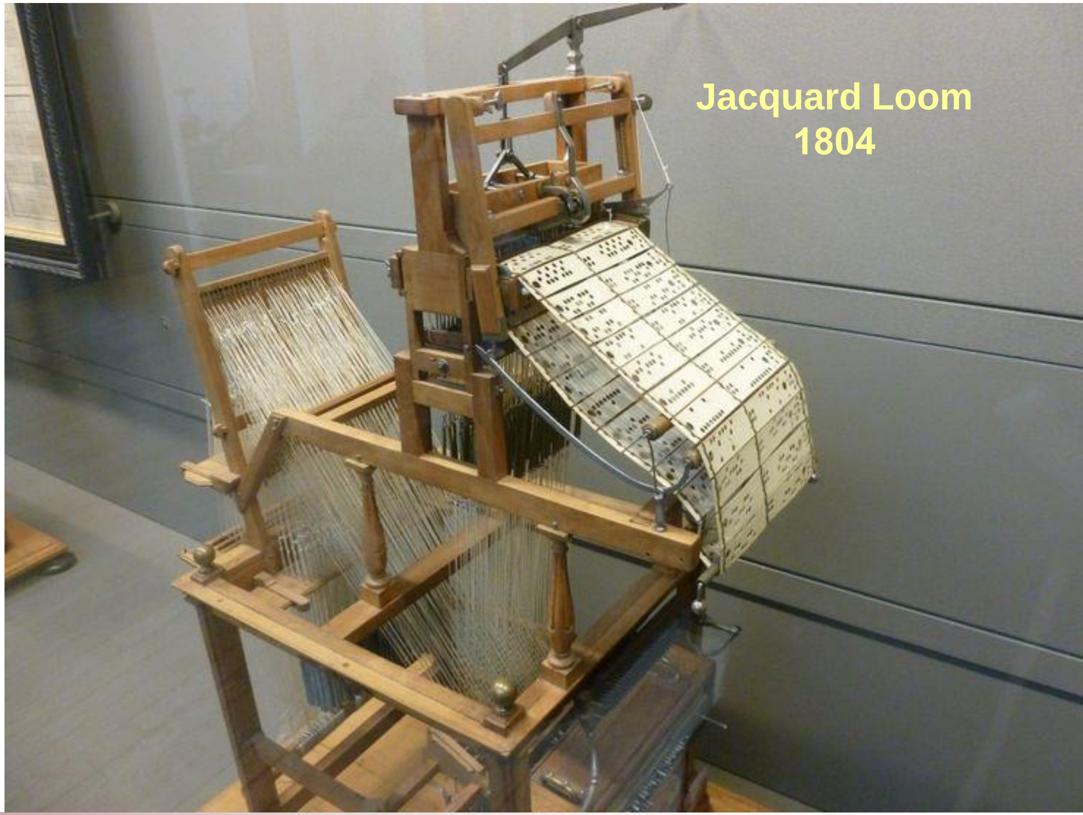
Engineering Software Intensive Systems

Introduction

Lecture 1

20 March 2025

Slides created by
Prof Amir Tomer
tomera@cs.technion.ac.il



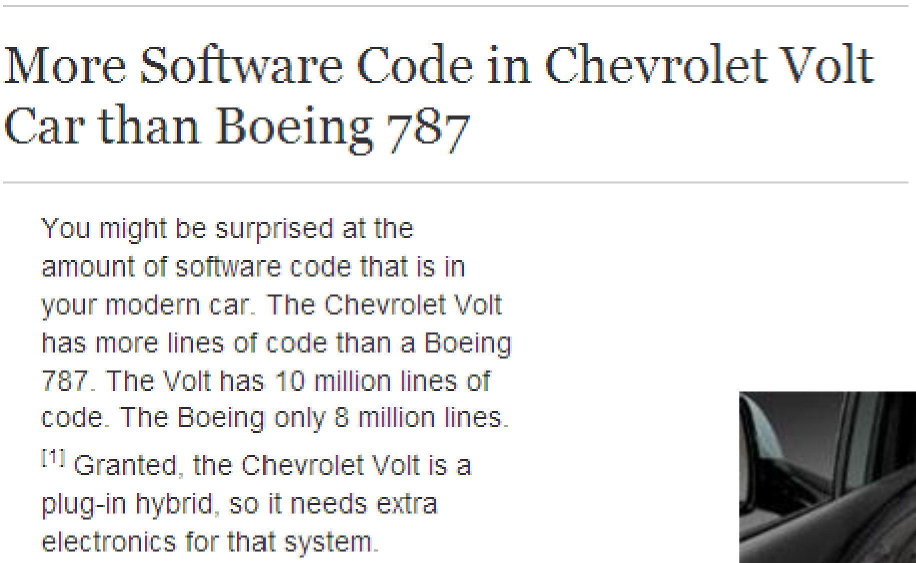
Jacquard Loom
1804

Topics for Today

- Introduction to Software Intensive Systems
- Software from a Systems Perspective

© Prof. Amir Tomer

Motivation: Most systems developed today are software intensive

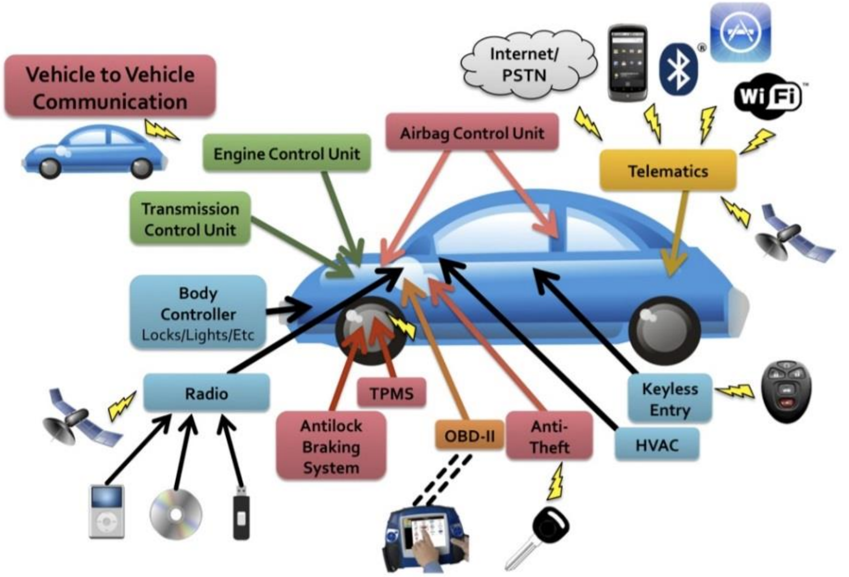


More Software Code in Chevrolet Volt Car than Boeing 787

You might be surprised at the amount of software code that is in your modern car. The Chevrolet Volt has more lines of code than a Boeing 787. The Volt has 10 million lines of code. The Boeing only 8 million lines. [1] Granted, the Chevrolet Volt is a plug-in hybrid, so it needs extra electronics for that system.

Text source: http://blissfulwriter.hubpages.com/hub/Software-Code-in-Your-Car
Picture source: http://www.ramanmedianetwork.com/ibm-software-drives-chevrolet-volt/

Where is the software?



- Component/Element Level
  - Software operates the element/component in the system
    - Ex: Antilock Braking System
  - Implementation based on: Software Design
- System Level
  - Software coordinates, synchronizes, enables joint operation of all parts of the system
    - Ex: Central computer on a communication network
  - Implementation based on: System and Software Architecture
- System of Systems
  - Software enables interoperability between different systems
    - Ex: Communication between autonomous vehicles and stop lights
  - Implementation based on: Network Architecture

Course Goals

- Transition from **programming** to **developing software intensive systems**
  - See the bigger picture
    - System level
    - The entire lifecycle
  - Focus on the whole development process (not just on the product)
- Personal experience with a software project
  - The various steps and milestones
  - Teamwork
- What you will get out of it
  - Understand how software is developed and the problems faced doing so
  - Understand central processes and advanced software engineering methods
  - Understand how the processes and methods affect the quality and reliability of software
  - Get a fundamental understanding of how to work on a software development team

© Prof. Amir Tomer

Course Details and Requirements

Meetings

- 13 lectures, 4 hours each
- Lecture contents found on the syllabus and course home page

Recitations

- Lectures integrate recitations and practice tasks with MS Office and Enterprise Architect

Course Requirements

- Attendance
- Turn in semester project (teams of 3-4 students)
- Project grade is the course grade (no exam)
- Project contains only the planning part of the project
- Development can be done on your own for another course

What's special about software

- Complexity ◀
  - Number of potential states
    - It's hard to understand everything
    - It's hard to test everything
- Conformity ◀
  - Software has no physical restraints
    - It must adapt itself to everything else
    - It must make up for limitations in other disciplines
- Invisibility ◀
  - There is no tangible physical model
- Changeability and Maintainability ◀
  - Upgrade cost = development time only
    - No physical materials
    - No production cost (except reinstallation)
    - No packaging and shipping
  - No component failure
    - Every failure is a design failure
    - If one copy fails, all will fail
    - How do you define "reliable software"?

> ***"…building software will always be hard. There is inherently no silver bullet."***
>
> **F. Brooks (1986)***

***Brooks, F.P., *No Silver Bullet: Essence and Accidents of Software Engineering*, IFIP 10th World Computing Conference, 1986**

© Prof. Amir Tomer

Complexity

- How many states can the window be in?

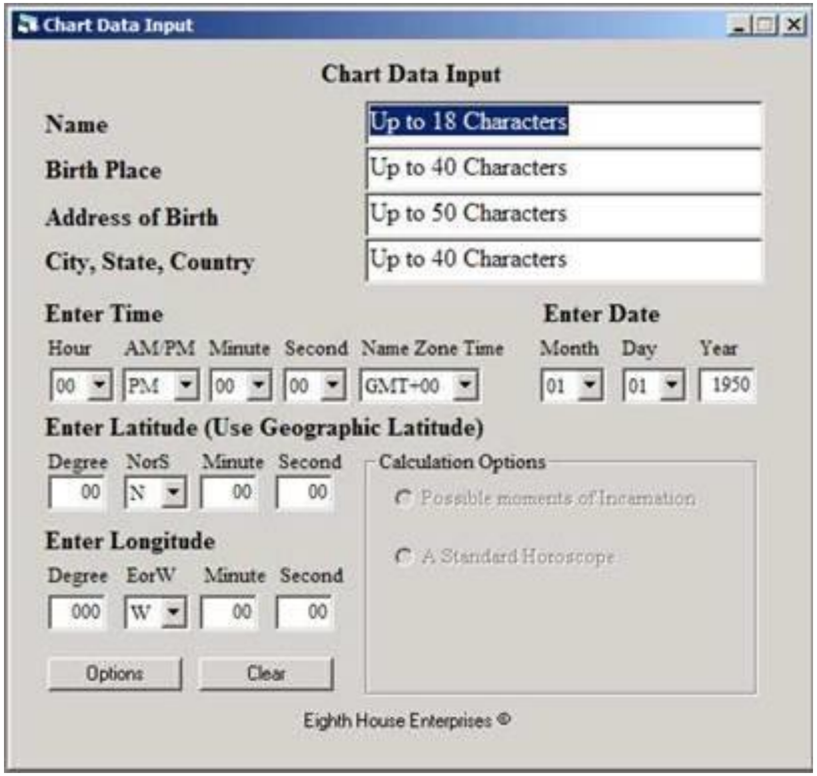


Chart Data Input

Chart Data Input

Name: Up to 18 Characters
Birth Place: Up to 40 Characters
Address of Birth: Up to 50 Characters
City, State, Country: Up to 40 Characters

Enter Time
Hour: 00
AM/PM: PM
Minute: 00
Second: 00
Name Zone Time: GMT+00

Enter Date
Month: 01
Day: 01
Year: 1950

Enter Latitude (Use Geographic Latitude)
Degree: 00
NorS: N
Minute: 00
Second: 00

Calculation Options
○ Possible moments of Incarnation
○ A Standard Horoscope

Enter Longitude
Degree: 000
EorW: W
Minute: 00
Second: 00

Eighth House Enterprises ©

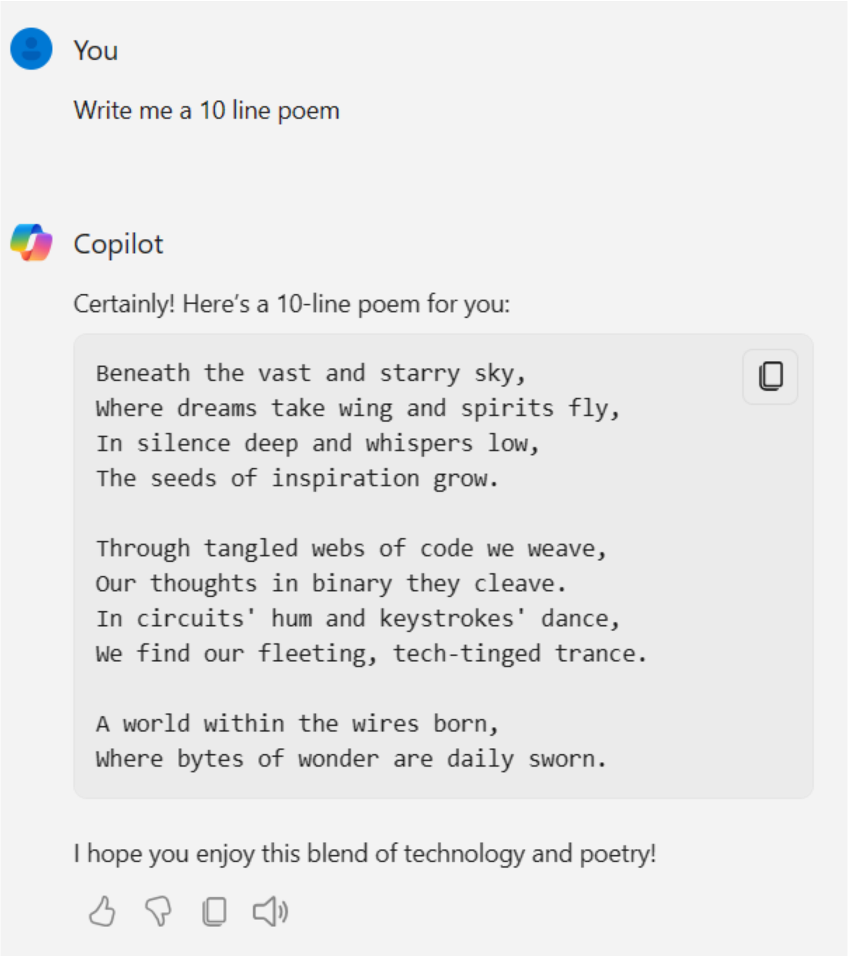
$$\left.\begin{array}{l}\sim 30^{18}\\ \sim 30^{40}\\ \sim 30^{50}\\ \sim 30^{40}\end{array}\right\}\ \sim 10^{255}$$

$$\left.\begin{array}{l}= 12 \times 2 \times 60 \times 60 \times 24 \times\\ \quad 12 \times 31 \times 100 = 77{,}137{,}920{,}000\\ = 360 \times 2 \times 60 \times 60 = 2{,}592{,}000\\ = 360 \times 2 \times 60 \times 60 = 2{,}592{,}000\end{array}\right\}\ \sim 10^{21}$$

- Can you analyze all the states?
- Can you check every possible state of the program?

MSExcel 2007: 850 * 77.1 = ?

Complexity – Non-Deterministic Software



You

Write me a 10 line poem

Copilot

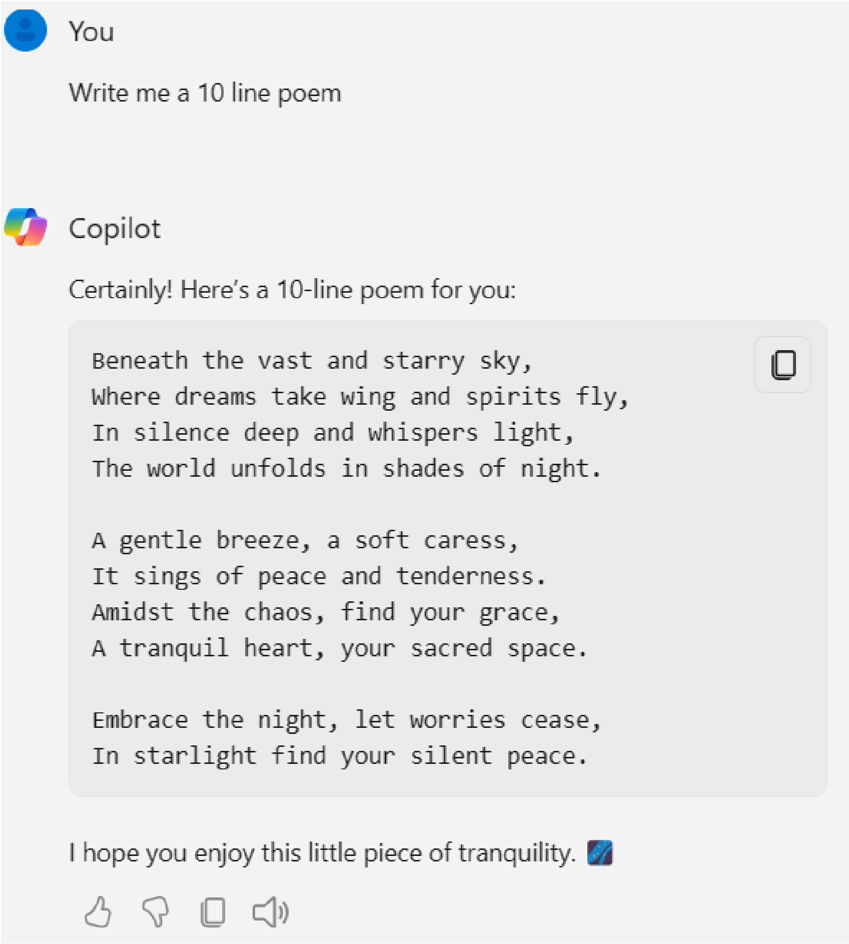
Certainly! Here's a 10-line poem for you:

Beneath the vast and starry sky,
Where dreams take wing and spirits fly,
In silence deep and whispers low,
The seeds of inspiration grow.

Through tangled webs of code we weave,
Our thoughts in binary they cleave.
In circuits' hum and keystrokes' dance,
We find our fleeting, tech-tinged trance.

A world within the wires born,
Where bytes of wonder are daily sworn.

I hope you enjoy this blend of technology and poetry!



You

Write me a 10 line poem

Copilot

Certainly! Here's a 10-line poem for you:

Beneath the vast and starry sky,
Where dreams take wing and spirits fly,
In silence deep and whispers light,
The world unfolds in shades of night.

A gentle breeze, a soft caress,
It sings of peace and tenderness.
Amidst the chaos, find your grace,
A tranquil heart, your sacred space.

Embrace the night, let worries cease,
In starlight find your silent peace.

I hope you enjoy this little piece of tranquility. 🌌

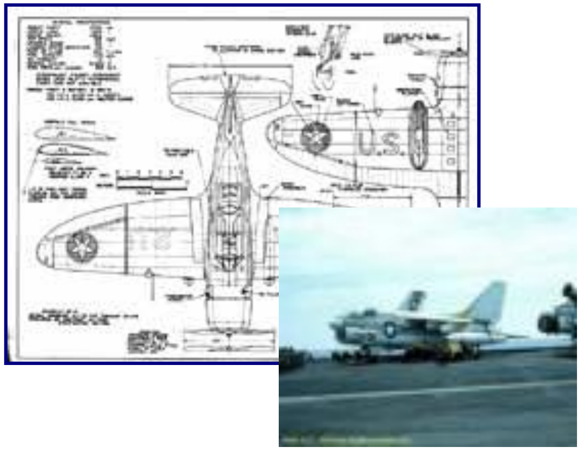
Conformity

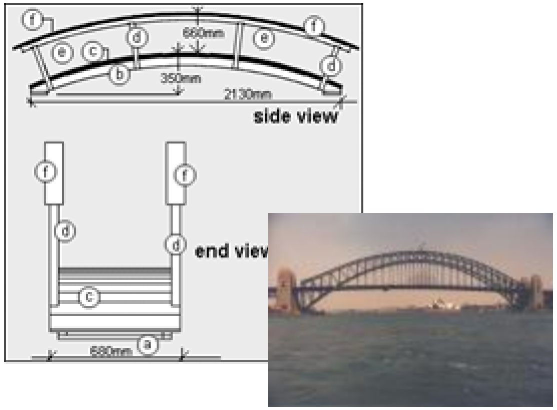
- Add additional abilities to hardware (mechanical, electronic)
  - Overcome physical limitations
  - Increase accuracy, speed, etc.
  - Add or change properties
  - Fix exceptions after the fact
- Software is delivered at the end of the development process
  - But you must take the software’s properties into consideration from the very beginning.

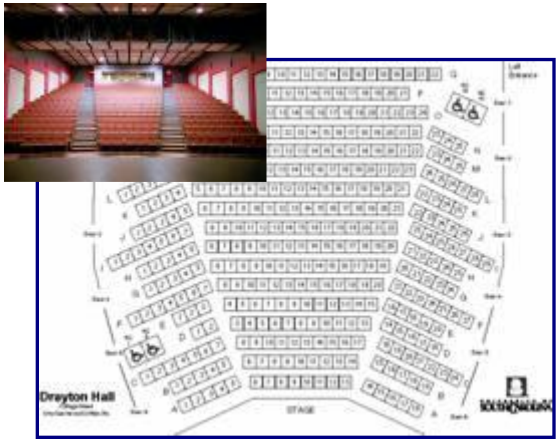
Invisibility

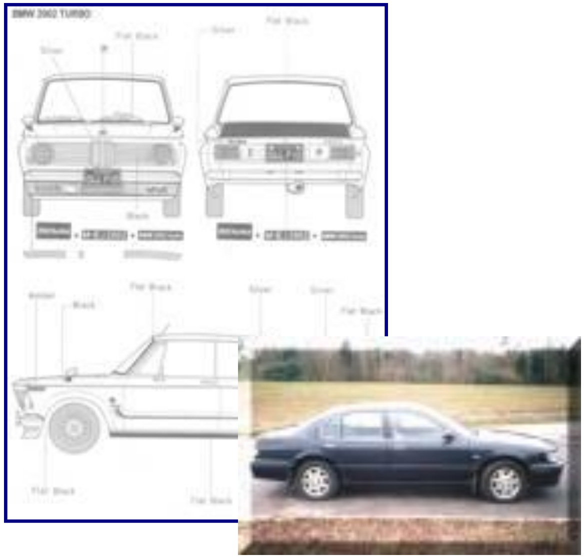
- Can you describe software using a physical model?

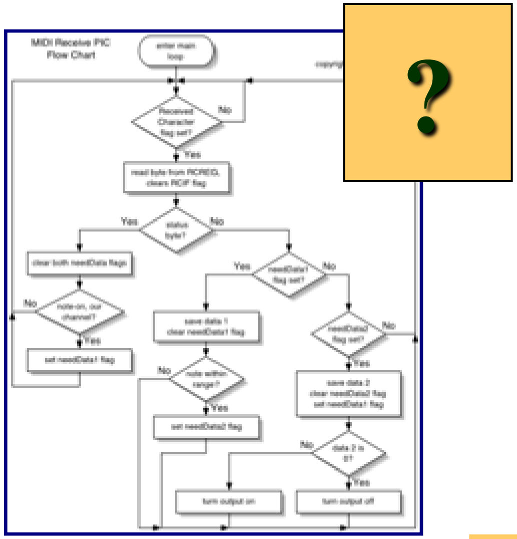
Cupboard
Bath & Shower
Kitchen
Table & Chairs
Sofa Beds
Wardrobe
Double Bed
Balcony



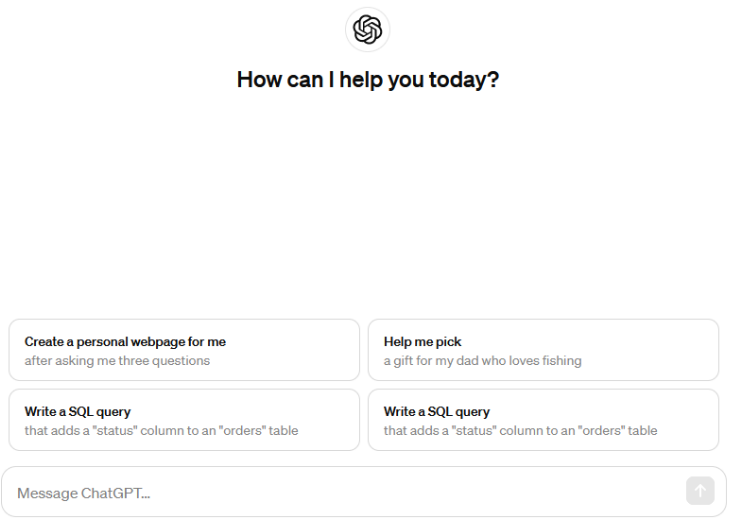







MIDI Receive PIC Flow Chart
enter main loop
Received Character flag set?
read byte from RCREG, clears RCIF flag
status byte?
clear both needData flags
needData1 flag set?
note-on, our channel?
save data 1 clear needData1 flag
needData2 flag set?
set needData1 flag
note within range?
save data 2 clear needData2 flag set needData1 flag
set needData2 flag
data 2 is 0?
turn output on
turn output off
Yes
No
?

Invisibility – How do LLMs work?



How can I help you today?

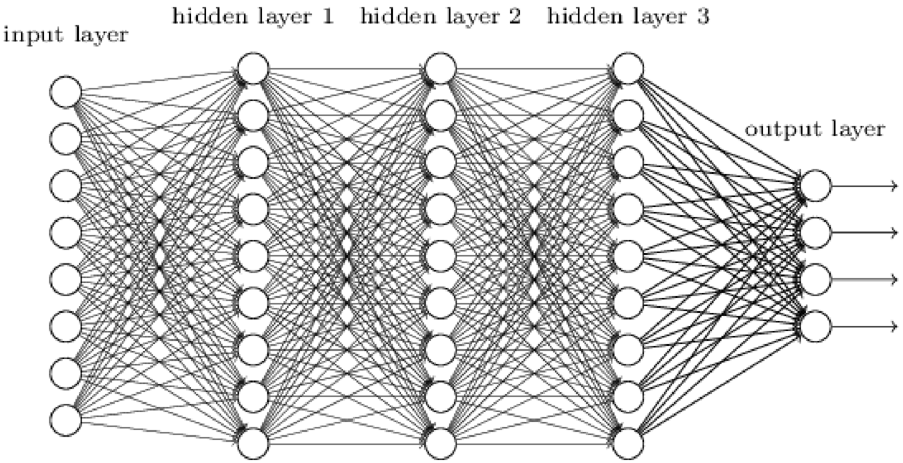
Create a personal webpage for me
after asking me three questions

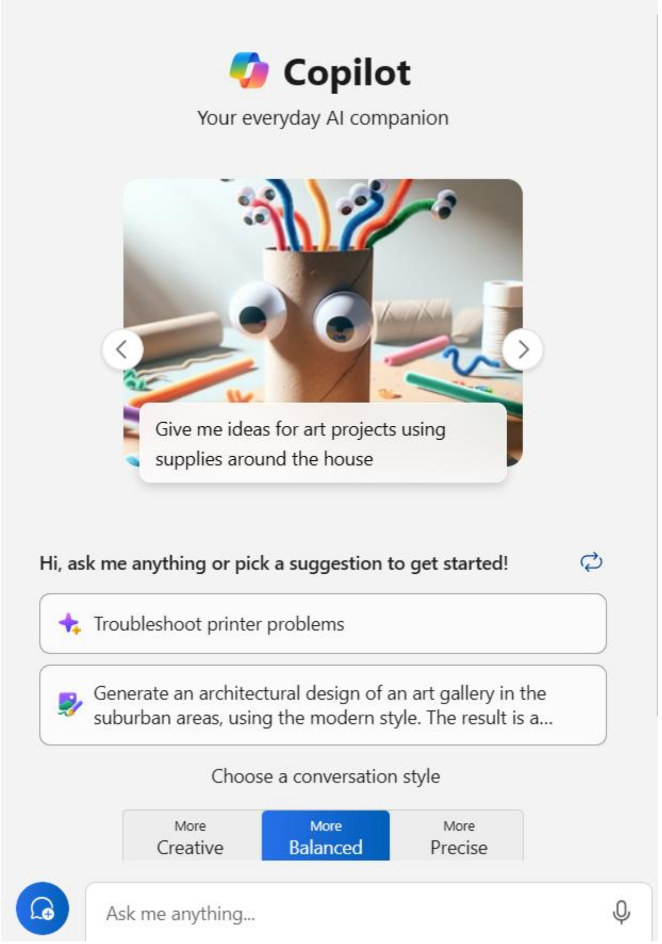
Help me pick
a gift for my dad who loves fishing

Write a SQL query
that adds a "status" column to an "orders" table

Write a SQL query
that adds a "status" column to an "orders" table

Message ChatGPT...





Copilot

Your everyday AI companion

Give me ideas for art projects using supplies around the house

Hi, ask me anything or pick a suggestion to get started!

Troubleshoot printer problems

Generate an architectural design of an art gallery in the suburban areas, using the modern style. The result is a...

Choose a conversation style

Ask me anything...

Changeability and Maintainability

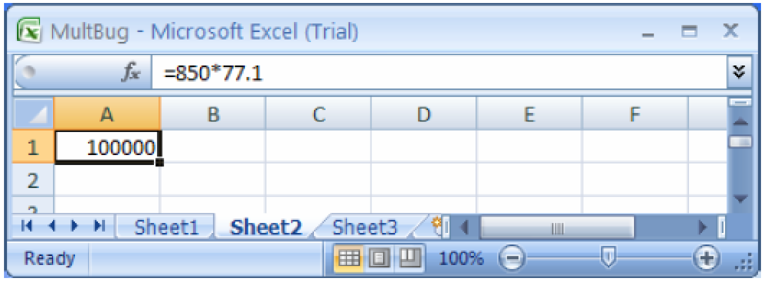
- Software is a “changeable” product
  - Software doesn’t need raw materials
  - Development tools are cheap (sort of) and available
  - There is no “production” process
  - Changing the model is quick and easy
- Therefore…
  - Software products can be expected to undergo regular changes and updates.

Maintenance (i.e. fixes, updates, adaptation) takes up the majority of a software’s lifetime.

"Random" Singular Bugs

- The embarrassing "bug" in Excel 2007

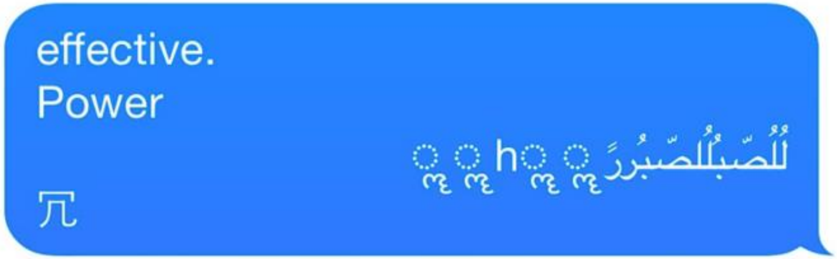
  850x77.1 = 65,535 = 1111111111111111_b



=850*77.1

100000

- The message that crashed iOS (in 2017, then another one in 2020)

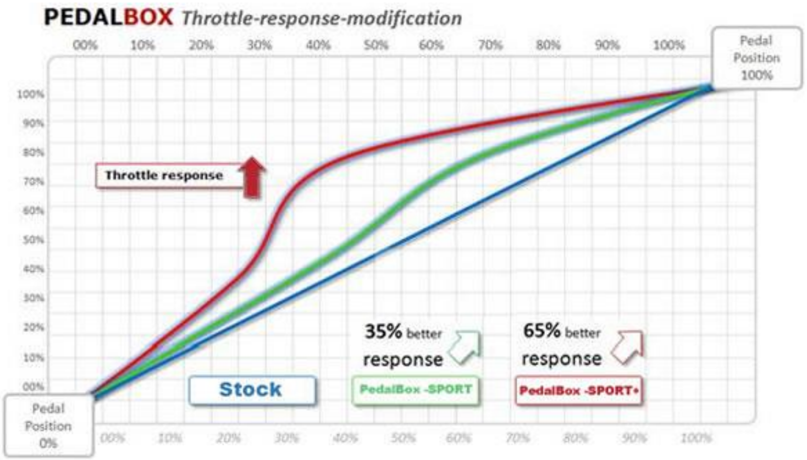


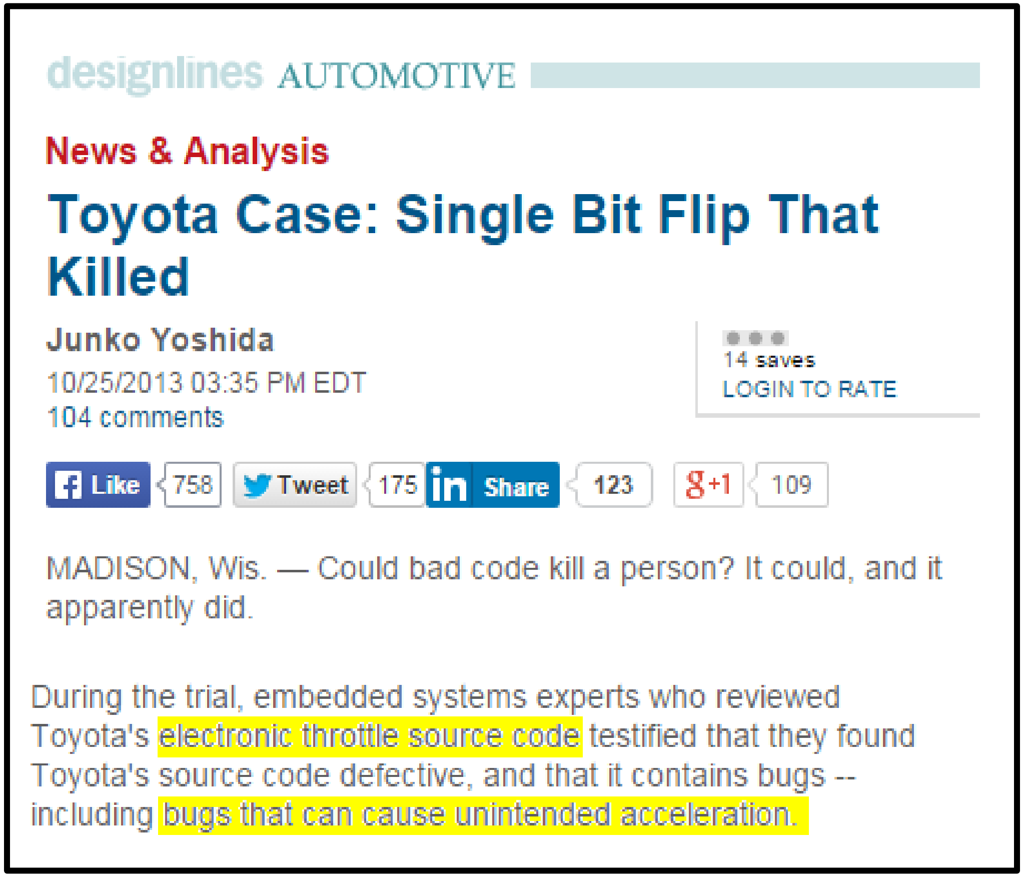
effective.
Power
لُلُصّبُلُلصّبُررً ॣ ॣh ॣ ॣ
冗

- The input that crashed Skype

http://:

Toyota unintended acceleration





News & Analysis

Toyota Case: Single Bit Flip That Killed

Junko Yoshida

10/25/2013 03:35 PM EDT

104 comments

MADISON, Wis. — Could bad code kill a person? It could, and it apparently did.

During the trial, embedded systems experts who reviewed Toyota's electronic throttle source code testified that they found Toyota's source code defective, and that it contains bugs -- including bugs that can cause unintended acceleration.

http://www.eetimes.com

When garbage is invisible

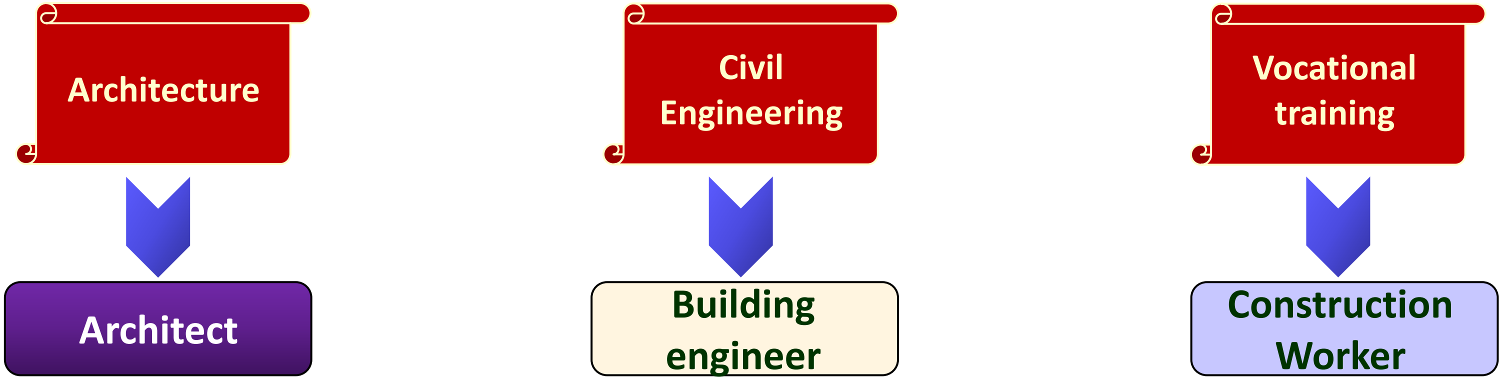
If software was made of wood…

… at the end of the project, everything would be covered in a mountain of sawdust.

© Prof. Amir Tomer

Software: Academic training for careers

Construction



Architecture

Architect

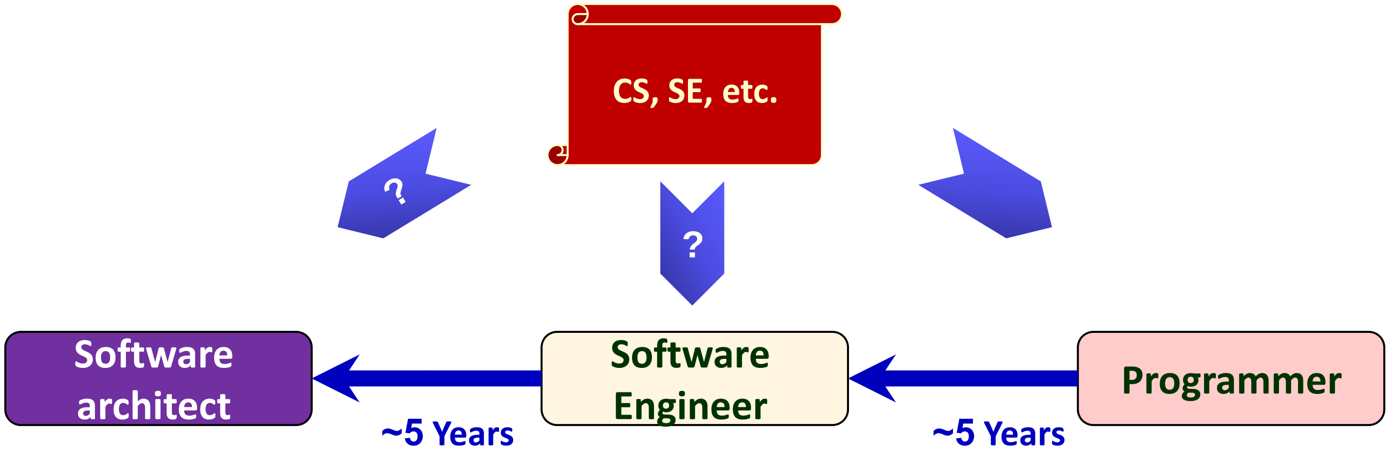
Civil Engineering

Building engineer

Vocational training

Construction Worker

Software



CS, SE, etc.

?

?

Software architect

Software Engineer

Programmer

~5 Years

~5 Years

So Far

- Introduction to Software Intensive Systems
- Software from a Systems Perspective

© Prof. Amir Tomer

Software Product = the final result of the software development process

- Software Product: The set of computer programs, procedures, and possibly associated documentation and data [various software standards]
- The software product is delivered in one of 3 ways

Standalone application (app)

- Software is provided separately (disk, digital download)
- Meant to be installed on a standard platform (computer, smartphone) owned by the user that includes an OS and basic programs
  - Games, bank account management software

Embedded software

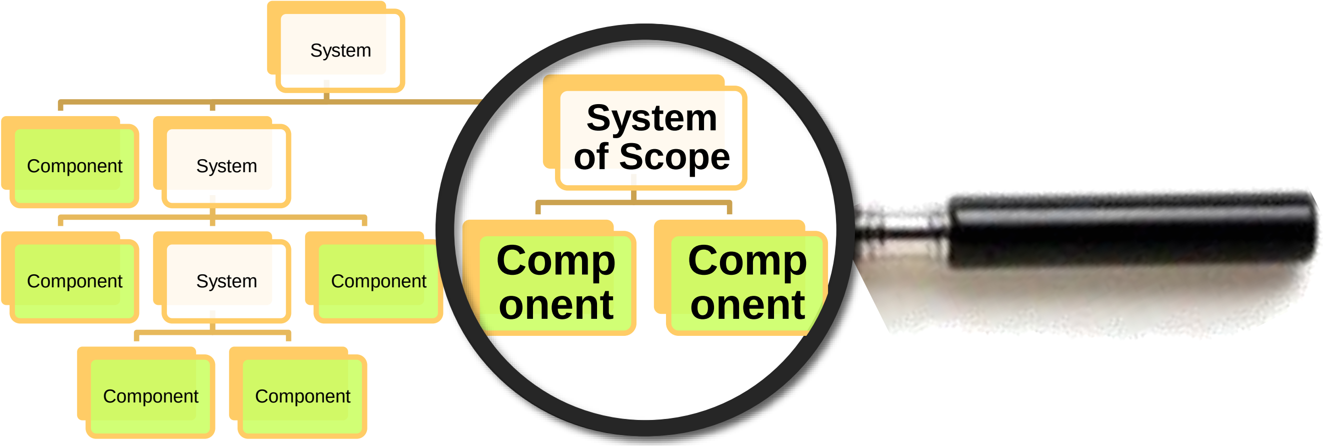
- Software is provided inside hardware (specialized)
- With an OS and underlying software as a single product
  - Car, medical device

Software as a Service

- Software isn't transferred to the customer or owned by her
- Customer just receives the required services
  - Google search, Office 365

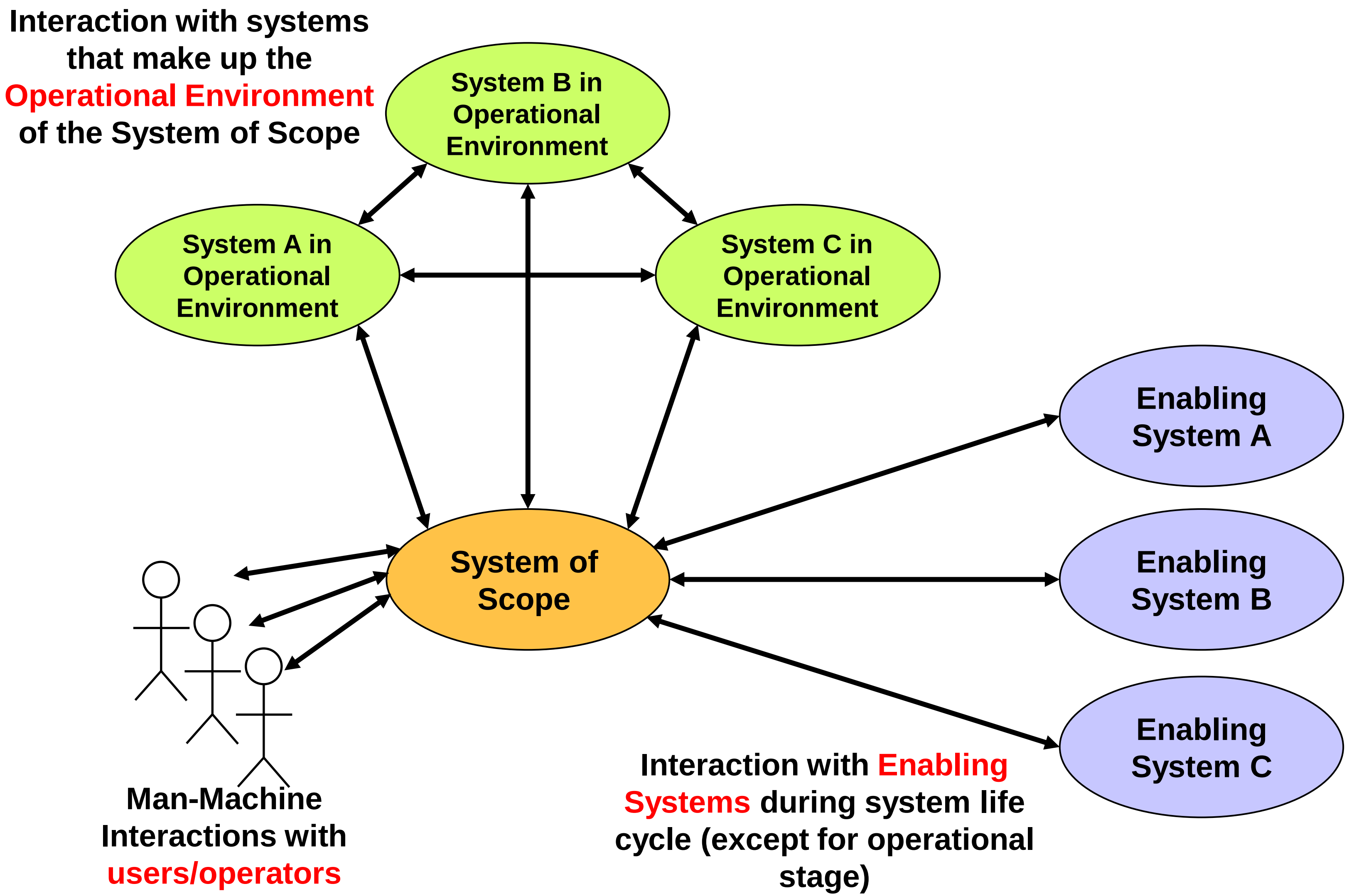
Software from a Systems Perspective

- A system (definition)
  - A combination of organized elements (components) that interact to achieve one or more declared goals [ISO/IEC/IEEE 15288]
- A system in a hierarchical/recursive structure
  - System components can be systems themselves



- System of Scope
  - The system the development process is working on, including all activities and products
- Software Intensive System
  - System in which software plays an important part in terms of functionality, development cost, development risks, or development time

System of Scope and its interfaces with the environment [ISO/IEC/IEEE 15288]



Interaction with systems that make up the Operational Environment of the System of Scope

System B in Operational Environment
System A in Operational Environment
System C in Operational Environment
Enabling System A
Enabling System B
Enabling System C
System of Scope

Man-Machine Interactions with users/operators

Interaction with Enabling Systems during system life cycle (except for operational stage)

System Attributes

- Systems (at any level) are defined by the following attributes

Environment	Goal(s)	Ingredients	Organization (Structure)	Interaction (Behavior)
• External factors (human, non-human) that the system works with	• Solution to the problem/need (customer, stakeholder)	• Parts of the solution	• Connections between ingredients • Connections between the system (and its parts) and the environment	• Interaction between elements or between elements and the environment to achieve the goal

In Class Practice

- Moodle is made up of the following elements
  - User interface
  - Teacher interface – edit content
  - Student interface – view content
  - Database of data and content
- Moodle as a System
  - What is its goal?
  - How are the components organized (organizational figure, structure)?
  - How do the parts interact?
  - To what other systems is it connected? (Environment)
  - What larger system is it part of?

Padlet link on Moodle

Typical levels of scope in development of SIS

Organization/Business Level

The operational framework in which the system will be installed and serve

Computer System Level

HW/SW system that serves the organization for some purpose

Software Component Level

Software is installed on the HW and users/other systems work with it

Software Element Level

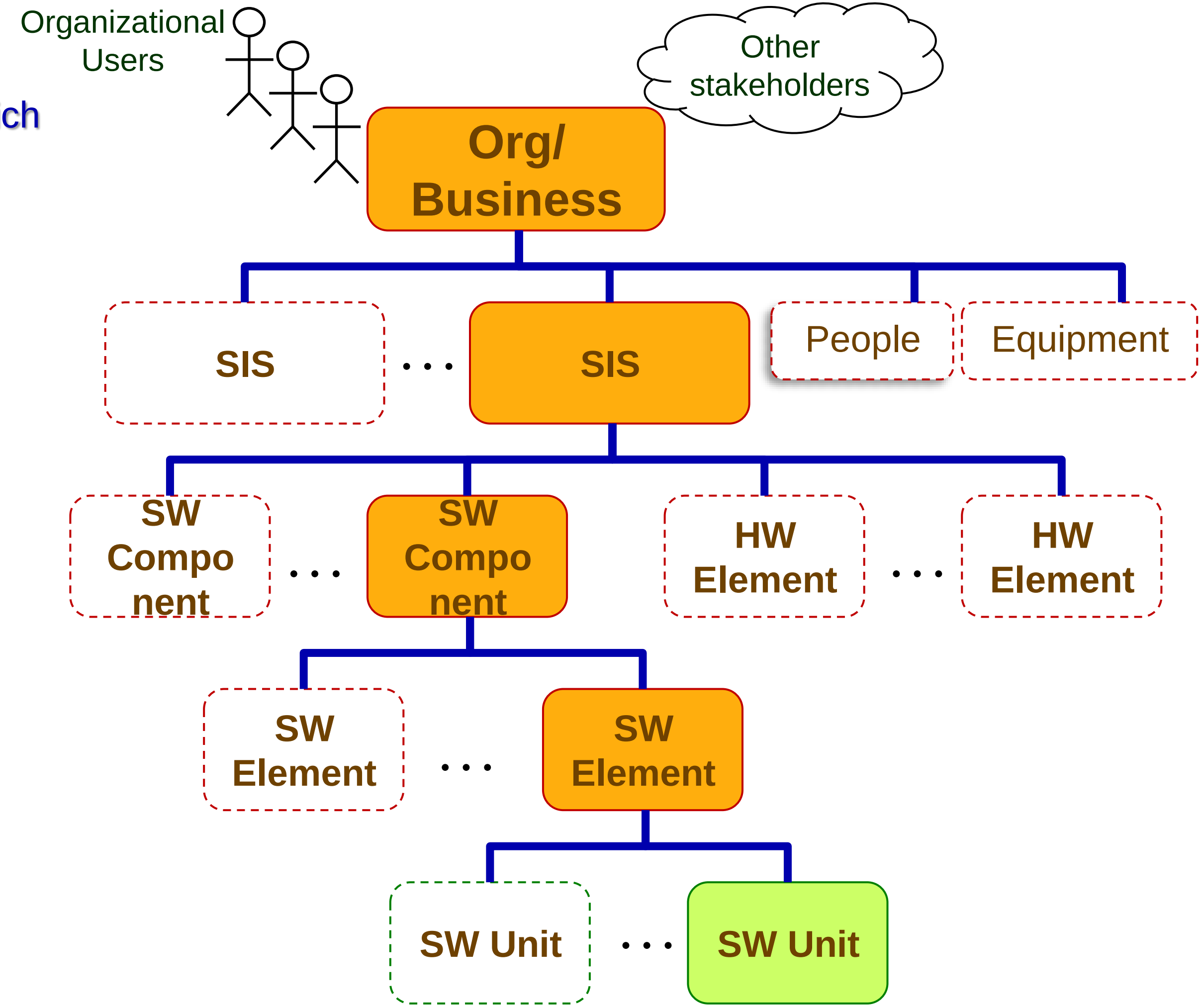
Elements the app is built from

Build elements: e.g. libraries

Runtime elements: e.g. DLLs

Software Unit Level

Code modules (e.g. classes) which make up elements



© Prof. Amir Tomer

Levels of scope: Medical system

Organization/Business Level

Hospital the system is installed in

Medical goal: diagnosis/treatment
Work logic: Medical procedure

Computer System Level

HW/SW system that helps hospital offer services

Operational task: CT imaging
Work logic: Take image, process, display

Software Component Level

Specialized software, task specific

Technical task: produce a series of images
Work logic: Operate the machine

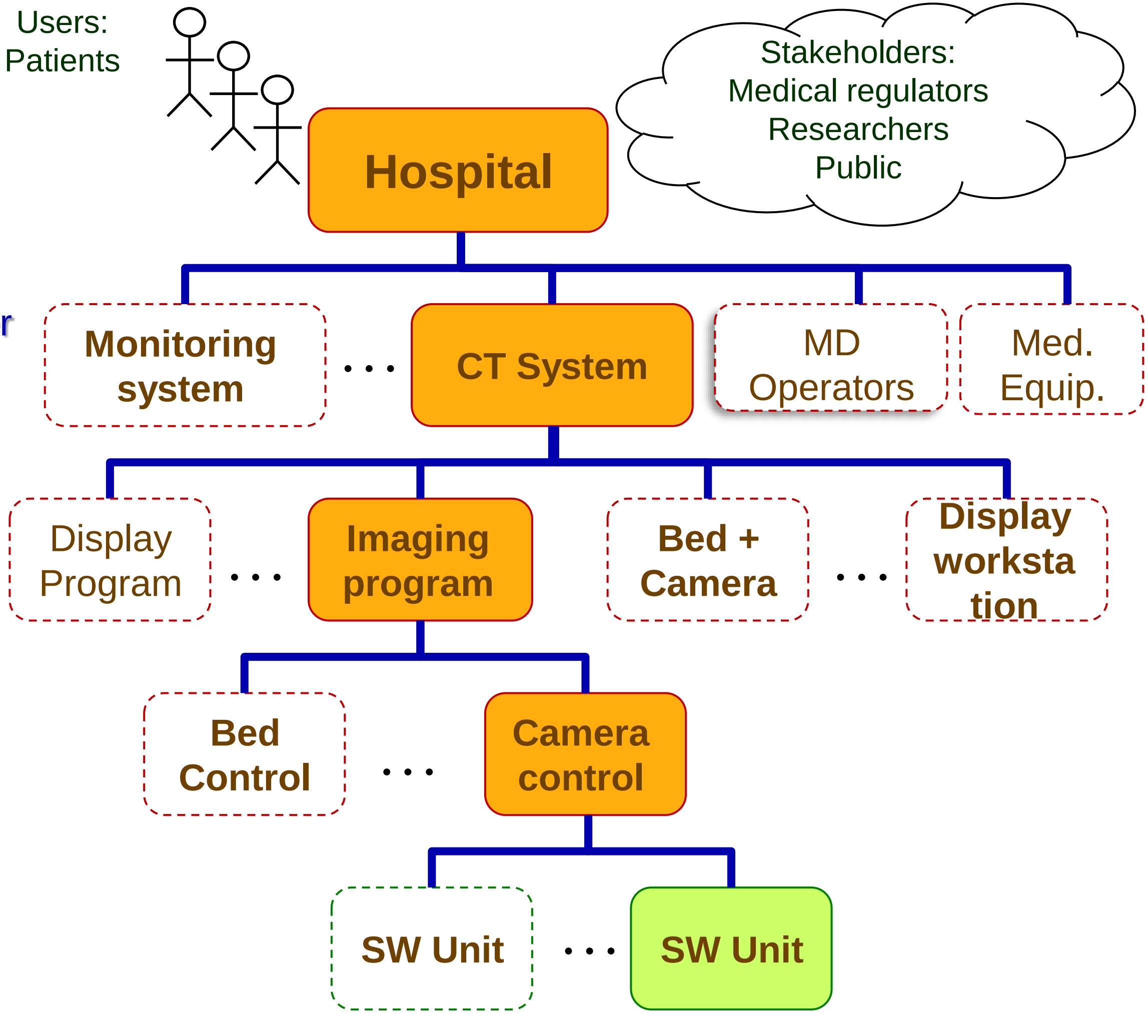
Software Element Level

Parts the software is built from

Build elements: image processing libraries
Runtime elements: Control tracking libraries

Software Unit Level

Code modules (e.g. classes) which make up elements



Users:
Patients

Stakeholders:
Medical regulators
Researchers
Public

Hospital

Monitoring system . . . **CT System** MD Operators Med. Equip.

Display Program . . . **Imaging program** **Bed + Camera** . . . **Display workstation**

Bed Control . . . **Camera control**

SW Unit . . . **SW Unit**

Levels of scope: In-car navigation (exercise)

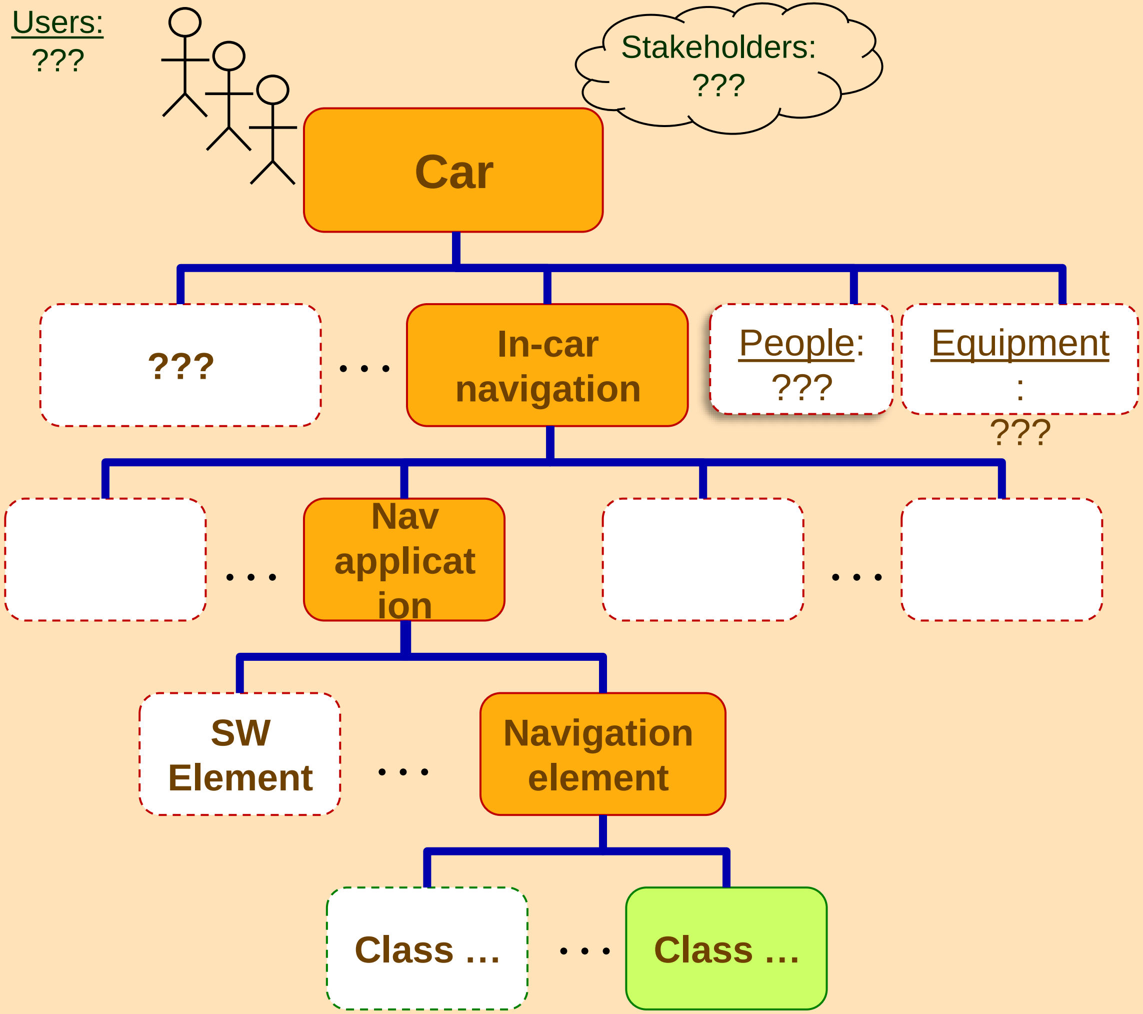
Organization/Business Level
- ???
- Goal: ???
- Work logic: ???

Computer System Level
- ???
- Operational goal: ???
- Work logic: ???

Software Component Level
- ???
- Technical goal: ???
- Work logic: ???

Software Element Level
- Elements the app is built from
- Build elements: ???
- Runtime elements: ???

Software Unit Level
- Code modules (e.g. classes) which make up elements



Fill in ???

Conclusion

- Introduction to Software Intensive Systems
- Software from a Systems Perspective

Personal Introduction

Instructor: **Michael J. May**

Ph.D. in Computer Science University of Pennsylvania, Philadelphia

Courses I teach (2nd year):

- Software Intensive Systems Engineering הנדסת מערכות עתירות תוכנה

Third year:

- Operating Systems מערכות הפעלה
- Introduction to Computer Networks מבוא לרשתות מחשבים

Fourth year:

- Communication and Information Security אבטחת תקשורת ומידע
- Distributed Systems מערכות מבוזרות
- Advanced Computer Networks רשתות מחשבים מתקדמות

My Research

Digitizing Cemeteries

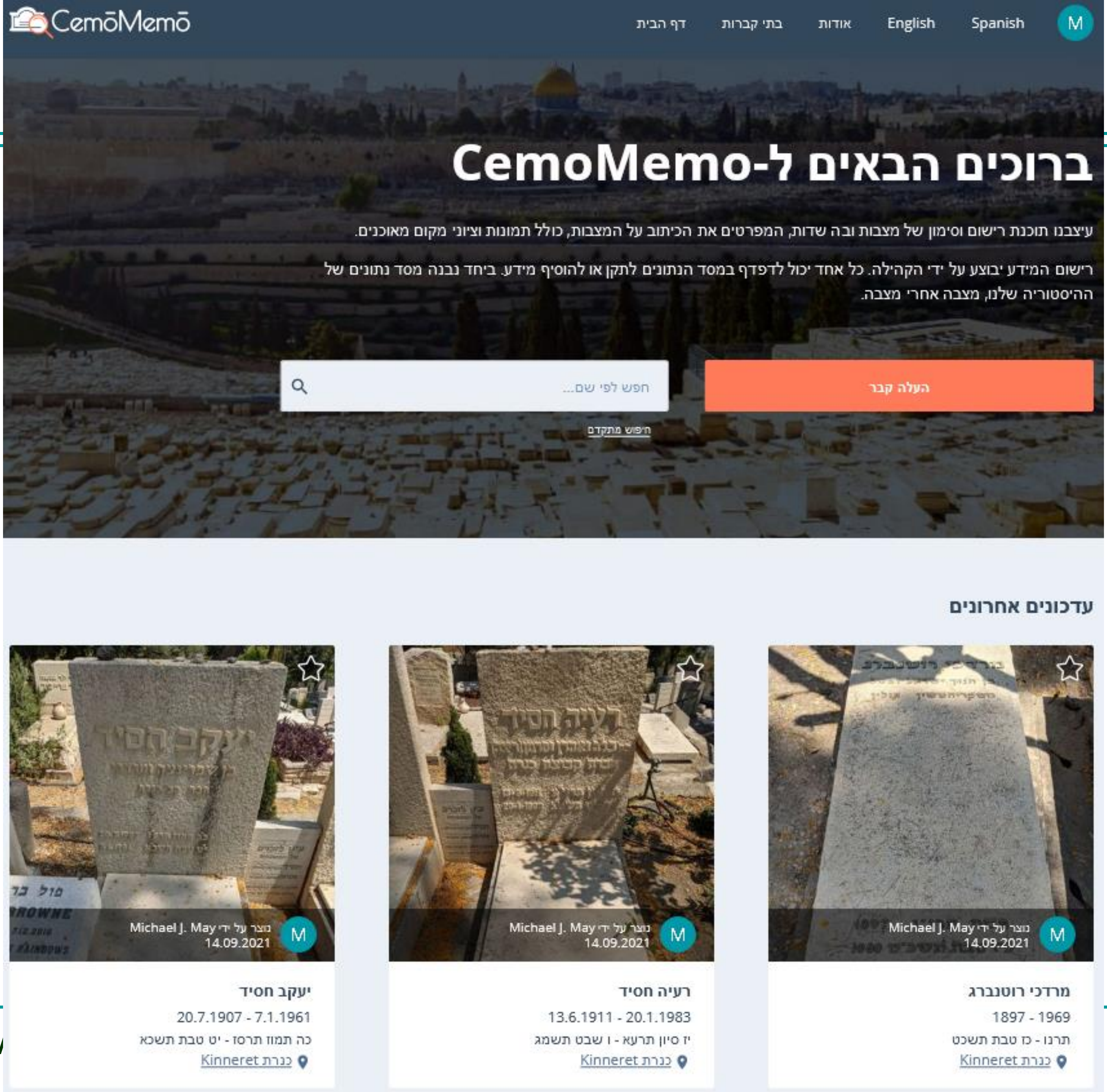
- Optical character recognition
- Machine learning
- Historical analysis

Secure communications

- Better communication techniques
- Formal analysis

Intelligent file processing

- File lifecycle
- Content aware differencing



ברוכים הבאים ל-CemoMemo

עיצבנו תוכנת רישום וסימון של מצבות ובה שדות, המפרטים את הכיתוב על המצבות, כולל תמונות וציוני מקום מאוכנים.

רישום המידע יבוצע על ידי הקהילה. כל אחד יכול לדפדף במסד הנתונים לתקן או להוסיף מידע. ביחד נבנה מסד נתונים של ההיסטוריה שלנו, מצבה אחרי מצבה.

עדכונים אחרונים

נוצר על ידי Michael J. May
14.09.2021

מרדכי רוטנברג
1897 - 1969
תרנו - כז טבת תשכט
כנרת Kinneret

נוצר על ידי Michael J. May
14.09.2021

רעיה חסיד
13.6.1911 - 20.1.1983
יז סיון תרעא - ו שבט תשמג
כנרת Kinneret

נוצר על ידי Michael J. May
14.09.2021

יעקב חסיד
20.7.1907 - 7.1.1961
כה תמוז תרסז - יט טבת תשכא
כנרת Kinneret

About My Courses

https://www2.kinneret.ac.il/mjmay

Every course has a web page and a Moodle page

- On the course page:
  - Syllabus, course location and time
  - Lecture schedule
  - Lecture slides (PDF)
  - Targil summaries (if I run them)
  - Reminders about assignment due dates (normally)
  - Code, programming data

- On Moodle:
  - Assignment solutions and grades
  - Previous tests
  - Other course announcements (changes, extensions, updates, etc).
  - Targil materials (if I don’t run them)

In Class

- Come to class on time (after 10 minutes, the door is locked)
- Don’t talk during class or disturb others – raise your hand.
- Read what I ask you to
- Perform and submit assignments on time
  - Late policy (same for all courses)
  - Don’t ask for personal extensions
  - Miluim, births, etc. as per the college policy
- Come to office hours if you need extra help
- If you have a complaint or request – try asking me first
- Don’t copy or turn in duplicate assignments
  - Group work policy per course
  - “We worked on it together” is not acceptable
  - Don’t copy from the internet
- Correct me (respectfully) if I make mistakes, say something silly, etc.