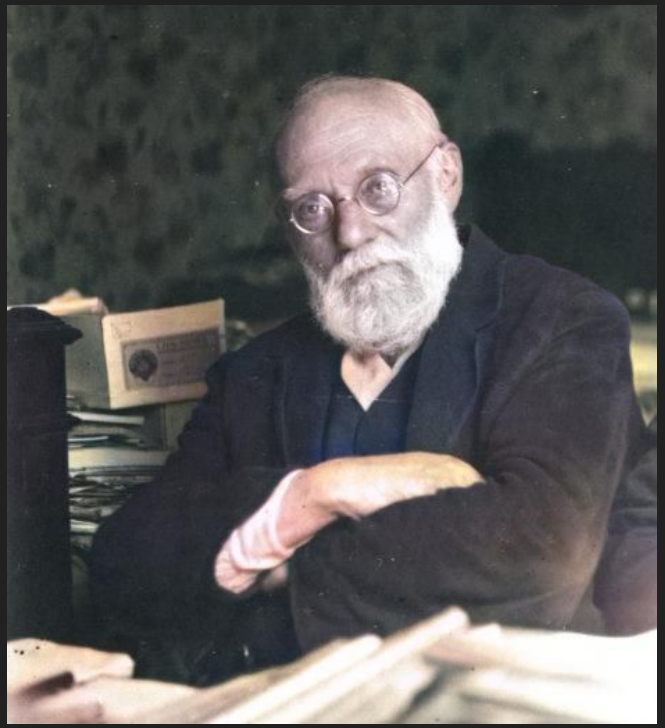
"We may say most aptly that the Analytical Engine weaves algebraic patterns just as the Jacquard-loom weaves flowers and leaves."

Ada Lovelace (1815 - 1852)

"[Debugging] is damnably troublesome work, and plagues me."

Ada Lovelace (1815 - 1852)

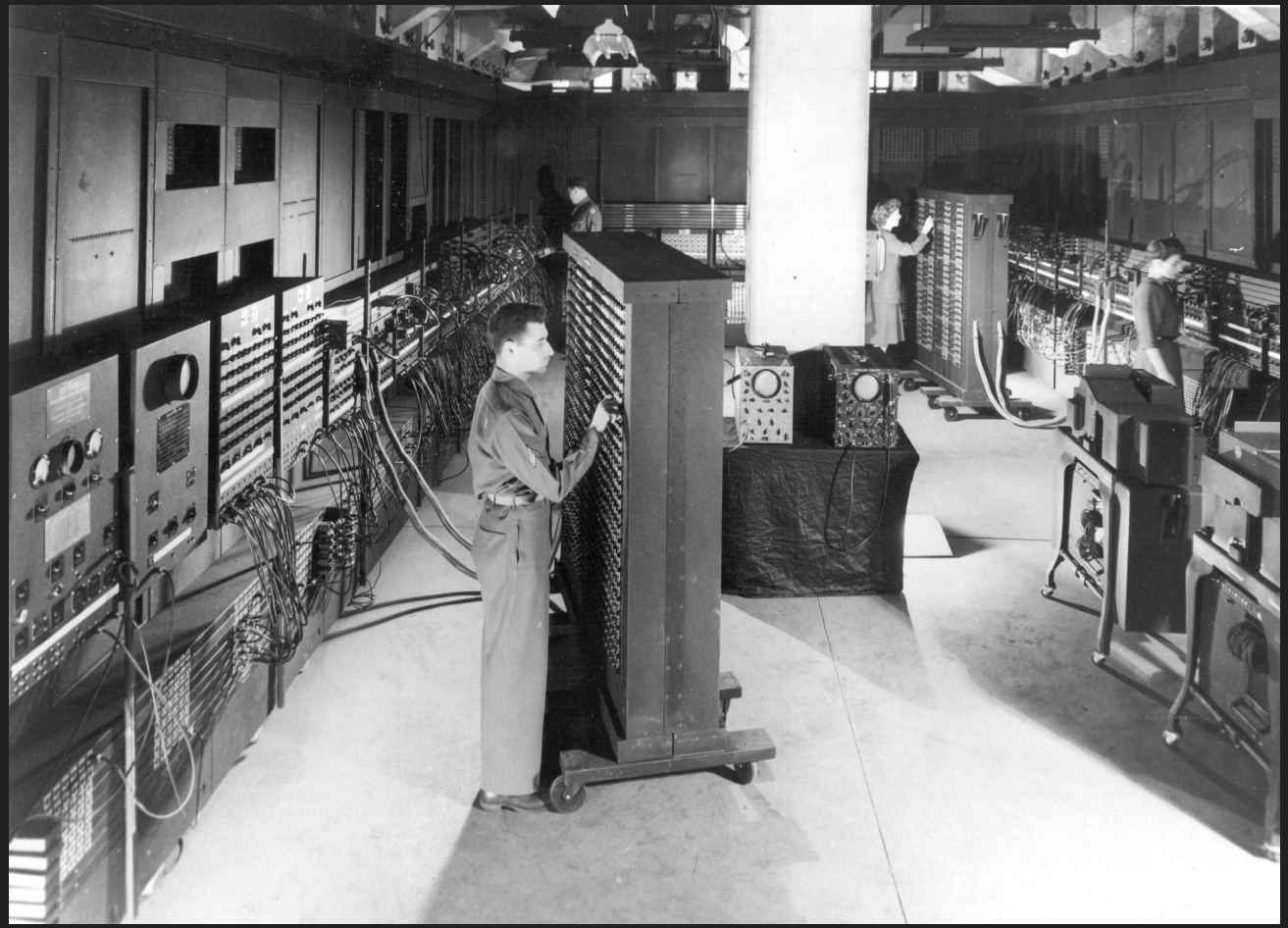
A moving image of the world



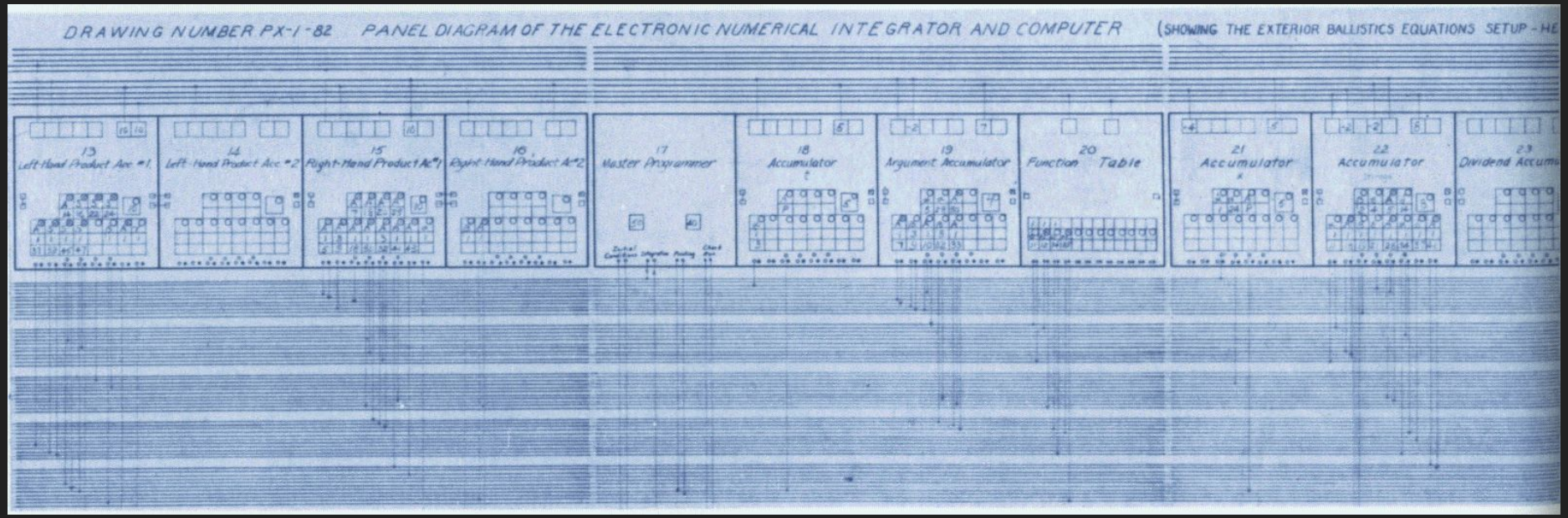
"From a distance, everyone will be able to read text, enlarged and limited to the desired subject, projected on an individual screen."

Paul Otlet (1868 - 1944)


SPECIES MUNDANEUM
X 081
Doc. N° 8504



By Unidentified U.S. Army photographer - Image from Historic Computer Images, Public Domain, https://commons.wikimedia.org/w/index.php?curid=26253110



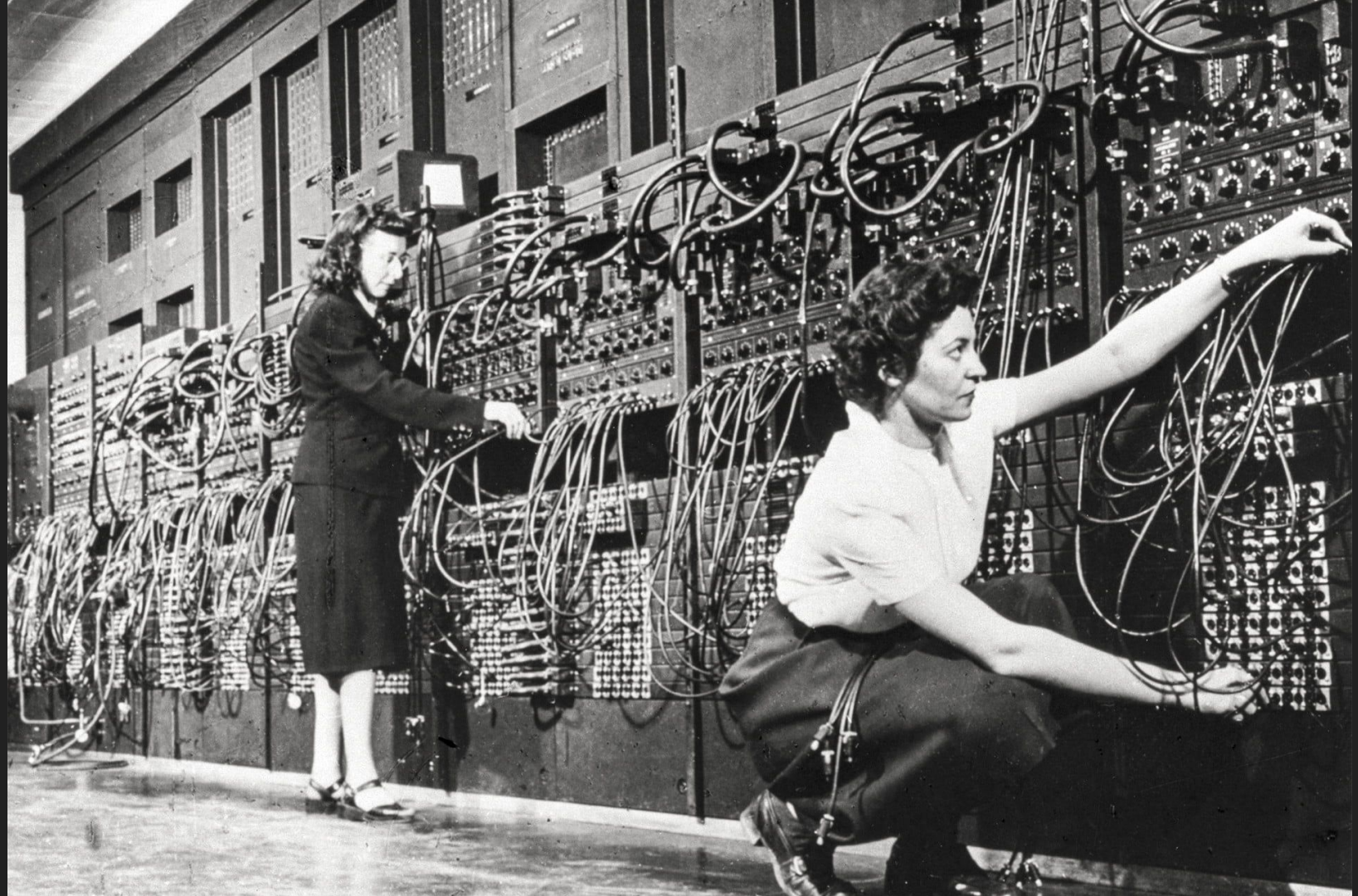
http://www.columbia.edu/cu/computinghistory/eniac.html

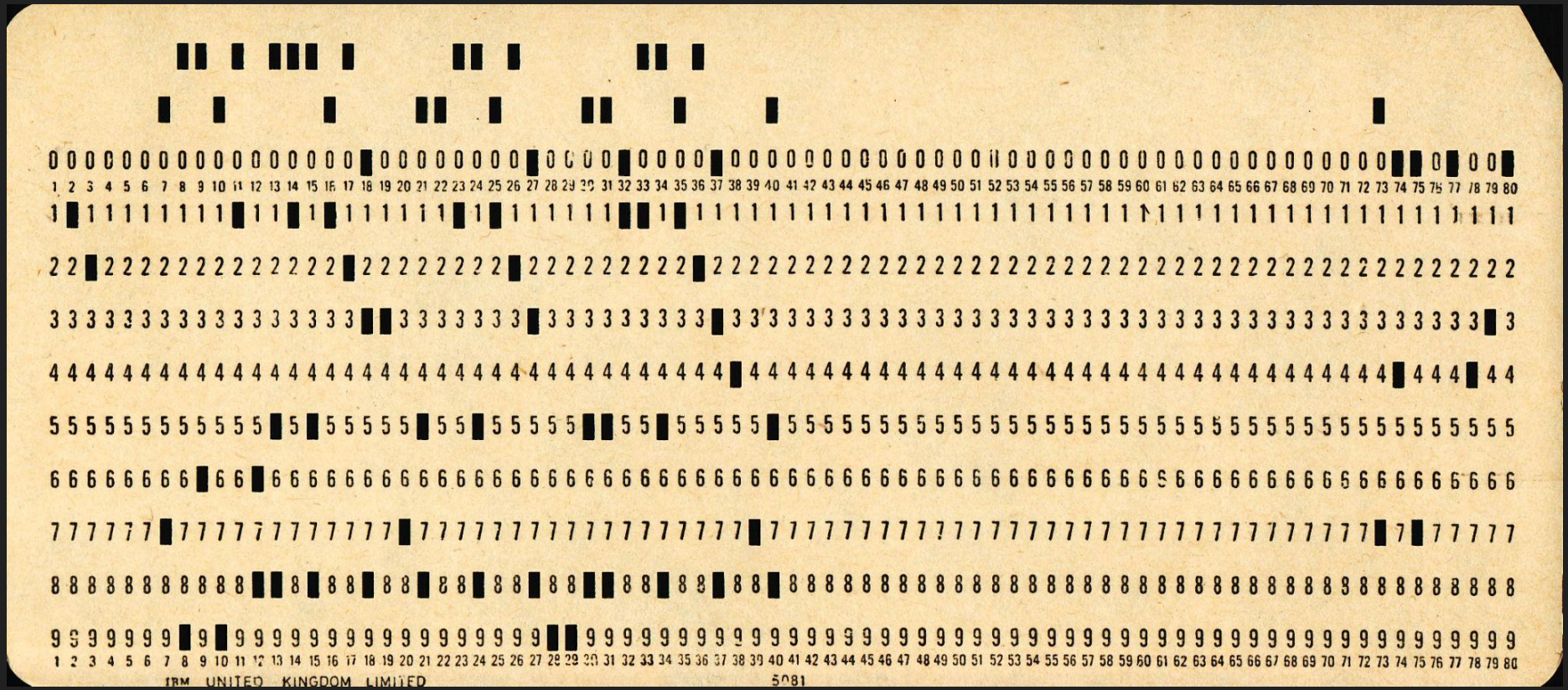
The "ENIAC Six" -- 1946

- *Betty Snyder Holberton*
- *Jean Jennings Bartik*
- *Kay McNulty Mauchly Antonelli*
- *Marlyn Wescoff Meltzer*
- *Ruth Lichterman Teitelbaum*
- *Frances Bilas Spence*

"The Computers: The Remarkable Story of the ENIAC Programmers" -- Kathy Kleimar

https://vimeo.com/ondemand/eniac6



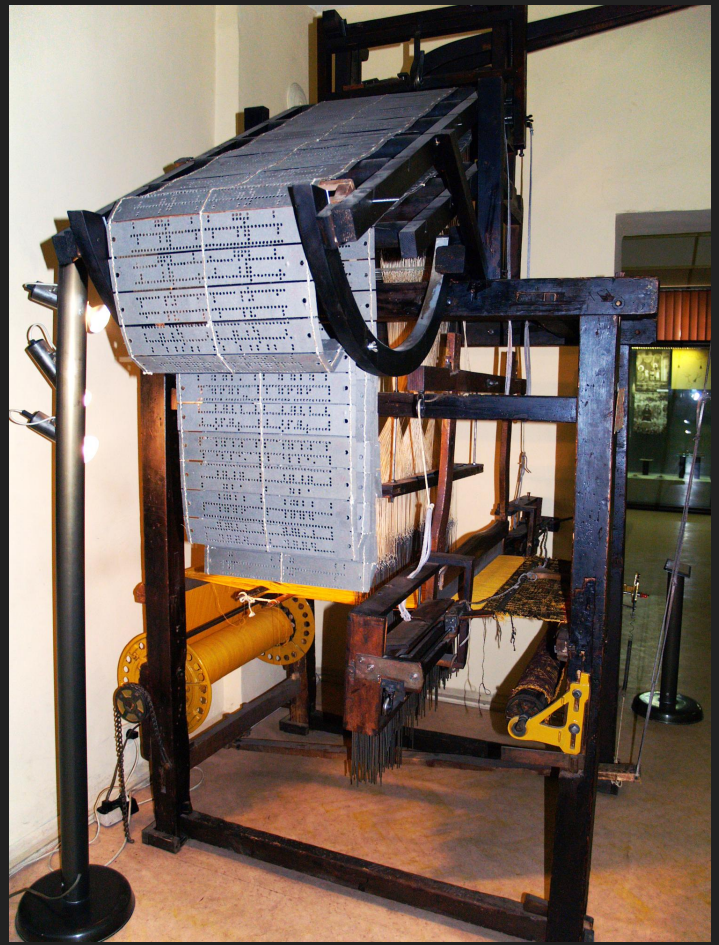


By NASA - Great Images in NASA Description, Public Domain, https://commons.wikimedia.org/w/index.php?curid=6455009

Dorothy Vaughan (1910 - 2008)

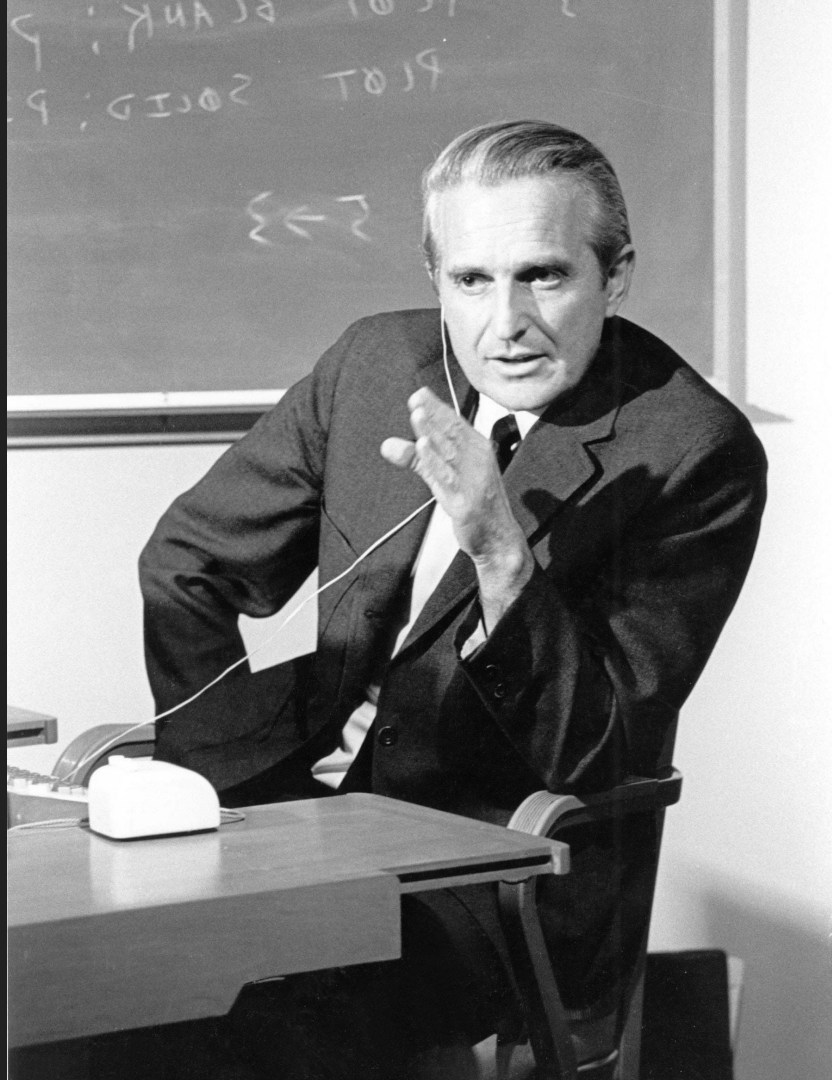
"We're going to need a lot of manpower to program that beast [the IBM 7090]."

-- Dorothy Vaughan
Mathematician
FORTRAN expert

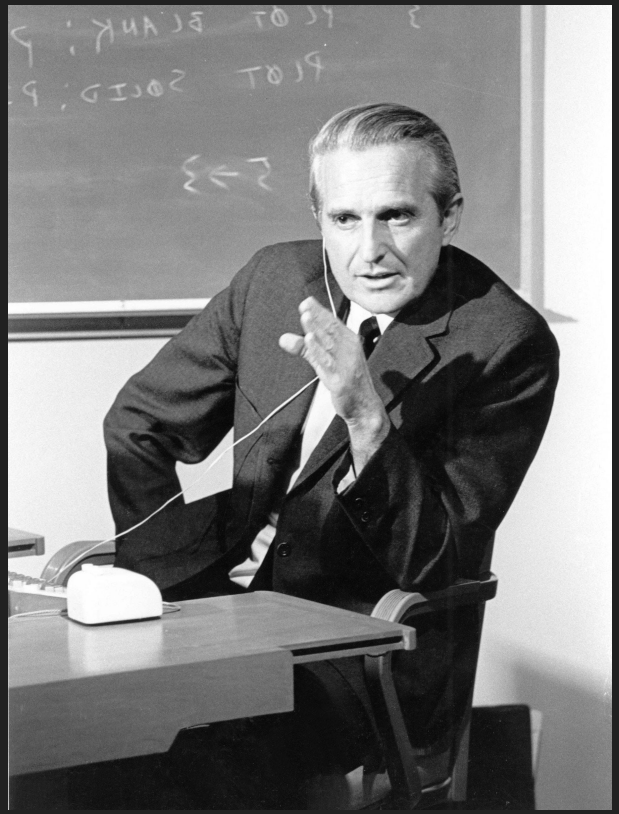



Where We Are
1950-2050

Demo Day 1968

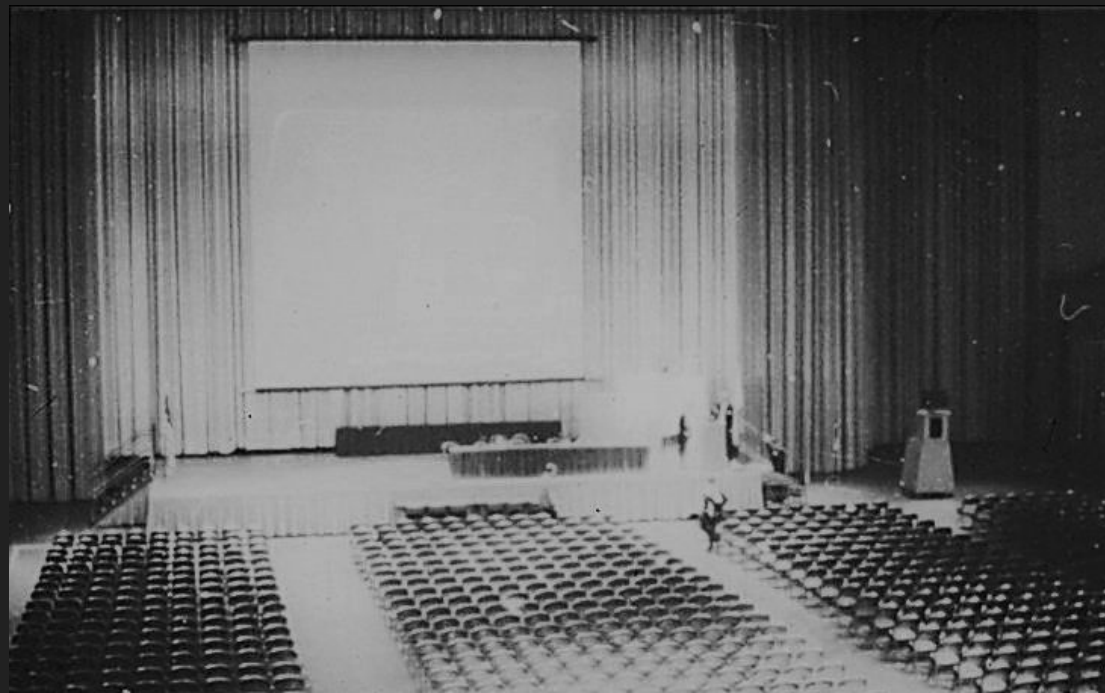


Technology should not aim to replace humans, rather amplify human capabilities.



"In 20 or 30 years, you'll be able to hold in your hand as much computing knowledge as exists now in the whole city, or even the whole world."

— Doug Engelbart (1925 - 2013)

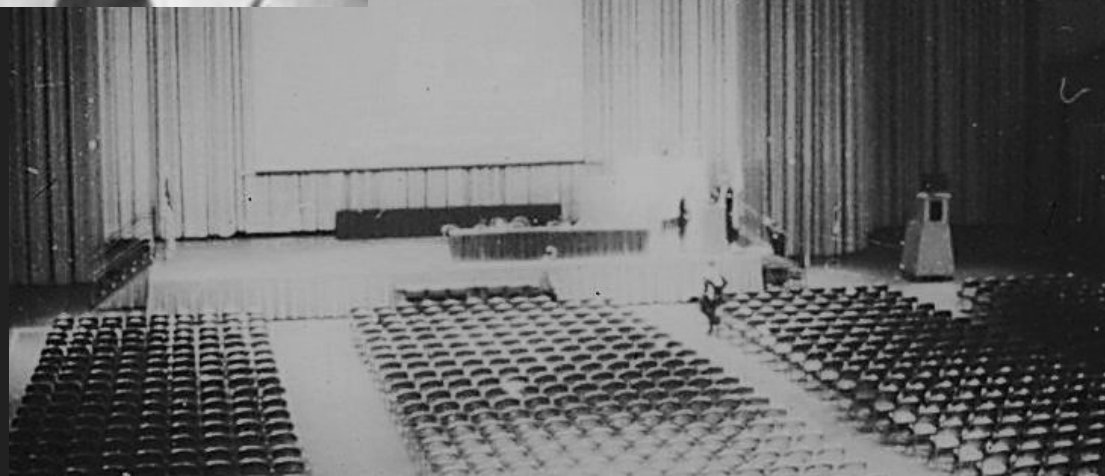


APPLES
BANANAS
CARROTS
LETTUCE
BEANS

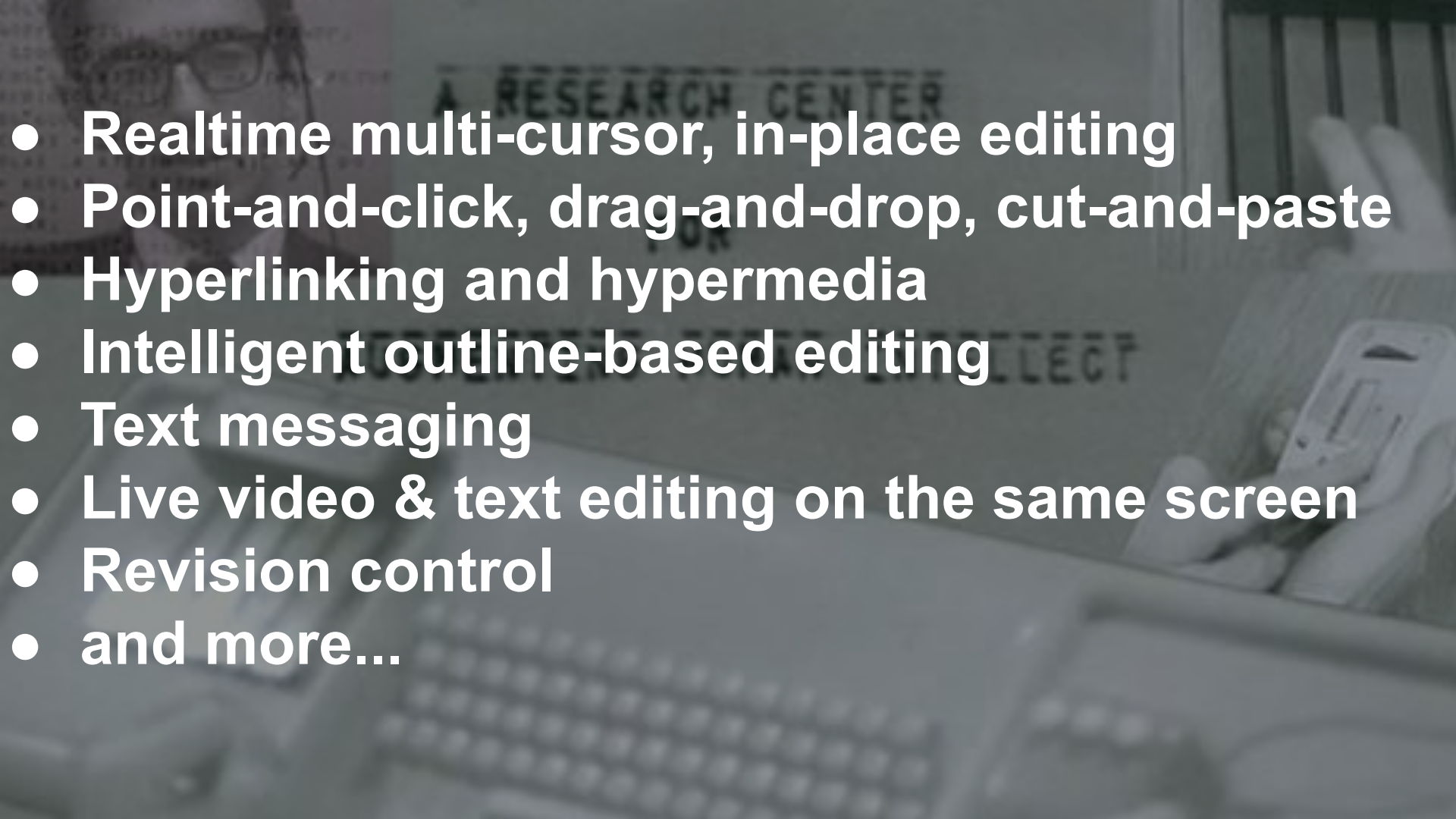
CANS
APPLE SAUCE
BEAN SOUP
TOMATO SOUP

CEREALS
BREAD
NOODLES (ELBOW KIND)
FRENCH BREAD

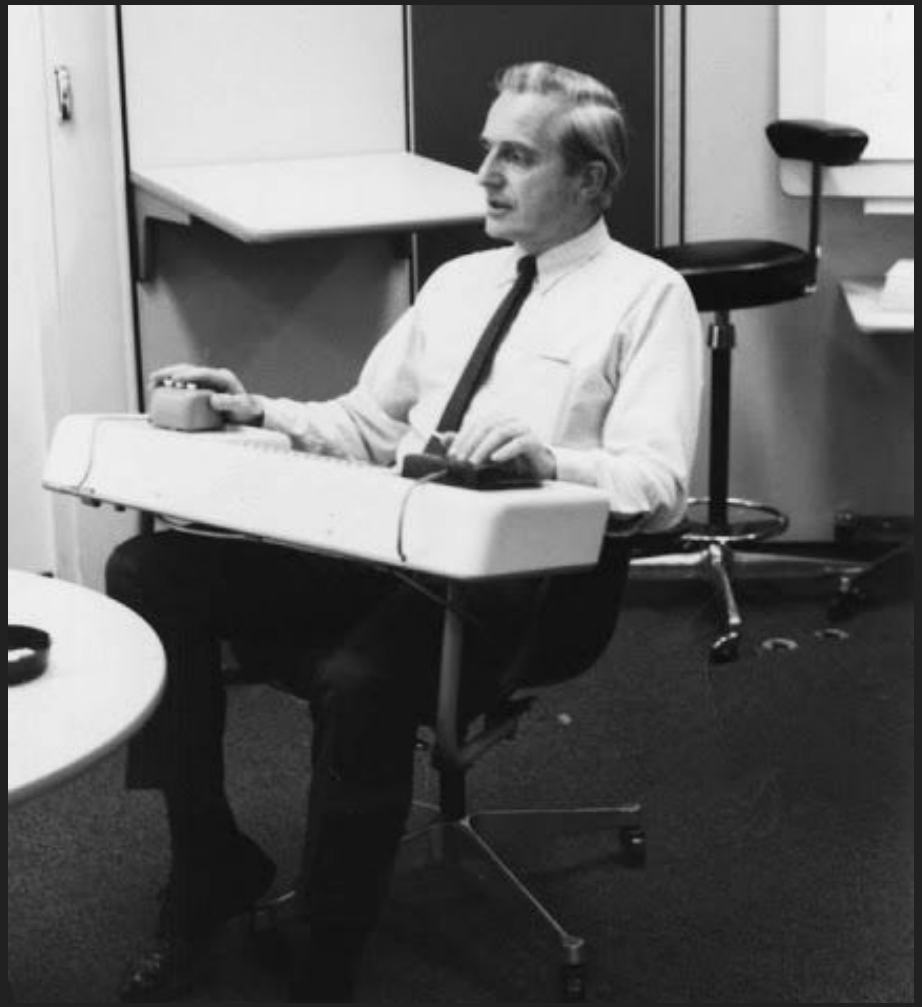
COLD LOCKER
MILK




APPLES
BANANAS
CARROTS
LETTUCE
BEANS
CANS
APPLE SAUCE
BEAN SOUP
TOMATO SOUP
CEREALS
BREAD
COLD LOCKER
MILK



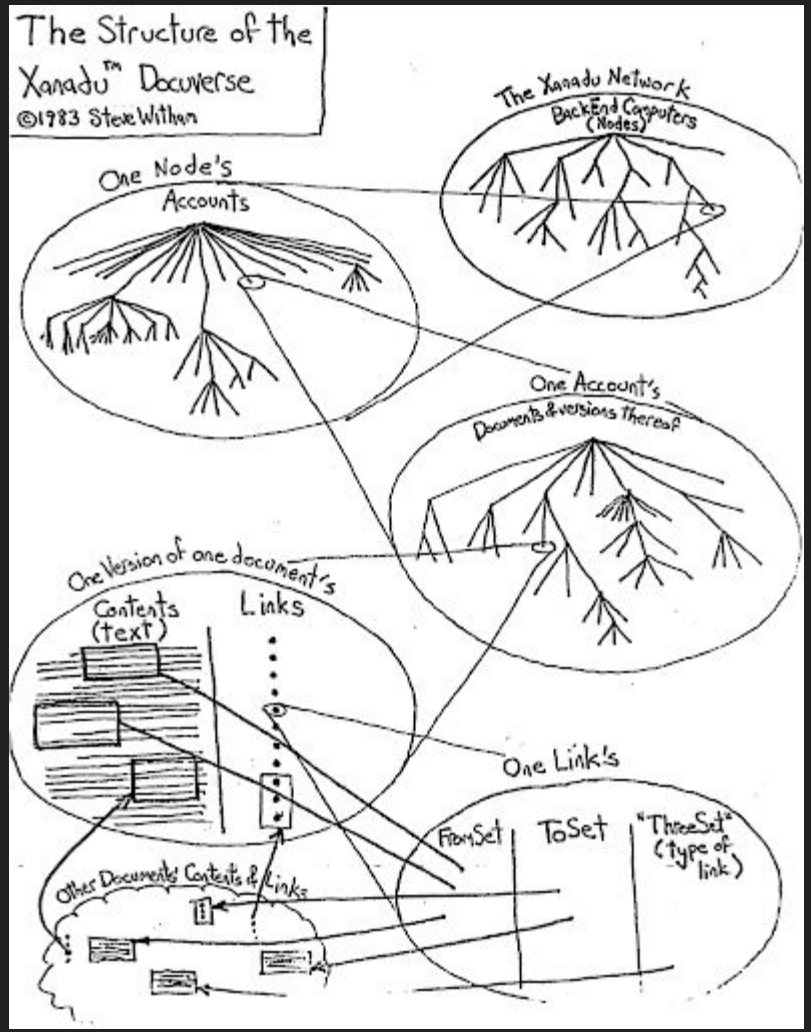
- Realtime multi-cursor, in-place editing
- Point-and-click, drag-and-drop, cut-and-paste
- Hyperlinking and hypermedia
- Intelligent outline-based editing
- Text messaging
- Live video & text editing on the same screen
- Revision control
- and more...

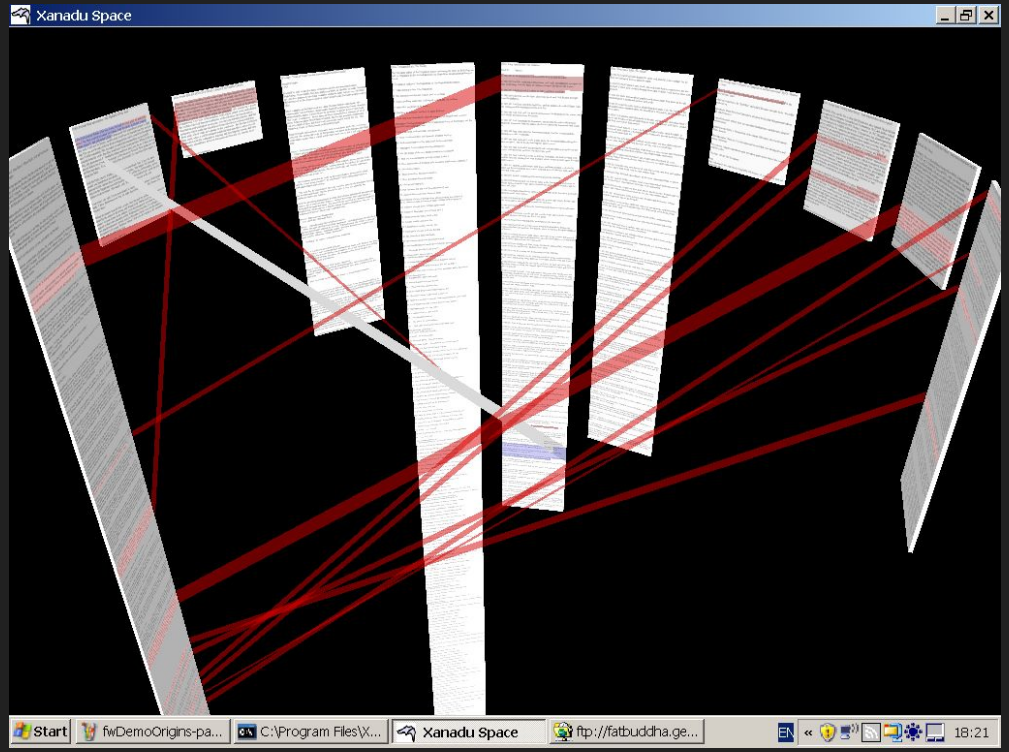


We should not impose regularity.

"The point was to be able to have a medium that would record all the connections and all the structures and all the thoughts that paper could not."

-- Ted Nelson (1937 --)


The Structure of the
Xanadu™ Docuverse
©1983 Ted Nelson
The Xanadu Network
Back End Computers
(Nodes)
One Node's
Accounts
One Account's
Documents & versions thereof
One Version of one document's
Contents
(text)
Links
One Link's
FromSet
ToSet
"ThreeSet"
(type of
link)
Other Documents' Contents & Links


Xanadu Space
Start
fwDemoOrigins-pa...
C:\Program Files\X...
Xanadu Space
ftp://fatbuddha.ge...
18:21

You can and must understand computers NOW.

COMPUTER

DREAM MACHINES

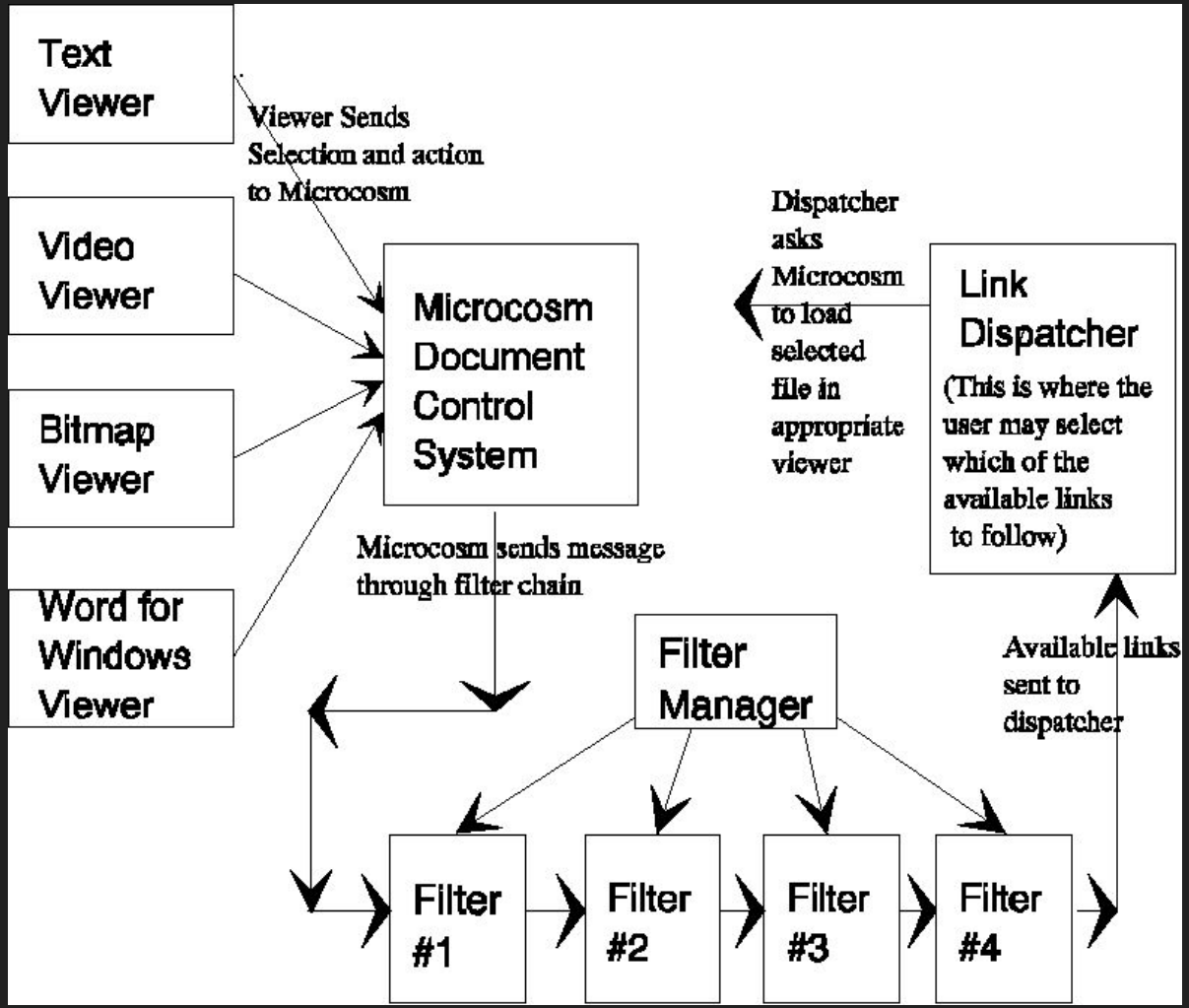
New Freedoms Through Computer Screens
— a Minority Report

This is the flip side of Computer Lib.

The whole foundation of hypertext is collaborative

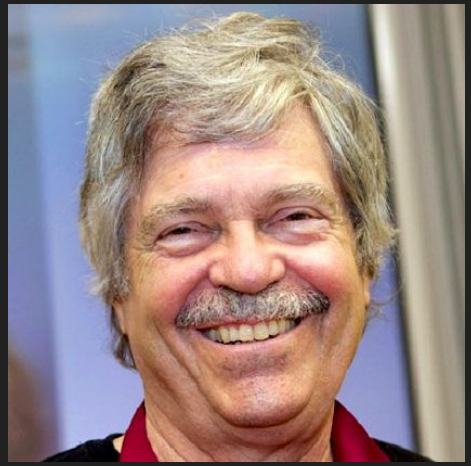
"The strength of the internet is its global interconnection."

-- Dame Wendy Hall (1952 --)

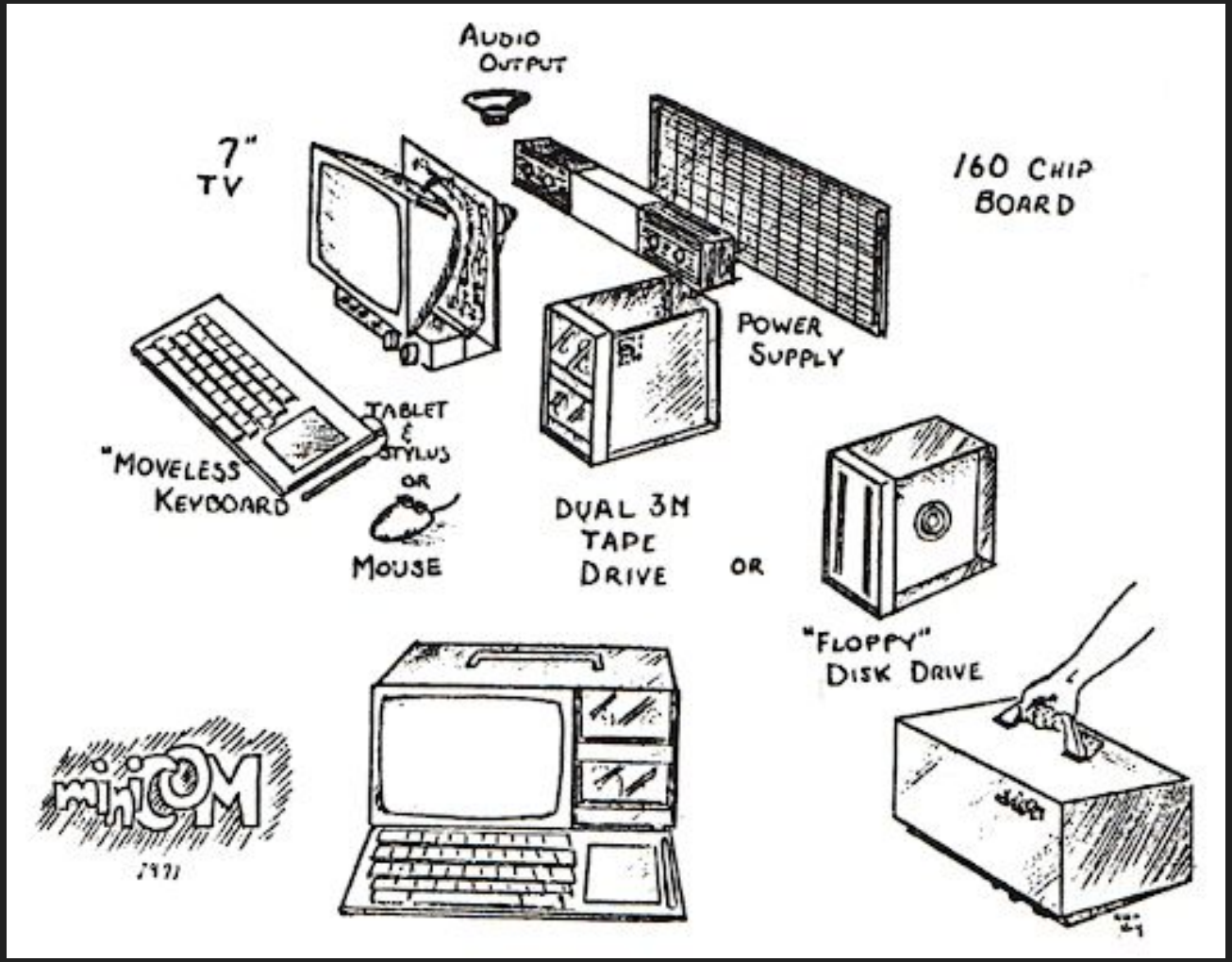

Text Viewer
Video Viewer
Bitmap Viewer
Word for Windows Viewer
Viewer Sends Selection and action to Microcosm
Microcosm Document Control System
Microcosm sends message through filter chain
Dispatcher asks Microcosm to load selected file in appropriate viewer
Link Dispatcher
(This is where the user may select which of the available links to follow)
Available links sent to dispatcher
Filter Manager
Filter #1
Filter #2
Filter #3
Filter #4


Microcosm

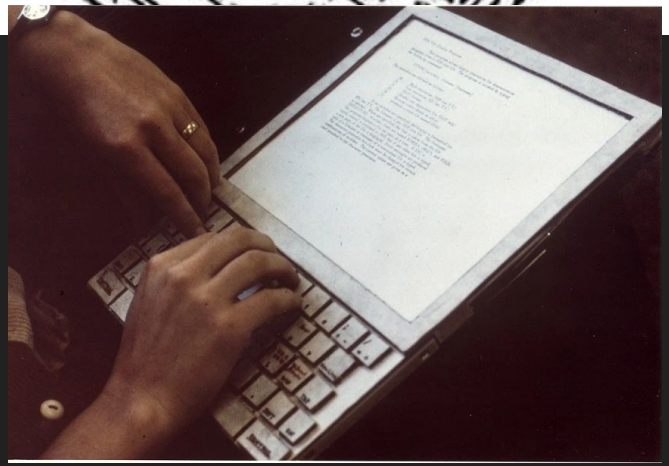
Adele Goldberg & Alan Kay -- 1970s



- *XEROX Parc*
- *Smalltalk Language*
- *Object-Oriented Programming*
- *Dynabook*

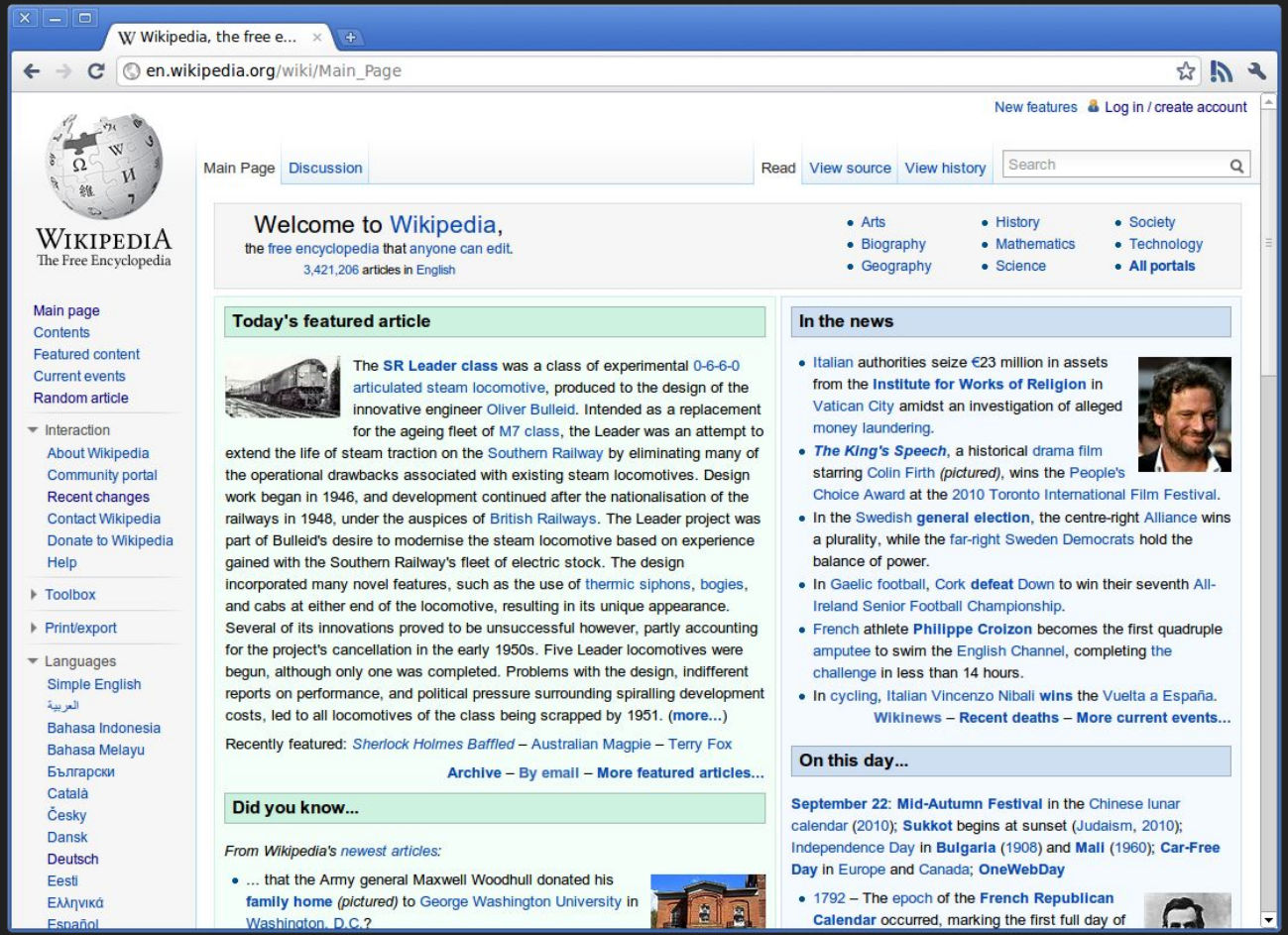

AUDIO OUTPUT
7" TV
160 CHIP BOARD
POWER SUPPLY
"MOVELESS KEYBOARD
TABLET & STYLUS OR MOUSE
DUAL 3M TAPE DRIVE
OR
"FLOPPY" DISK DRIVE
miniCOM
1971


12"
9
4 1/2"
TILES



```
5  with Text_IO; use Text_IO;
6  with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;
7
8  procedure bernoulli is
9    Bern: Integer:= 4;
10
11 begin
12    Put_Line("Which B(n) would you like to compute:");
13    Get(Bern);
14
15    declare
16       N: array(1..Bern+1) of Integer;
17       D: array(1..Bern+1) of Integer;
18       new_denom: Integer;
19       gcd_result: Integer;
20
21       function GCD (X, Y: Integer) return Integer is
22
23          X1: Integer:= X;
24          Y1: Integer:= Y;
25          Old_X :  Integer;
26
27       begin
28
29          while (Y1 /= 0) loop
30             -- x, y := y, x mod y
31             Old_X := X1;
32             X1 := Y1;
33             Y1 := Old_X mod Y1;
34          end loop;
35
36          return X1;
37
38       end GCD;
39
```

https://marquisdegeek.com/code_ada99



By Own work (Own screenshot), BSD, https://commons.wikimedia.org/w/index.php?curid=11562194

Where We Will Be

2050 - ??

Let's Review...

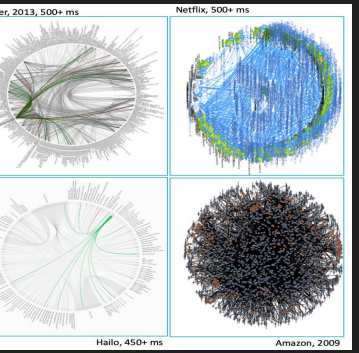
- Written instructions in tables (Lovelace) for mechanical computers (Babbage)
- Wiring diagrams (McNulty, etc.) for electrical relay computers (ENIAC)
- Punch cards (Vaughn) for electronic computers (IBM)
- Screen text (Engelbart) for connected semiconductor computers (Hall)



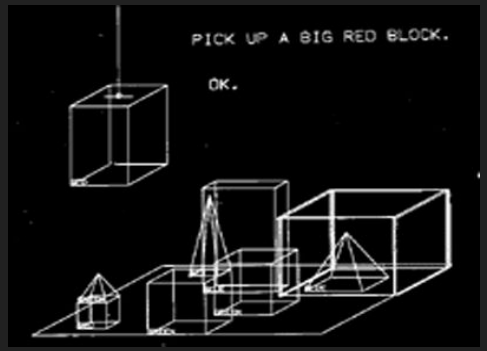

What's Next?

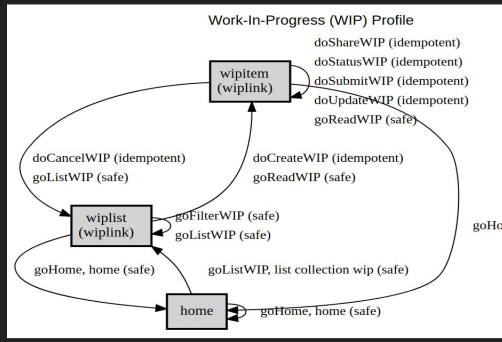
In the Future we will...

- **Connect** services the way we currently connect servers
- **Program** the network instead of single machines
- **Describe** problem spaces, not solutions
- Machines, endpoints, protocols and formats will "**disappear**"

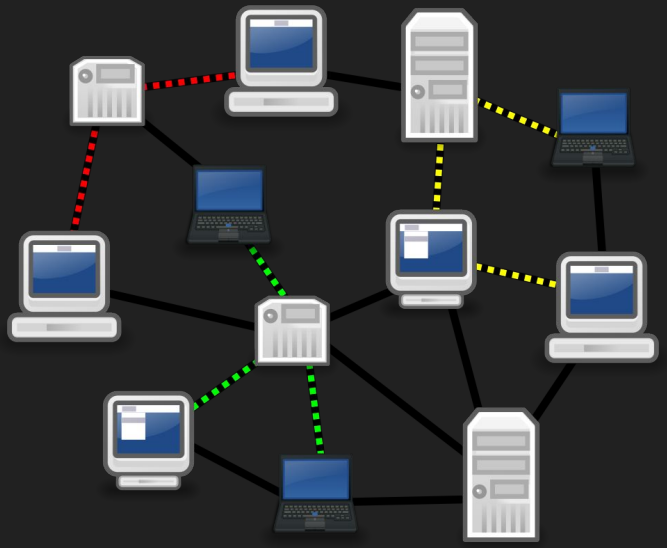


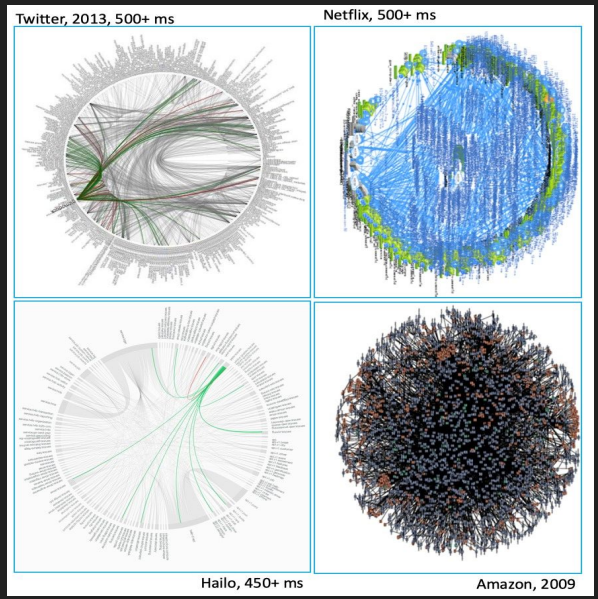

```
#
# Get weekly groceries
#

# load control data
CONFIG LOAD grocery.config

# find running services for shopping, shipping, & payment
REQUEST WITH-URL $$service-registry$$
CALL WITH-FORM search WITH-PARAMS {"name": "shopping-cart"}
STACK PUSH WITH-RESPONSE cart
CALL WITH-FORM search WITH-PARAMS {"name": "shipping"}
STACK PUSH WITH-RESPONSE shipping
CALL WITH-FORM search WITH-PARAMS {"name": "payment"}
STACK PUSH WITH-RESPONSE payment

# load the cart, set delivery, and pay
CALL WITH-NAME fill-cart WITH-CONFIG $$shopping-items$$
CALL WITH-NAME shipping-address WITH-CONFIG $$home-address$$
CALL WITH-NAME submit-payment WITH-CONCFIG $$payment-profile$$

# all done
ECHO "Groceries will arrive on $$shipping-date$$!"
```





From Connected Machines to Connected Services





https://www.infoq.com/articles/cloud-native-architecture-adoption-part2/

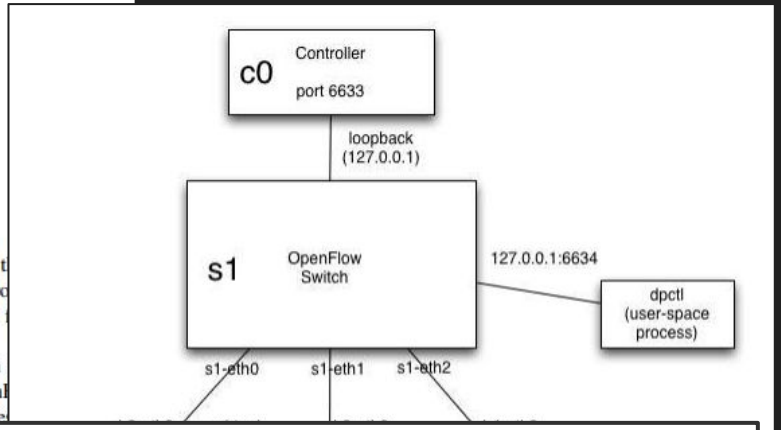
Programming the Network Instead of Single Machines

Languages for Software-Defined Networks

Nate Foster*, Michael J. Freedman†, Arjun Guha*, Rob Harrison‡,
Naga Praveen Katta†, Christopher Monsanto†, Joshua Reich†, Mark Reitblatt*,
Jennifer Rexford†, Cole Schlesinger†, Alec Story*, and David Walker†
*Cornell University †Princeton University ‡U.S. Military Academy

***Abstract*—Modern computer networks perform a bewildering array of tasks, from routing and traffic monitoring, to access control and server load balancing. Yet, managing these networks is unnecessarily complicated and error-prone, due to a hetero-**

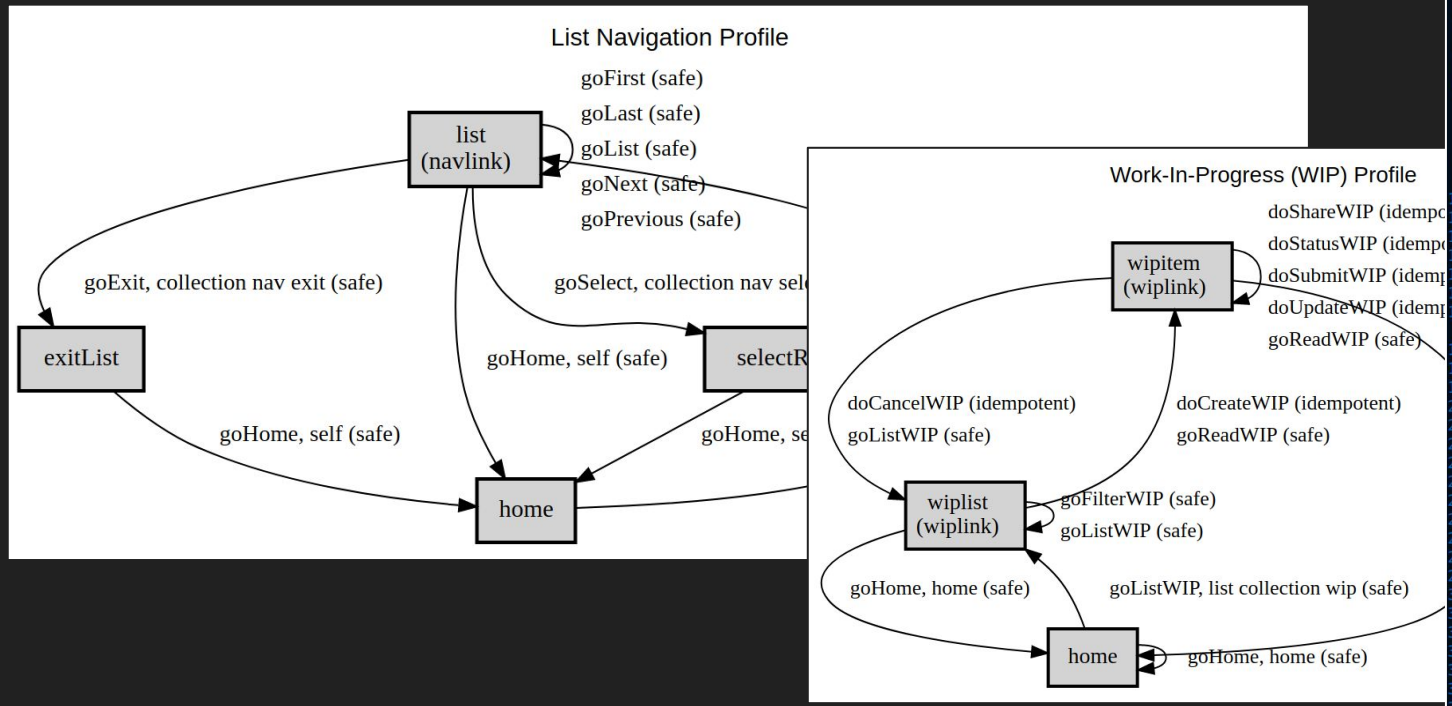
forwarding paths for each user [5]. To balance t
between back-end servers in a data center, the contro
split flows over several server replicas and migrate
[6], [7].
sible to program
ols such as Openl
tches the feature

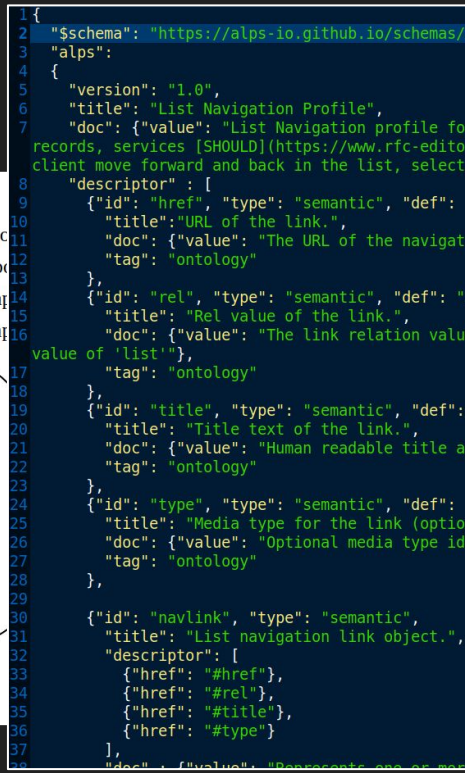



```
- hosts:
    - pc1.example.com
    - pc3.example.com
  tasks:
    - name: install Apache
      action: apt pkg=apache2 state=present
    - name: ensure Apache is running
      action: service name=apache2 state=ru
- hosts: dns_servers
  roles:
    - dns_server
    - ntp
```

```
# SIREN example
GOTO http://rwcbook10.herokuapp.com
SIREN LINKS
SIREN ENTITIES
SIREN ACTIONS

GOTO WITH-REL taskFormListByUser WITH-QUERY {"assignedUser" : "alice"}
```

Focus on Describing Problem Spaces, not Solutions





```
{
  "$schema": "https://alps-io.github.io/schemas/
  "alps":
  {
    "version": "1.0",
    "title": "List Navigation Profile",
    "doc": {"value": "List Navigation profile fo
records, services [SHOULD](https://www.rfc-edito
client move forward and back in the list, select
    "descriptor" : [
      {"id": "href", "type": "semantic", "def":
        "title":"URL of the link.",
        "doc": {"value": "The URL of the navigat
        "tag": "ontology"
      },
      {"id": "rel", "type": "semantic", "def": "
        "title": "Rel value of the link.",
        "doc": {"value": "The link relation valu
value of 'list'"},
        "tag": "ontology"
      },
      {"id": "title", "type": "semantic", "def":
        "title": "Title text of the link.",
        "doc": {"value": "Human readable title a
        "tag": "ontology"
      },
      {"id": "type", "type": "semantic", "def":
        "title": "Media type for the link (optio
        "doc": {"value": "Optional media type id
        "tag": "ontology"
      },

      {"id": "navlink", "type": "semantic",
        "title": "List navigation link object.",
        "descriptor": [
          {"href": "#href"},
          {"href": "#rel"},
          {"href": "#title"},
          {"href": "#type"}
        ],
```

https://rwmbook.github.io/alps-documents/

Machines, Endpoints, Protocols, Formats will "disappear"

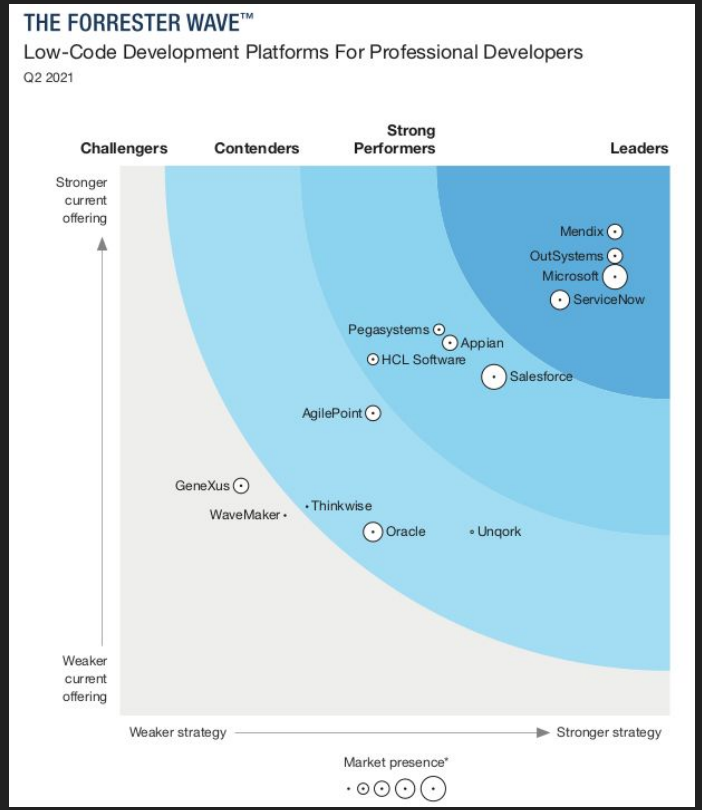
```
#
# Get weekly groceries
#

# load control data
CONFIG LOAD grocery.config

# find running services for shopping, shipping, & payment
REQUEST WITH-URL $$service-registry$$
CALL WITH-FORM search WITH-PARAMS {"name": "shopping-cart"}
STACK PUSH WITH-RESPONSE cart
CALL WITH-FORM search WITH-PARAMS {"name": "shipping"}
STACK PUSH WITH-RESPONSE shipping
CALL WITH-FORM search WITH-PARAMS {"name": "payment"}
STACK PUSH WITH-RESPONSE payment

# load the cart, set delivery, and pay
CALL WITH-NAME fill-cart WITH-CONFIG $$shopping-items$$
CALL WITH-NAME shipping-address WITH-CONFIG $$home-address$$
CALL WITH-NAME submit-payment WITH-CONCFIG $$payment-profile$$

# all done
ECHO "Groceries will arrive on $$shipping-date$$!"
```

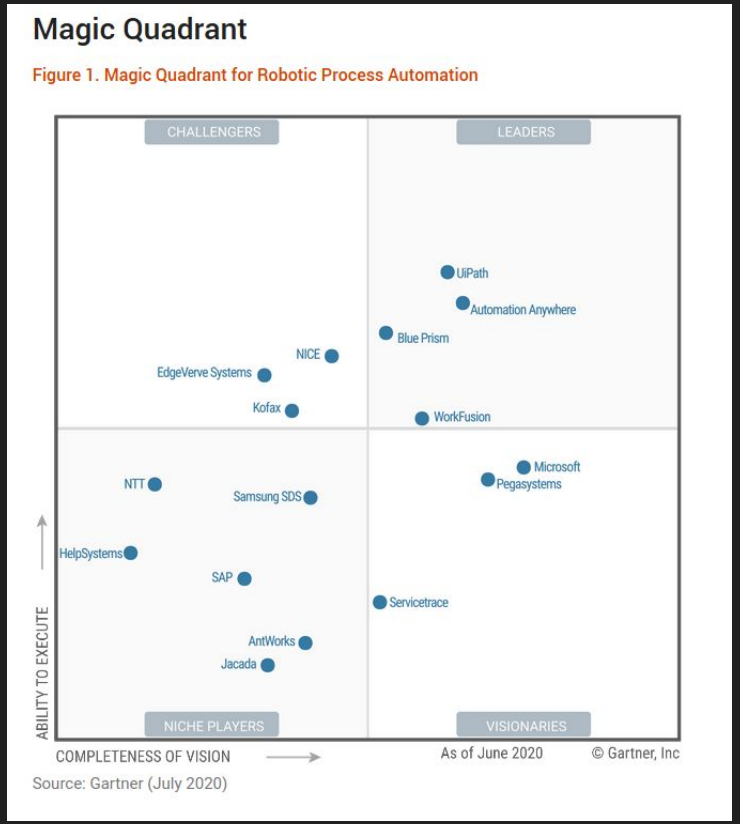
https://github.com/rwmbook/hyper


And So...

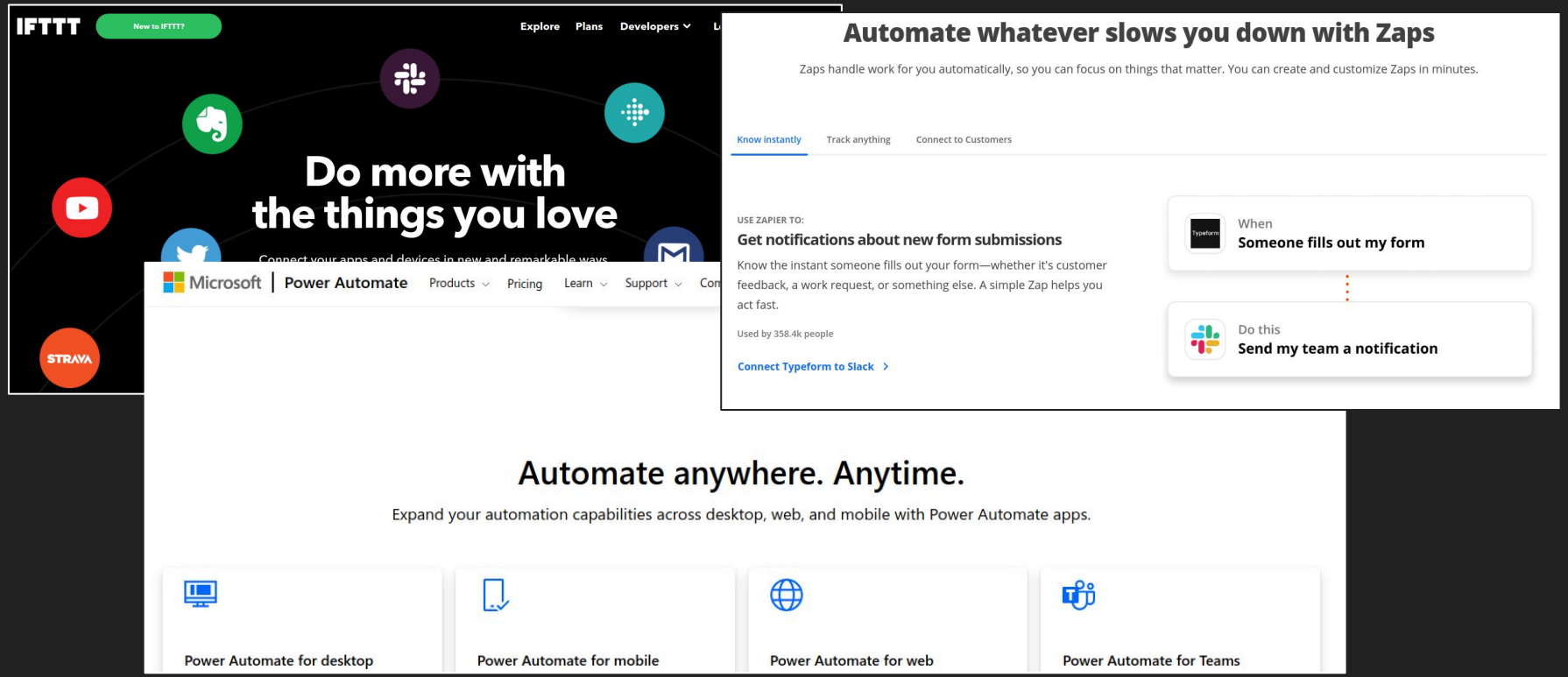
What can you do right now?

Low-Code & Automation -- *Be Prepared*

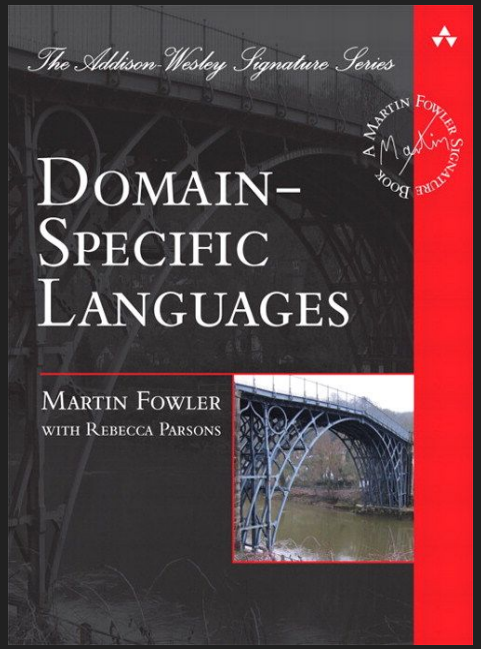


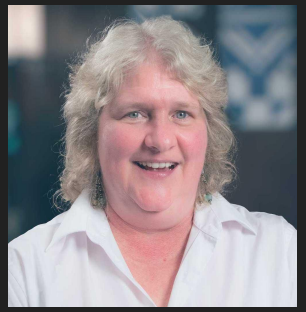


Decentralized Orchestration -- *Start Exploring*



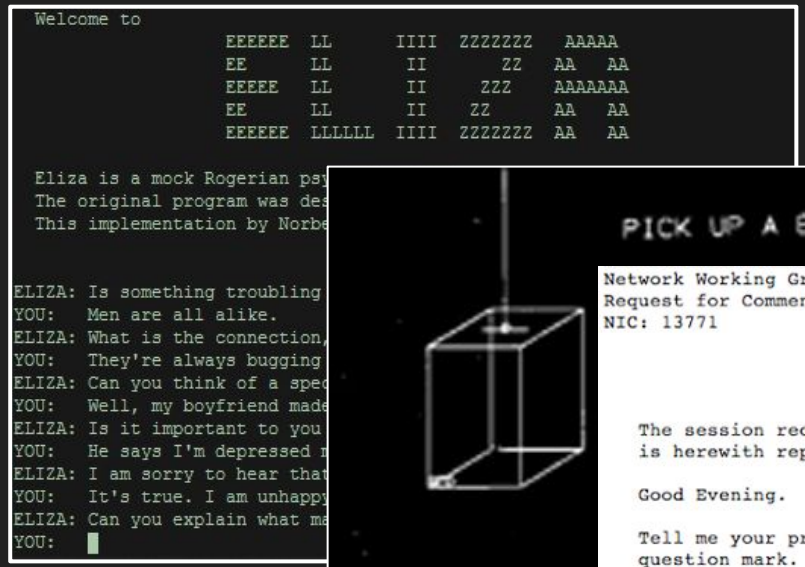
Domain-Specific Languages -- *Stay Tuned*





- Parsons & Fowler book
- DevOps spawned several DSL
- RPA and Low-Code will create more
- Think :
  - GDPR-lang
  - HIPPA-lang
  - BIAN-lang
  - FHIR-lang
  - ACCORD-lang

AI, Chatbots, and Speech -- *Be Wary!*



Network Working Group V. Cerf
Request for Comments: 439 SU-ERL
NIC: 13771 21 January 1973

PARRY Encounters the DOCTOR

The session recorded below actually happened on 18 September 1972 and is herewith reproduced for

Good Evening.

Tell me your problems. Ple question mark.

*People get on my nerves so

I am not sure I understand

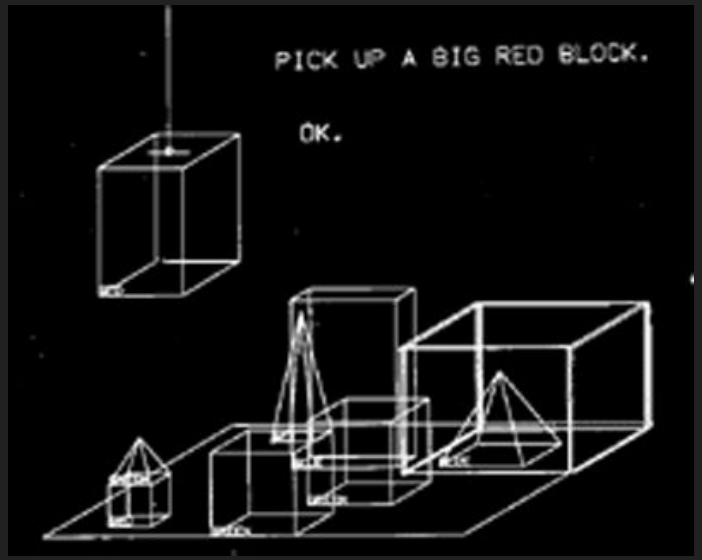
*You should pay more attent

Suppose you should pay more

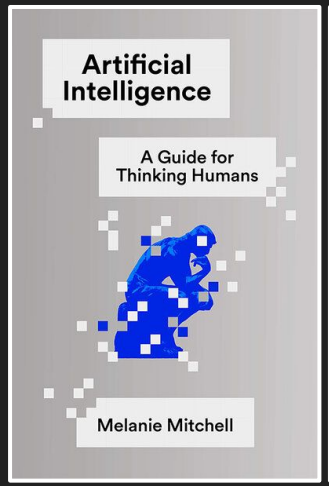
*You're entitled to your ow

Alternatively, look to Task-Focused Microworlds (TFMs)

- Microworlds are domain-specific
- Task-focused means no need to plan, learn, reason
- TFM's can scale
- Think topic-specific help systems:
  - DoNotPay.com (parking)
  - Various Health "experts"
  - CoPilot at Github (?)



Artificial Intelligence & Machine Learning
Stay Informed



Melanie Mitchell & Joy Buolamwini

HyperCLI & HyperLANG: A DSL for APIs -- *Experiment*

hyper : Interactive Hypermedia Shell

Exploring an interactive REPL/shell for interacting with HTTP-based hypermedia services

Summary

The **hyper** utility is a simple command-line style shell/REPL for interacting with an online services/APIs. While a fully-functional HTTP client, **hyper** is especially good at dealing with hypermedia services including Collection+JSON, SIREN, and HAL. There are plans to add support for PRAG+JSON, MASH+JSON, and possibly UBER in the future.

Along with HTTP- and mediatype-aware commands, **hyper** also supports some convience functionality like SHELL commands, configuration file management, and a LIFO stack to handle local memory variabes.

Importantly, **hyper** is not just a shell/REPL, it is a hypermedia DSL. It encourages users to `think' in hypermedia. Rather than writing complex HTTP queries that look like this (an example that works fine in **hyper**):

```
ACTIVATE http://localhost:8181/task/
  WITH-METHOD PUT
  WITH-BODY title=testing&tags=hyper&completeFlag=false
  WITH-ENCODING application/x-www-form-urlencoded
  WITH-HEADERS {"if-none-match":"*"}
```

The **hyper** shell can also use mediatype-aware convience commands to locate, parse, fill, and execute inline hypermedia controls. This results in a much more readable **hyper** exeperience:

```
STACK PUSH {
  "title":"testing",
  "tags":"hyper",
  "completeFlag":"false"
}

ACTIVATE http://locahost:8181/home/
ACTIVATE WITH-FORM taskFormAdd WITH-STACK
```

```
#
# testing SIREN support
#

# get resource
GO WITH-URL http://rwcbook10.herokuapp.com/

# view representation
SIREN LINKS
SIREN PROPERTIES
SIREN ENTITIES
SIREN ACTIONS

# select elements in the response
SIREN ID rmqzgqfq3d
SIREN NAME taskFormAdd
SIREN REL self

# execute JSONPath query
SIREN PATH $.entities.*[?(@property=='id'&&@.match(/rmqzgqfq3d/i))]^.[id,title,href,type]

# use rels & names to make requests w/ args
GO WITH-REL self
GO WITH-NAME taskFormListByUser WITH-QUERY {"assignedUser":"alice"}
SIREN ENTITIES

#
# EOF
#
```

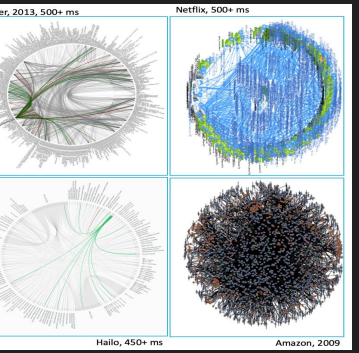
HyperCLI & HyperLANG: A DSL for APIs -- *Experiment*

```
#
# WITH-FORM testing : @rwmbook 2021-06
#
# The forms named taskFormListByUser and taskFormAdd are in the response
# WITH-FORM {formname} pulls all the HTTP details (method, url, encoding)
# WITH-STACK takes the item on the top of the stack to fill in form fields

# get the list representation
GOTO http://rwcbook10.herokuapp.com

# add to the stack & execute the query
STACK PUSH {"assignedUser":"alice"}
GOTO WITH-FORM taskFormListByUser WITH-STACK

# add to the stack and execute the write
STACK PUSH {"title":"just\.\another\.\one","tags":"with-test","completeFlag":"false"}
GOTO WITH-FORM taskFormAdd WITH-STACK

EXIT

#
# EOF
#
```

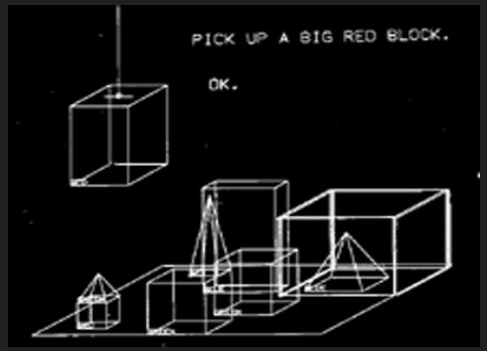
https://github.com/rwmbook/hyper

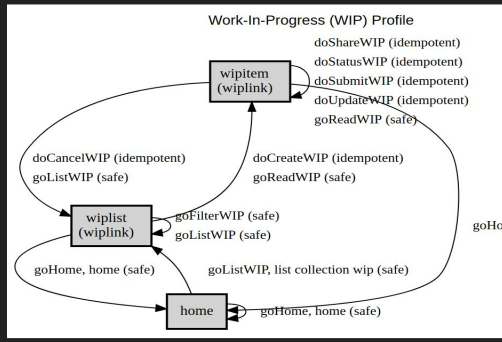
In the Future, we will ...

- Program the Network of Services,
- With Domain-Specific languages,
- To Enable Task-Focused Bots,
- Operating in Well-Described Problem Spaces.




```
#
# Get weekly groceries
#

# load control data
CONFIG LOAD grocery.config

# find running services for shopping, shipping, & payment
REQUEST WITH-URL $$service-registry$$
CALL WITH-FORM search WITH-PARAMS {"name": "shopping-cart"}
STACK PUSH WITH-RESPONSE cart
CALL WITH-FORM search WITH-PARAMS {"name": "shipping"}
STACK PUSH WITH-RESPONSE shipping
CALL WITH-FORM search WITH-PARAMS {"name": "payment"}
STACK PUSH WITH-RESPONSE payment

# load the cart, set delivery, and pay
CALL WITH-NAME fill-cart WITH-CONFIG $$shopping-items$$
CALL WITH-NAME shipping-address WITH-CONFIG $$home-address$$
CALL WITH-NAME submit-payment WITH-CONCFIG $$payment-profile$$

# all done
ECHO "Groceries will arrive on $$shipping-date$$!"
```



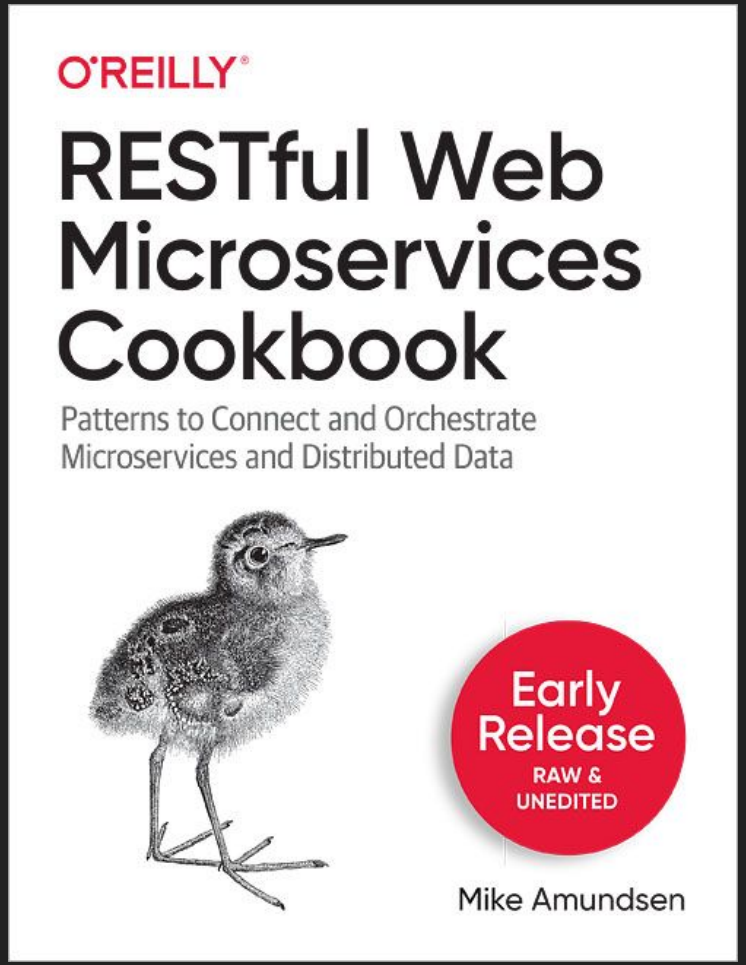


One more thing...

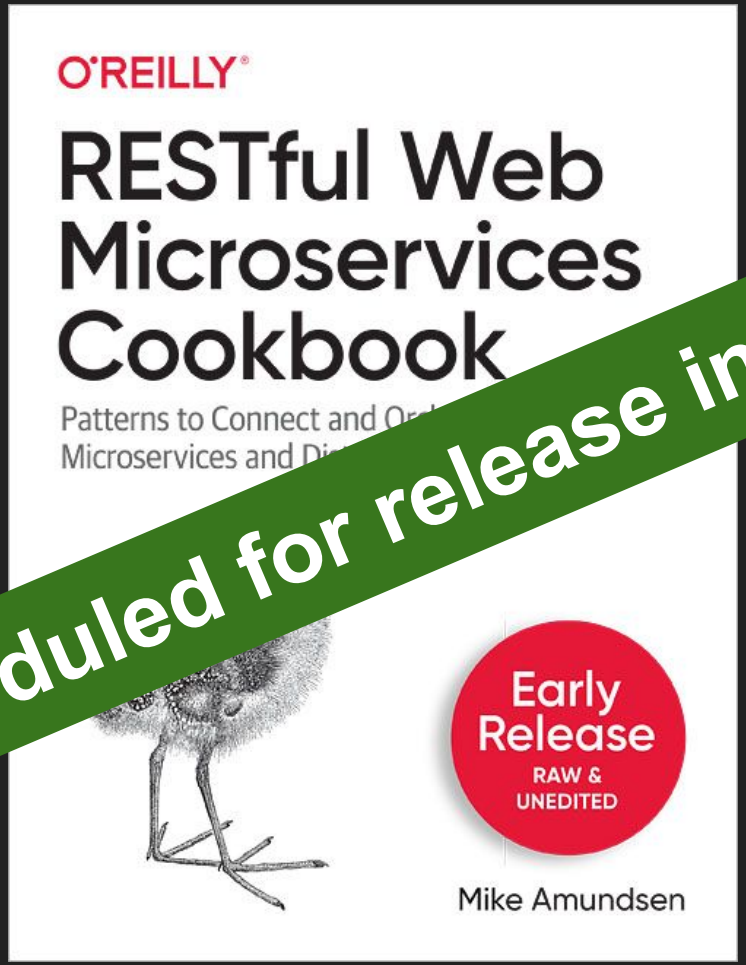
We must learn from the future!

"Those who ignore the mistakes of the future are bound to make them."

Joseph D. Miller, 2006



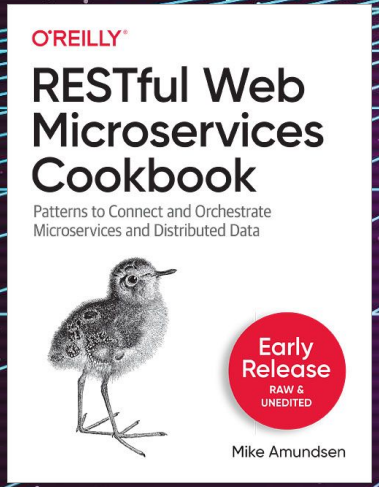
https://learning.oreilly.com/library/view/restful-web-microservices/9781098106737/



Scheduled for release in 2022

https://learning.oreilly.com/library/view/restful-web-microservices/9781098106737/

APIs of the Future: Are You Ready?



Mike Amundsen
@mamund
http://linkedin.com/mamund
youtube.com/mamund