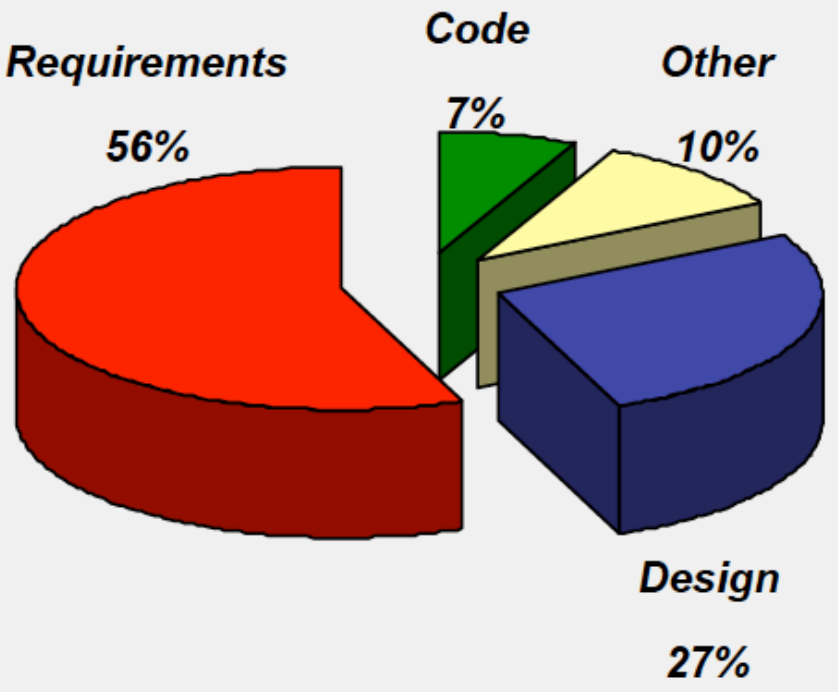
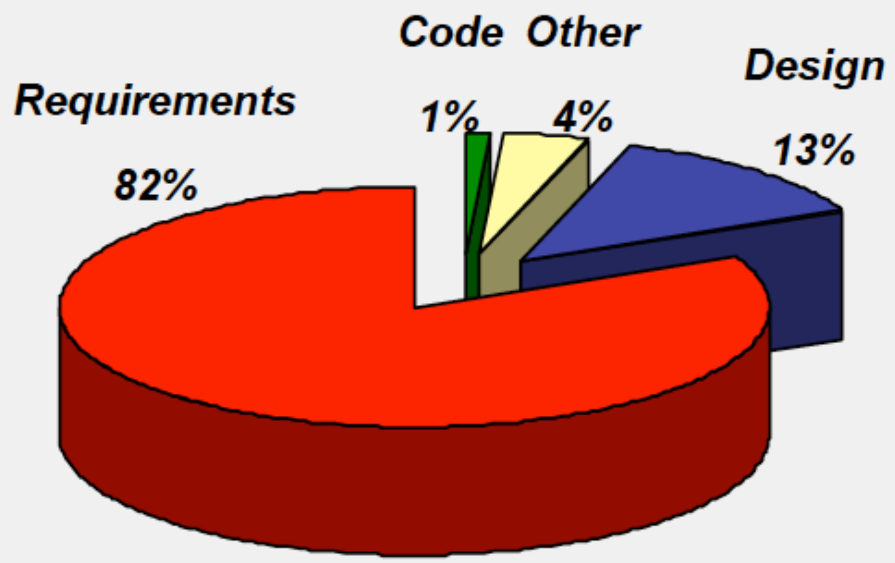
What is Design?

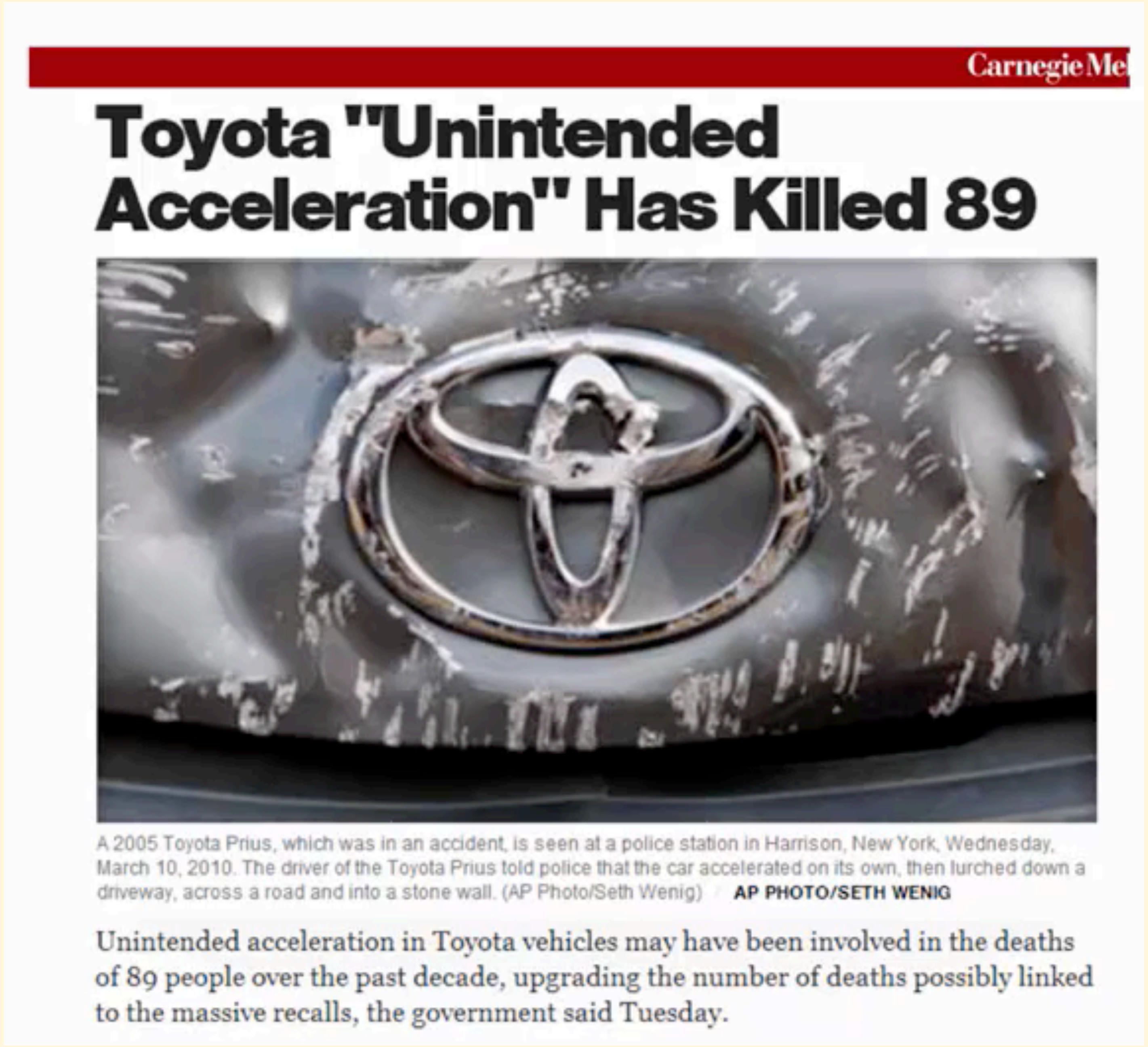
- Using Eiffel the language, method and tool
  - as a <u>design</u> tool
- In an engineering software design course
- Requiring <u>rigorous</u> design for mission critical systems

Distribution of Defects



Distribution of Effort to Fix Defects



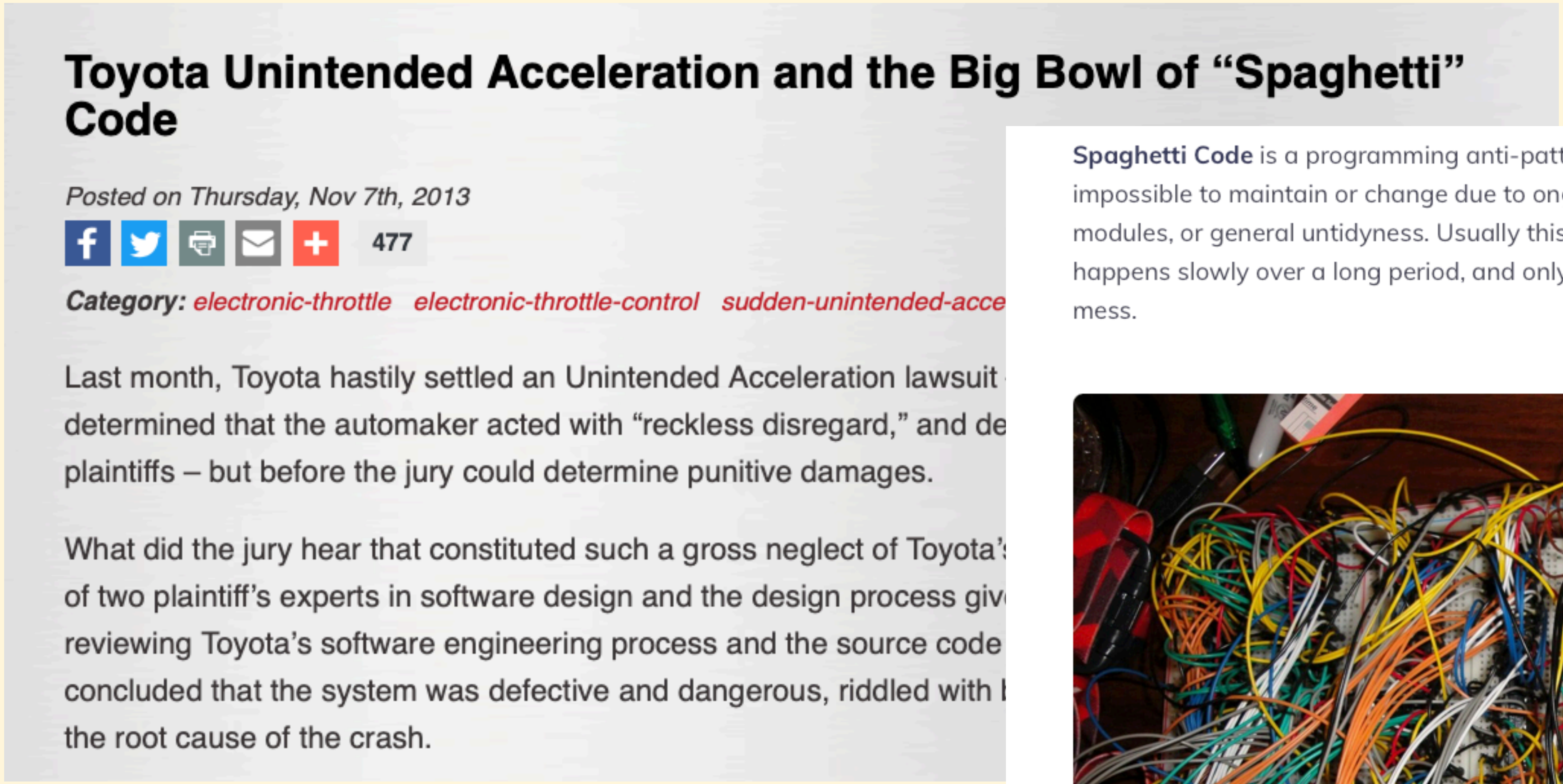


Carnegie Mel

Toyota "Unintended Acceleration" Has Killed 89

A 2005 Toyota Prius, which was in an accident, is seen at a police station in Harrison, New York, Wednesday, March 10, 2010. The driver of the Toyota Prius told police that the car accelerated on its own, then lurched down a driveway, across a road and into a stone wall. (AP Photo/Seth Wenig) / **AP PHOTO/SETH WENIG**

Unintended acceleration in Toyota vehicles may have been involved in the deaths of 89 people over the past decade, upgrading the number of deaths possibly linked to the massive recalls, the government said Tuesday.



Toyota Unintended Acceleration and the Big Bowl of “Spaghetti” Code

Posted on Thursday, Nov 7th, 2013

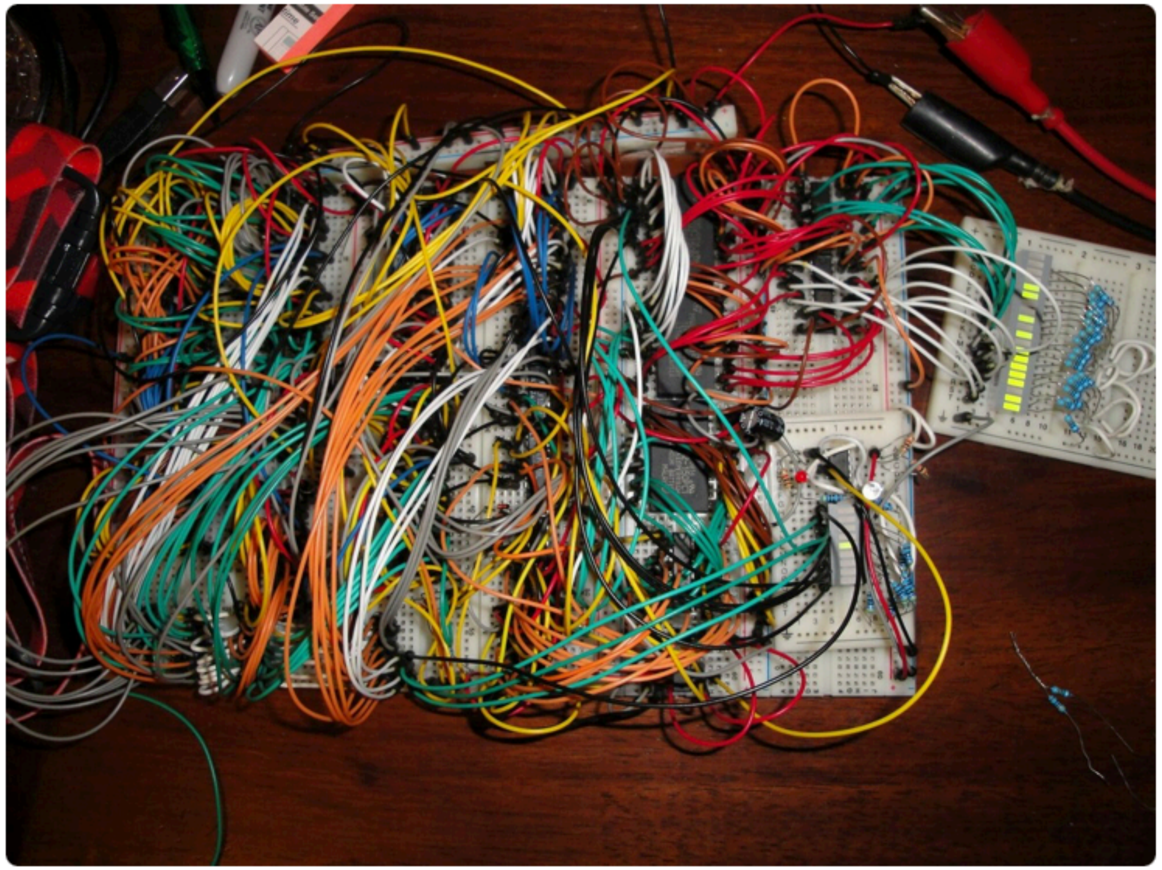
477

Category: *electronic-throttle* *electronic-throttle-control* *sudden-unintended-acce*

Last month, Toyota hastily settled an Unintended Acceleration lawsuit
determined that the automaker acted with “reckless disregard,” and de
plaintiffs – but before the jury could determine punitive damages.

What did the jury hear that constituted such a gross neglect of Toyota’
of two plaintiff’s experts in software design and the design process giv
reviewing Toyota’s software engineering process and the source code
concluded that the system was defective and dangerous, riddled with
the root cause of the crash.

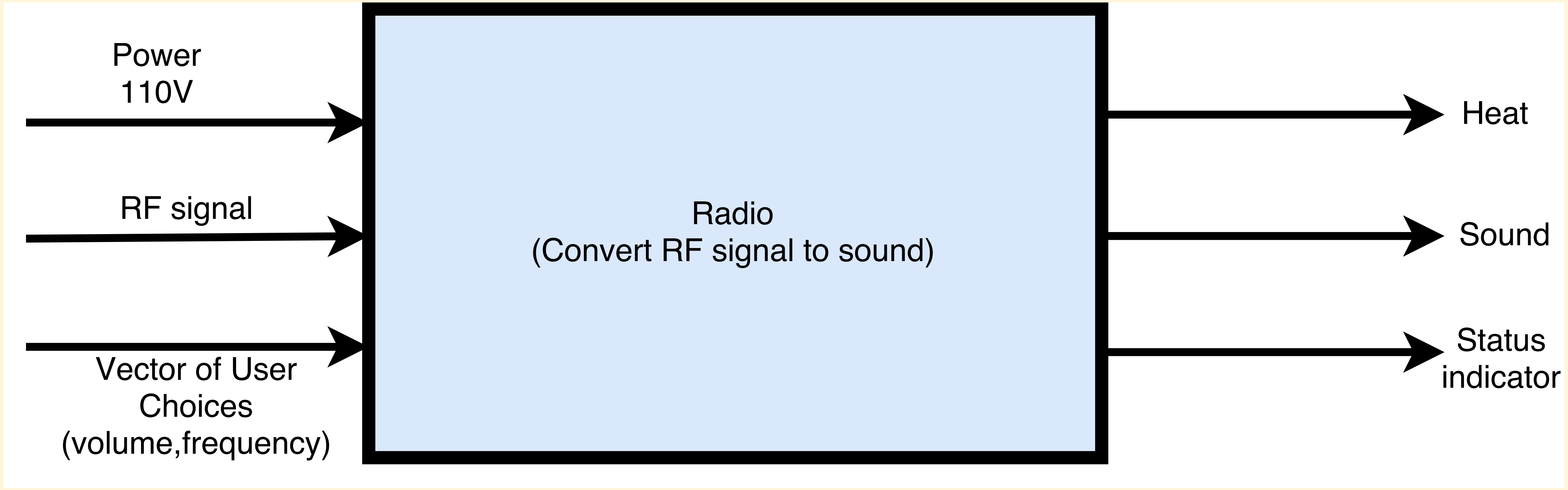
Spaghetti Code is a programming anti-pattern in which code becomes almost impossible to maintain or change due to ongoing changes, interactivity between modules, or general untidyness. Usually this does not happen all at once; rather, it happens slowly over a long period, and only by coming back later do you notice the mess.



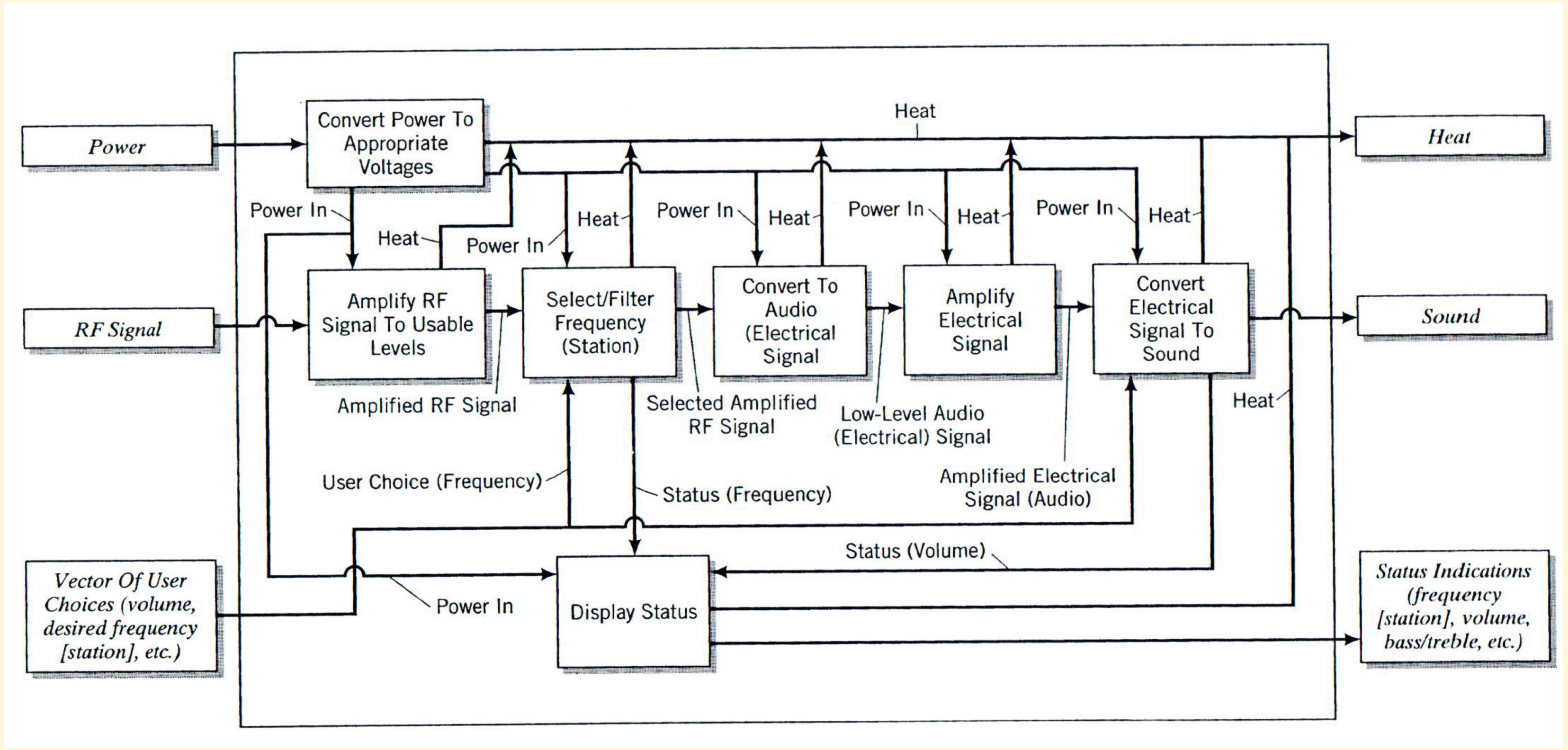
Which one defuses the bomb?

https://exceptionnotfound.net/spaghetti-code-the-daily-software-anti-pattern/

What is Design?



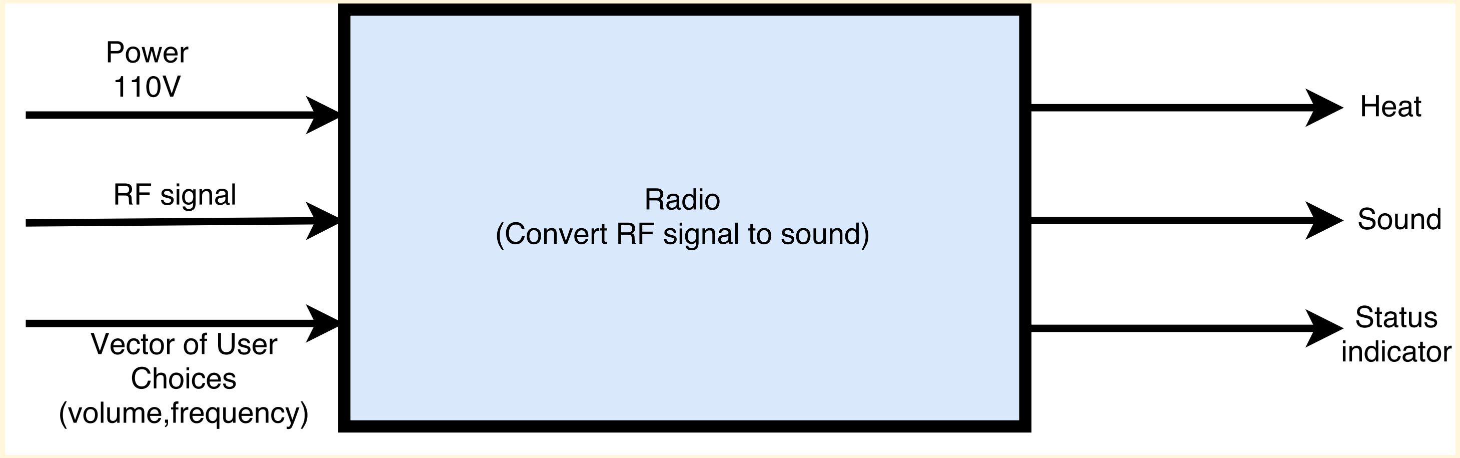
What is Design?



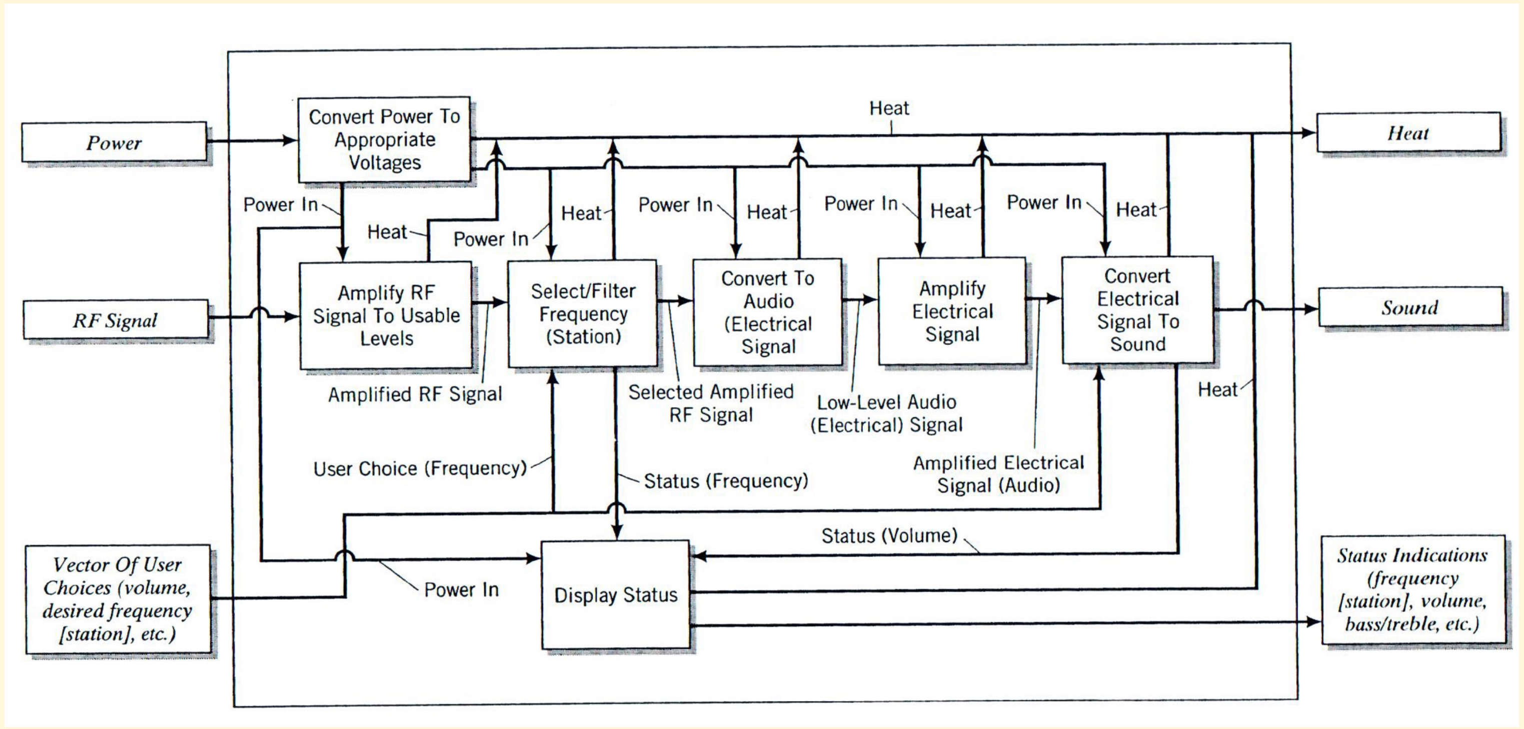
Power
RF Signal
Vector Of User Choices (volume, desired frequency [station], etc.)
Convert Power To Appropriate Voltages
Amplify RF Signal To Usable Levels
Select/Filter Frequency (Station)
Convert To Audio (Electrical Signal
Amplify Electrical Signal
Convert Electrical Signal To Sound
Display Status
Heat
Power In
Amplified RF Signal
Selected Amplified RF Signal
Low-Level Audio (Electrical) Signal
Amplified Electrical Signal (Audio)
User Choice (Frequency)
Status (Frequency)
Status (Volume)
Heat
Sound
Status Indications (frequency [station], volume, bass/treble, etc.)

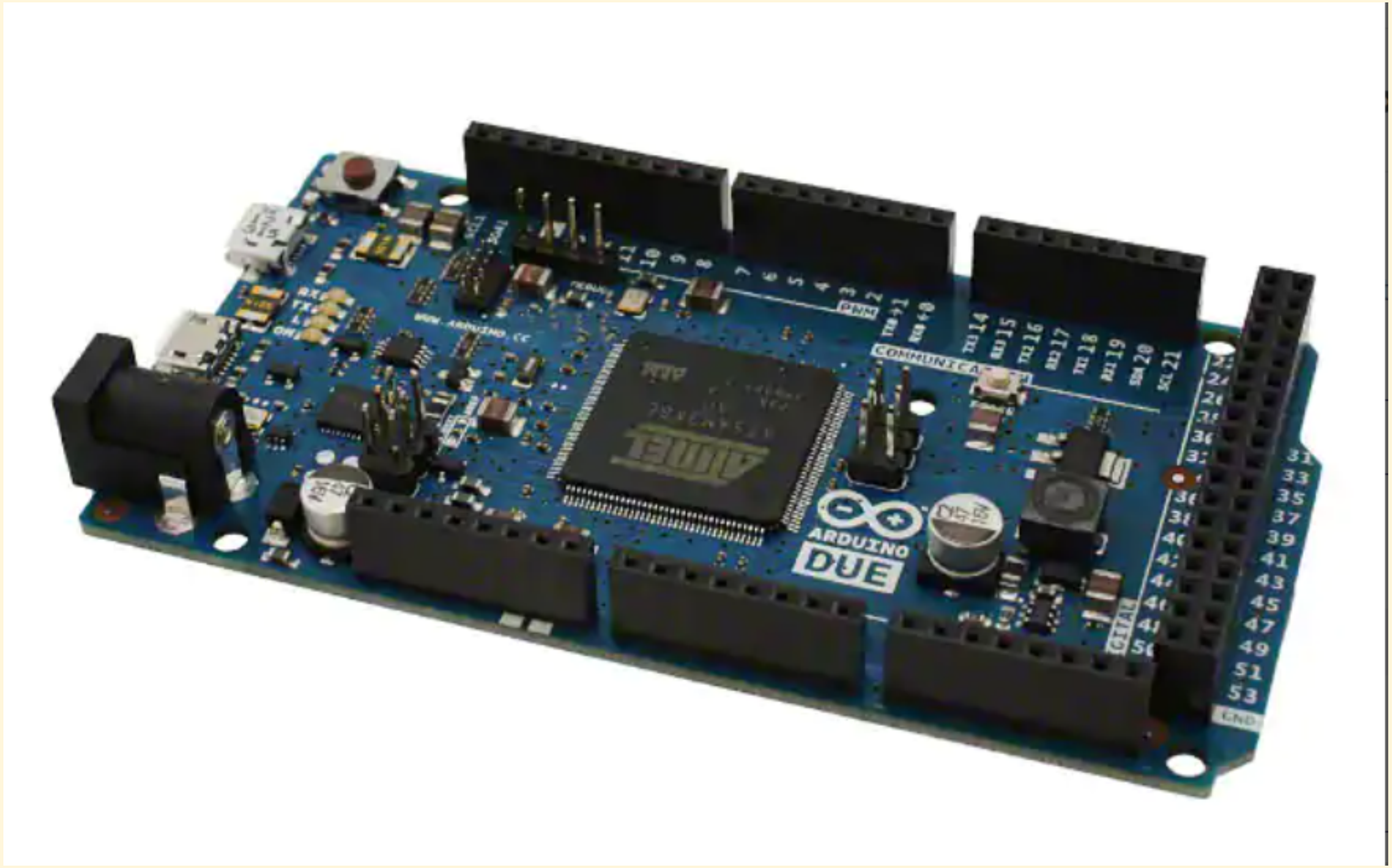
What is Design?

Specification


Power
110V
RF signal
Vector of User Choices (volume,frequency)
Radio
(Convert RF signal to sound)
Heat
Sound
Status indicator

Architecture (Blueprint)


Power
RF Signal
Vector Of User Choices (volume, desired frequency [station], etc.)
Convert Power To Appropriate Voltages
Amplify RF Signal To Usable Levels
Select/Filter Frequency (Station)
Convert To Audio (Electrical Signal
Amplify Electrical Signal
Convert Electrical Signal To Sound
Display Status
Heat
Power In
Amplified RF Signal
Selected Amplified RF Signal
Low-Level Audio (Electrical) Signal
Amplified Electrical Signal (Audio)
User Choice (Frequency)
Status (Frequency)
Status (Volume)
Heat
Sound
Status Indications (frequency [station], volume, bass/treble, etc.)

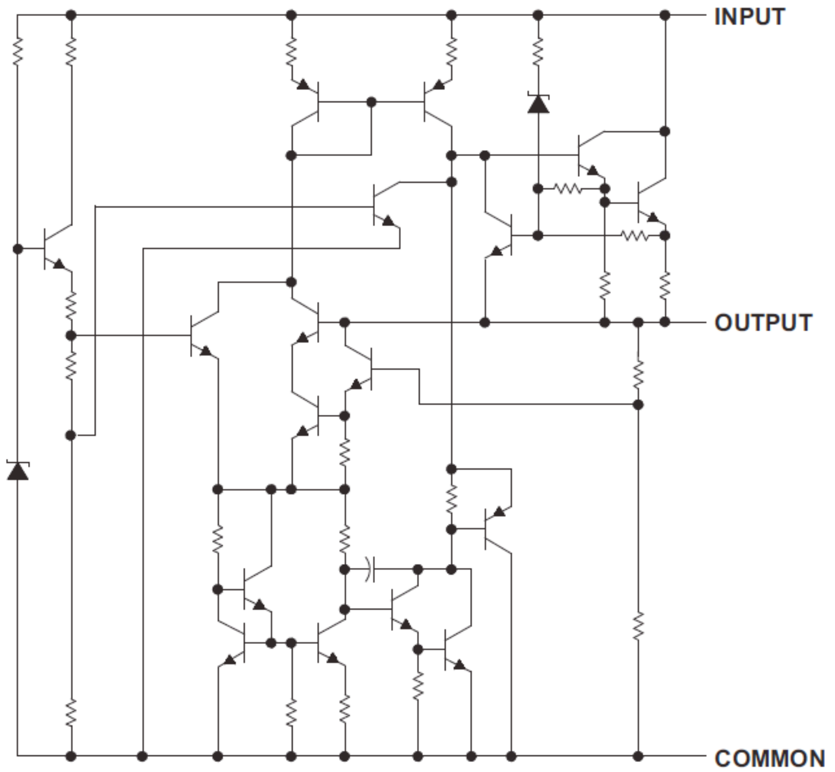

ARDUINO
DUE
COMMUNICATION
PWM
TX3 14
RX3 15
TX2 16
RX2 17
TX1 18
RX1 19
SDA 20
SCL 21
WWW.ARDUINO.CC

8 Detailed Description

8.1 Overview

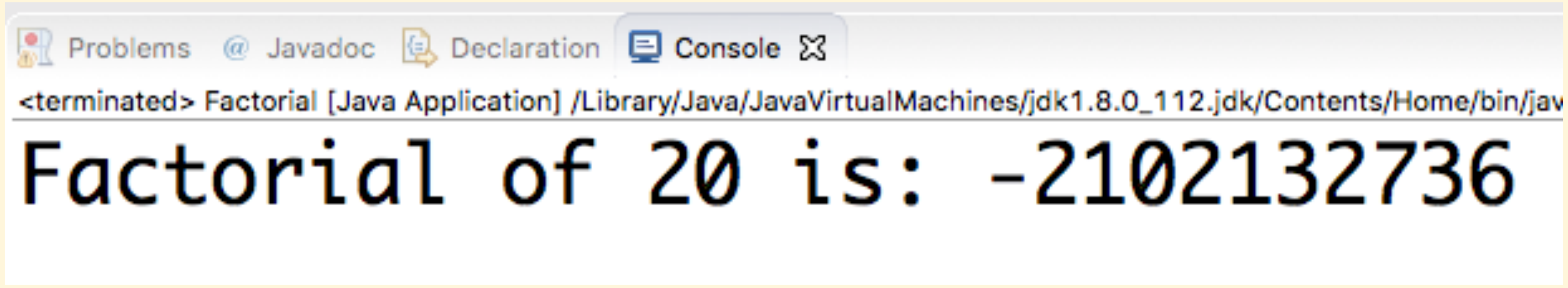
This series of fixed-voltage integrated-circuit voltage regulators is designed for a wide range of applications. These applications include on-card regulation for elimination of noise and distribution problems associated with single-point regulation. Each of these regulators can deliver up to 1.5 A of output current. The internal current-limiting and thermal-shutdown features of these regulators essentially make them immune to overload. In addition to use as fixed-voltage regulators, these devices can be used with external components to obtain adjustable output voltages and currents, and also can be used as the power-pass element in precision regulators.

8.2 Functional Schematic



Given the specification of each component, we can design and use Mathematics to predict the overall behaviour of the system of components

Where are the Specification Sheets and Blueprints for Software Components?

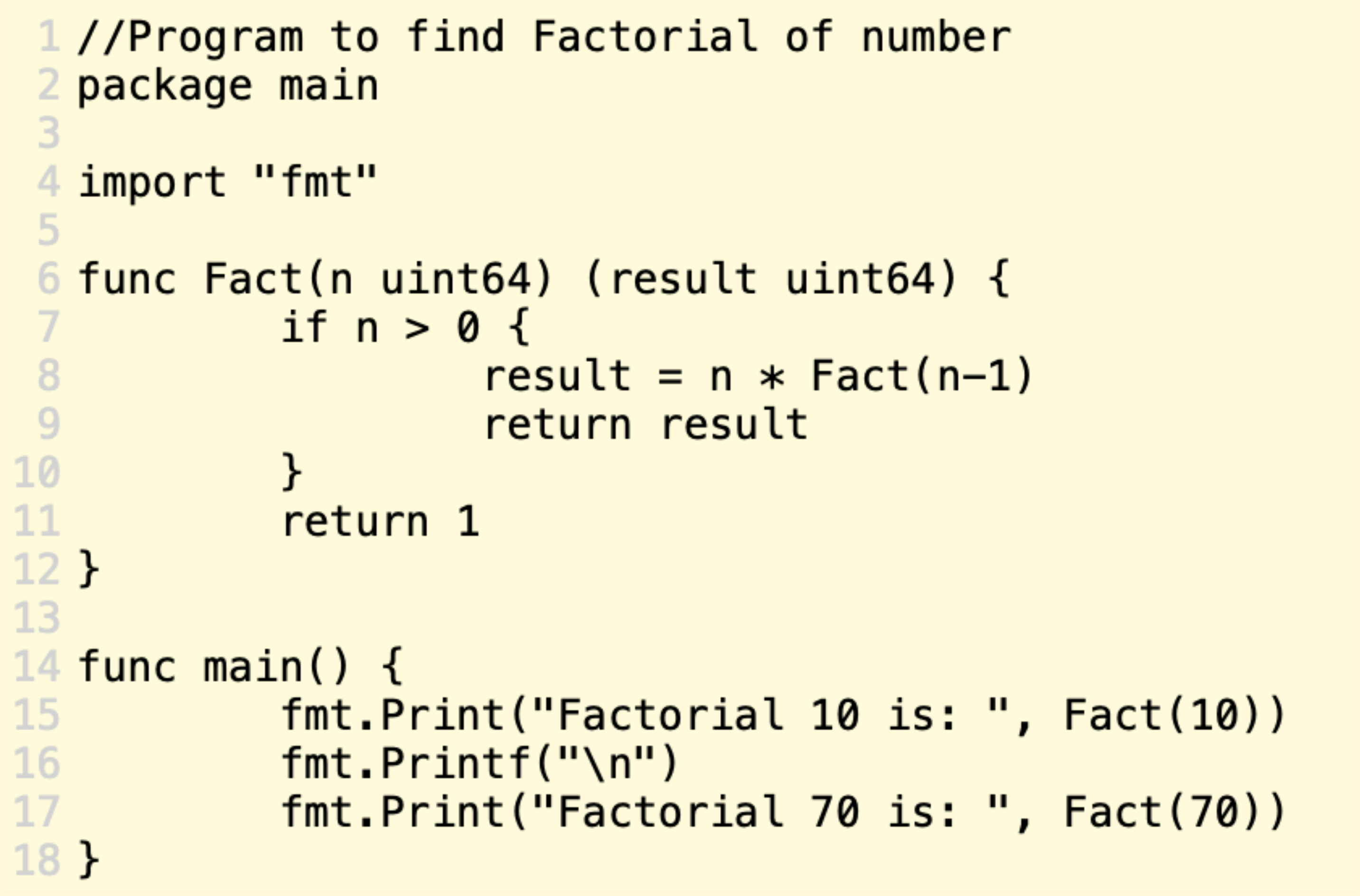


```
<terminated> Factorial [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_112.jdk/Contents/Home/bin/jav
Factorial of 20 is: -2102132736
```

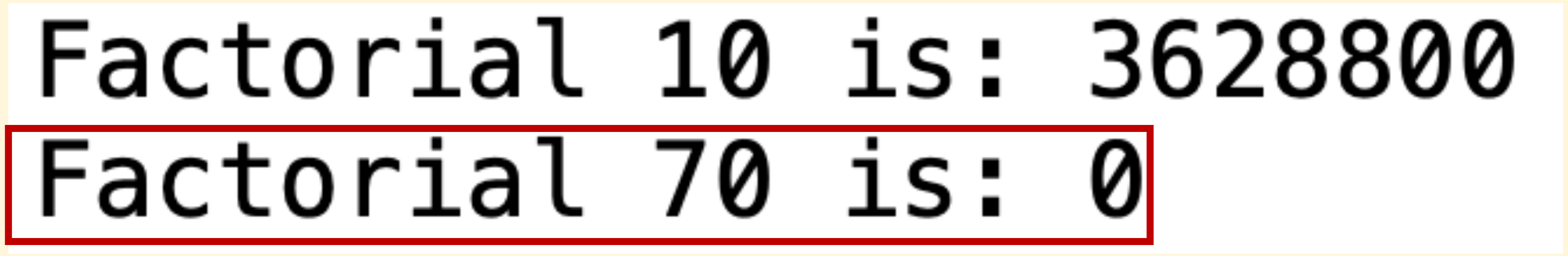
```
static int factorial(int n)
{
    int output;
    if(n==1){
      return 1;
    }
    //Recursion: Function calling itself!!
    output = factorial(n-1)* n;
    return output;
}
```

Precondition needed

Golang

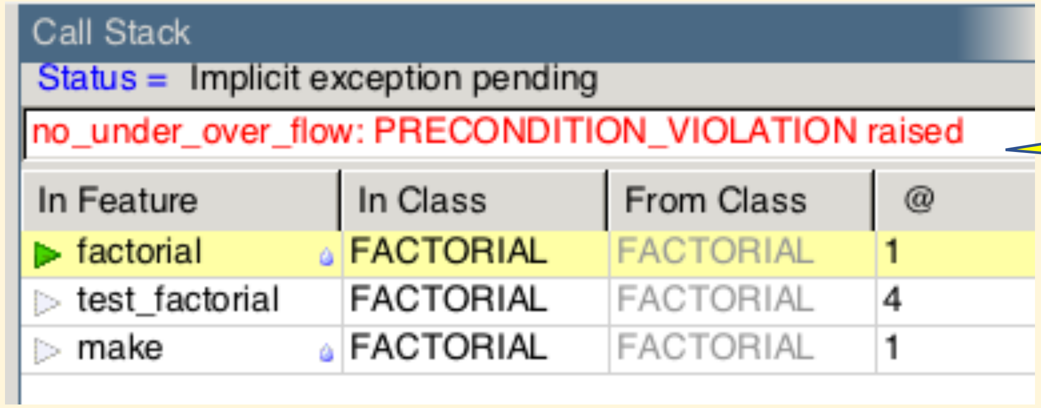


```go
//Program to find Factorial of number
package main

import "fmt"

func Fact(n uint64) (result uint64) {
        if n > 0 {
                result = n * Fact(n-1)
                return result
        }
        return 1
}

func main() {
        fmt.Print("Factorial 10 is: ", Fact(10))
        fmt.Printf("\n")
        fmt.Print("Factorial 70 is: ", Fact(70))
}
```



```
Factorial 10 is: 3628800
Factorial 70 is: 0
```

```
factorial(n:INTEGER): INTEGER
        require
                no_under_over_flow:
                   0 <= n and n <= 12
        do
                if n = 0 then
                        Result := 1
                else
                        Result := n*factorial(n-1)
                end
        ensure
                correct_result:
                   -- Result = (∏i ∈ 1..n: i)
        end
```



Call Stack
Status = Implicit exception pending
no_under_over_flow: PRECONDITION_VIOLATION raised

In Feature	In Class	From Class	@
▶ factorial	FACTORIAL	FACTORIAL	1
▷ test_factorial	FACTORIAL	FACTORIAL	4
▷ make	FACTORIAL	FACTORIAL	1

Runtime Assertion Checking

Nanaimo doctors say electronic health record system unsafe, should be shut down

http://www.theprovince.com/

BY CINDY E. HARNETT, VICTORIA TIMES COLONIST MAY 27, 2016

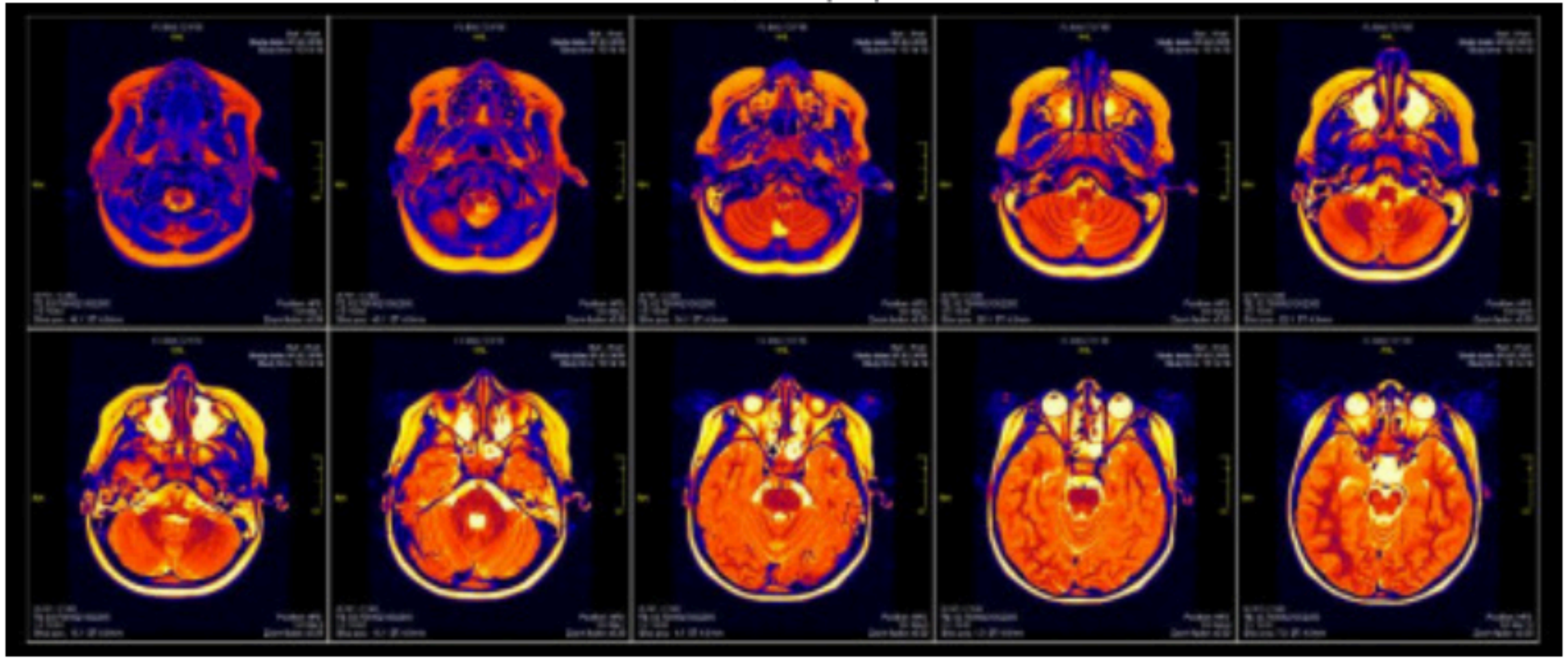
Implementation of a $174-million Vancouver Island-wide electronic health record system in Nanaimo Regional General Hospital — set to expand to Victoria by late 2017 — is a huge failure, say senior physicians.

A bug in fMRI software could invalidate 15 years of brain research

http://www.sciencealert.com/a-bug-in-fmri-software-could-invalidate-decades-of-brain-research-scientists-discover

BEC CREW 6 JUL 2016

There could be a very serious problem with the past 15 years of research into human brain activity, with a new study suggesting that a bug in fMRI software could invalidate the results of some 40,000 papers.



Precondition

Postcondition

Invariant

```
feature -- sqrt
    epsilon: REAL_64

    sqrt(x:REAL_64): REAL_64
            -- return square root of `x'
        require
            argument_non_negative: x >= 0
        local
            guess: REAL_64
        do
            from
                guess := 1
            until
                (guess*guess - x).abs <= epsilon
            loop
                guess := .5*(guess + x/guess)
            end
            Result := guess
        ensure
            result_accurate: (Result * Result - x).abs <= epsilon
            epsilon_unchanged: epsilon = old epsilon
        end

invariant
    correct_to_3_decimal_places:
        0 < epsilon and epsilon <= .001
```

Two questions in software engineering?

- Is a design acceptable that is
  - (a) not feasible
  - or (b) not correct

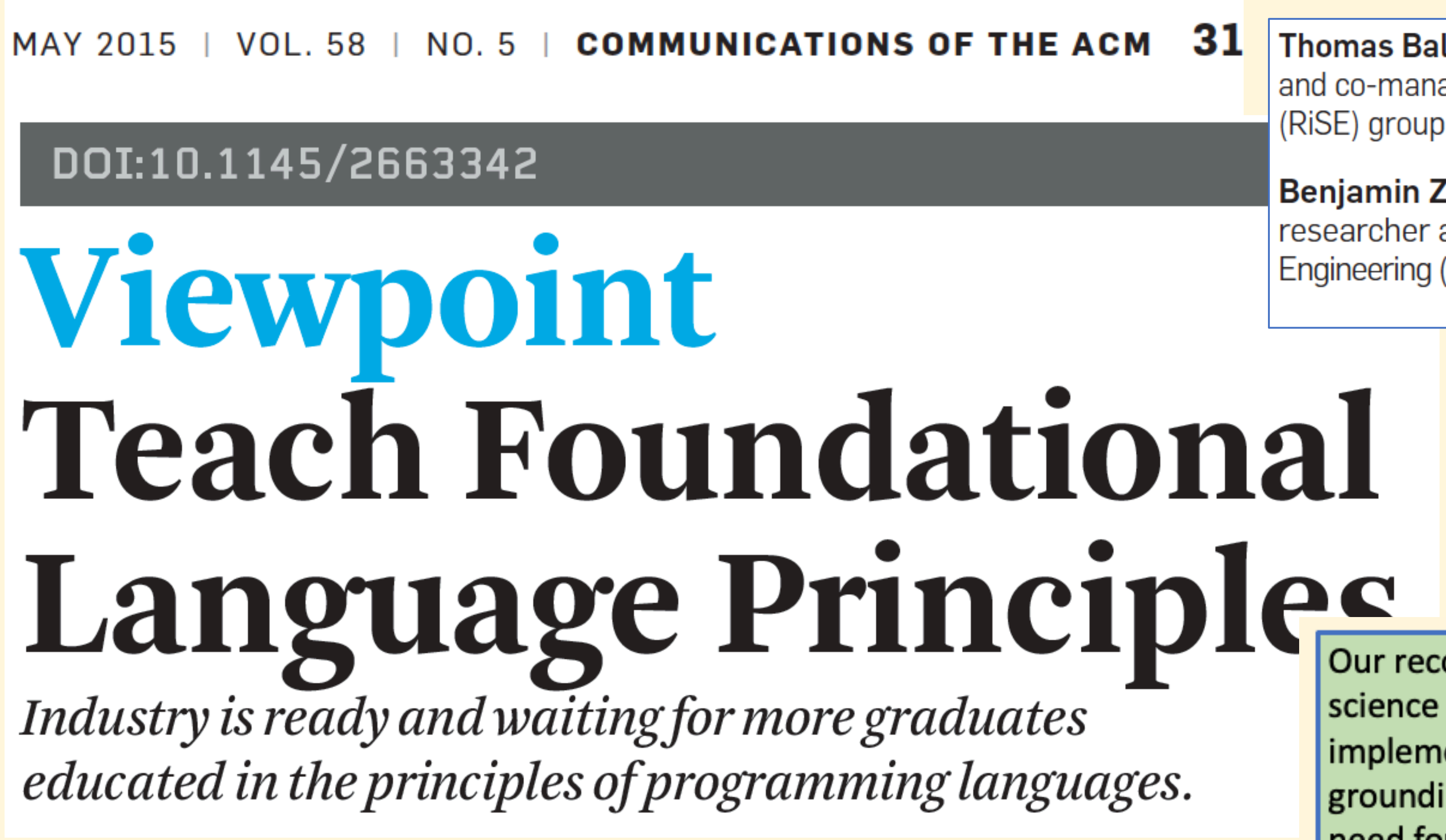
- Should students be taught to write formal specifications? (see next slide)

Recall that in engineering a licensed engineer must demonstrate that the design is safe and fit for purpose

Agree, yes or no?

Our recommendations are threefold, ... First, computer science majors, many of whom will be the designers and implementers of next-generation systems, should get a grounding in logic, ... To designers of complex systems, the need for formal specs should be as obvious as the need for blueprints of a skyscraper.

The methods, tools, and materials for educating students about "formal specs" are ready for prime time. Mechanisms such as $\boxed{\text{"design by contract,"}}$ now available in mainstream programming languages, should be taught as part of introductory programming . ... We are failing our computer science majors if we do not teach them about the value of formal specifications.


MAY 2015 | VOL. 58 | NO. 5 | COMMUNICATIONS OF THE ACM 31

DOI:10.1145/2663342

Viewpoint

Teach Foundational Language Principles

Industry is ready and waiting for more graduates educated in the principles of programming languages.

Thomas Ball (tball@microsoft.com) is a principal researcher and co-manager of the Research in Software Engineering (RiSE) group at Microsoft Research, Redmond, WA.

Benjamin Zorn (zorn@microsoft.com) is a principal researcher and co-manager of the Research in Software Engineering (RiSE) group at Microsoft Research, Redmond, WA.

Our recommendations are threefold, ... First, computer science majors, many of whom will be the designers and implementers of next-generation systems, should get a grounding in logic, ... To designers of complex systems, the need for formal specs should be as obvious as the need for blueprints of a skyscraper.

The methods, tools, and materials for educating students about "formal specs" are ready for prime time. Mechanisms such as "design by contract," now available in mainstream programming languages, should be taught as part of introductory programming . ... We are failing our computer science majors if we do not teach them about the value of formal specifications.

Outside Student Comfort Zone

OO Basics

Memory model: stack vs. heap
Encapsulation
Composition vs. Inheritance
Polymorphism
Static Typing, Dynamic Binding

Design

Abstraction
Divide and Conquer
Specifications vs, Implementations
Program to interface not to implementation
Modularity
Encapsulate what varies
Information Hiding
Uniform Access Principal
Open-Close Principle
Single Choice Principle
Justified Design Decisions

Applying Design Patterns

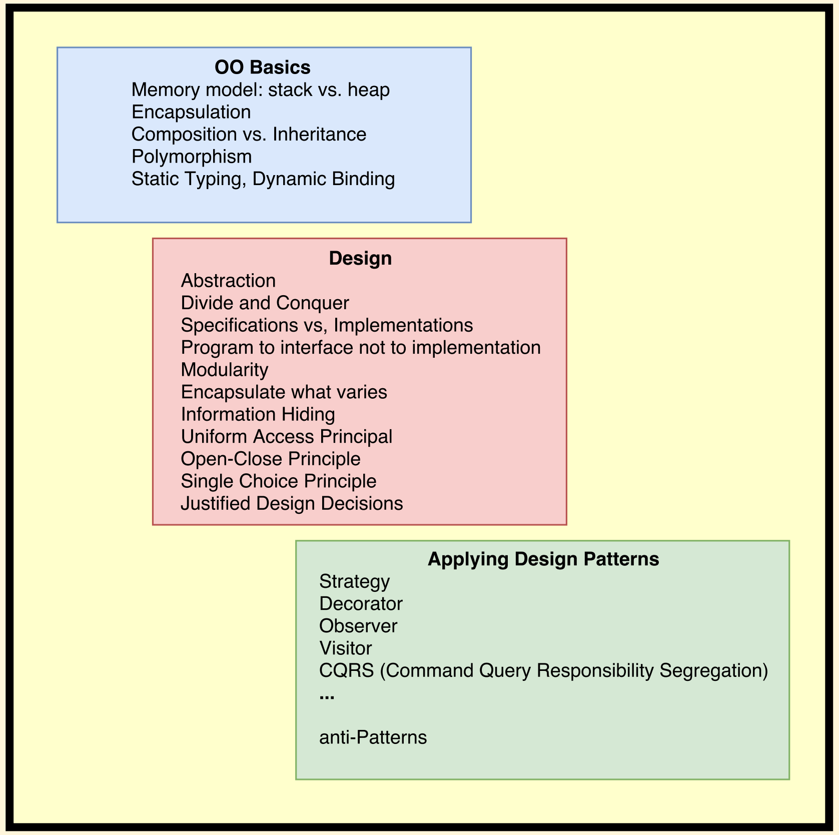
Strategy
Decorator
Observer
Visitor
CQRS (Command Query Responsibility Segregation)
...

anti-Patterns

DbC is used throughout

Why DbC?

1. **Self Documenting**: Specification is also in the program text.
   - Contract View
2. **Contract/Blame**:
   - the client of a module must satisfy the precondition.
   - The Supplier must satisfy the postcondition
   - Loosely coupled modules via their API (app program interface)
3. **Verification**: Verify that the implementation satisfies the specication.
4. **Testing**: Exhaustively test software products using Specification Tests.
5. **Exceptions** are raised only when there are contract violations
   - Avoids code bloat by
   - eliminates the need for constant defensive programming.
6. **Subcontracting**: ensures the Liskov substitution principle so that inheritance is used correctly.

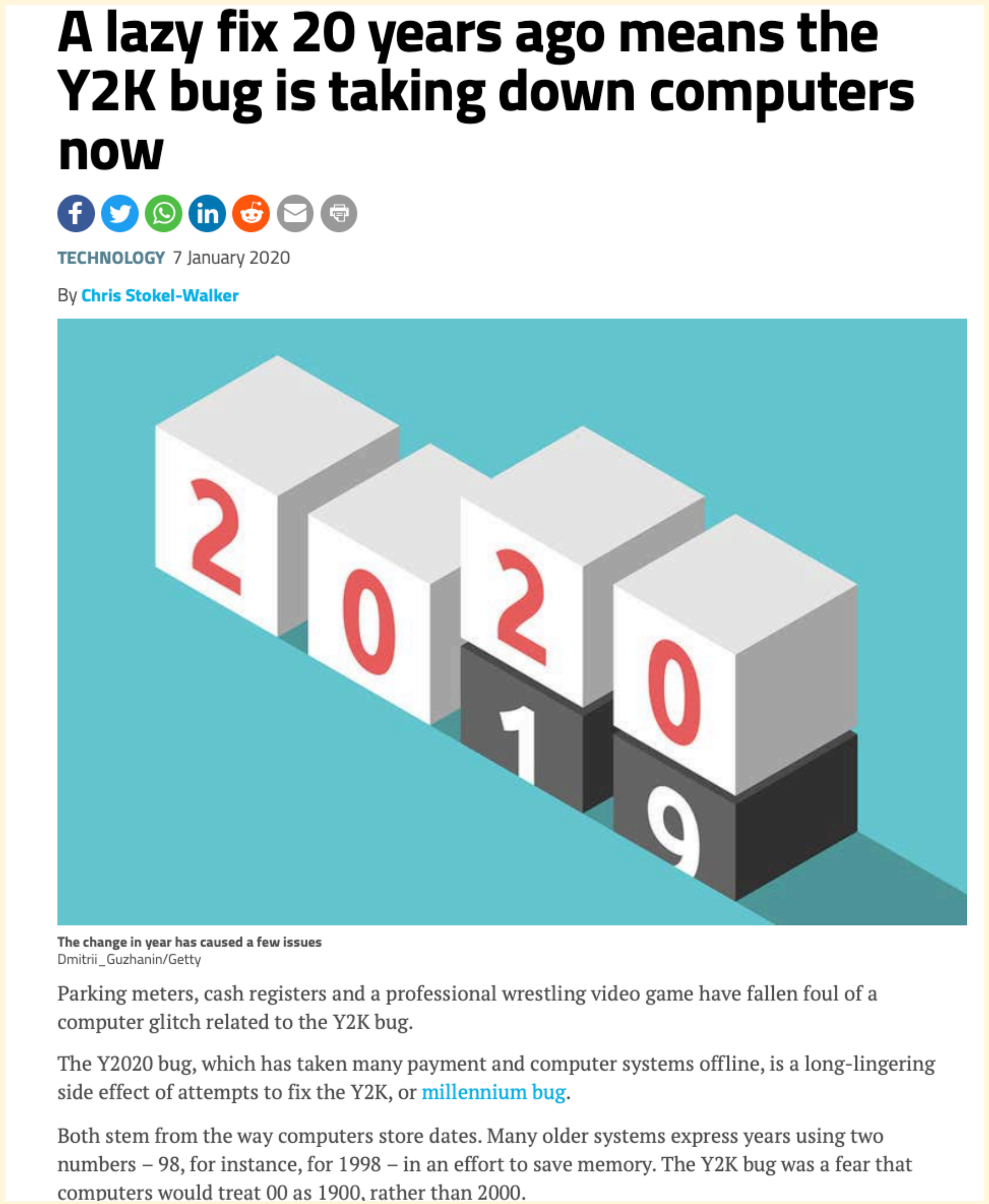


OO Basics
Memory model: stack vs. heap
Encapsulation
Composition vs. Inheritance
Polymorphism
Static Typing, Dynamic Binding

Design
Abstraction
Divide and Conquer
Specifications vs, Implementations
Program to interface not to implementation
Modularity
Encapsulate what varies
Information Hiding
Uniform Access Principal
Open-Close Principle
Single Choice Principle
Justified Design Decisions

Applying Design Patterns
Strategy
Decorator
Observer
Visitor
CQRS (Command Query Responsibility Segregation)
...

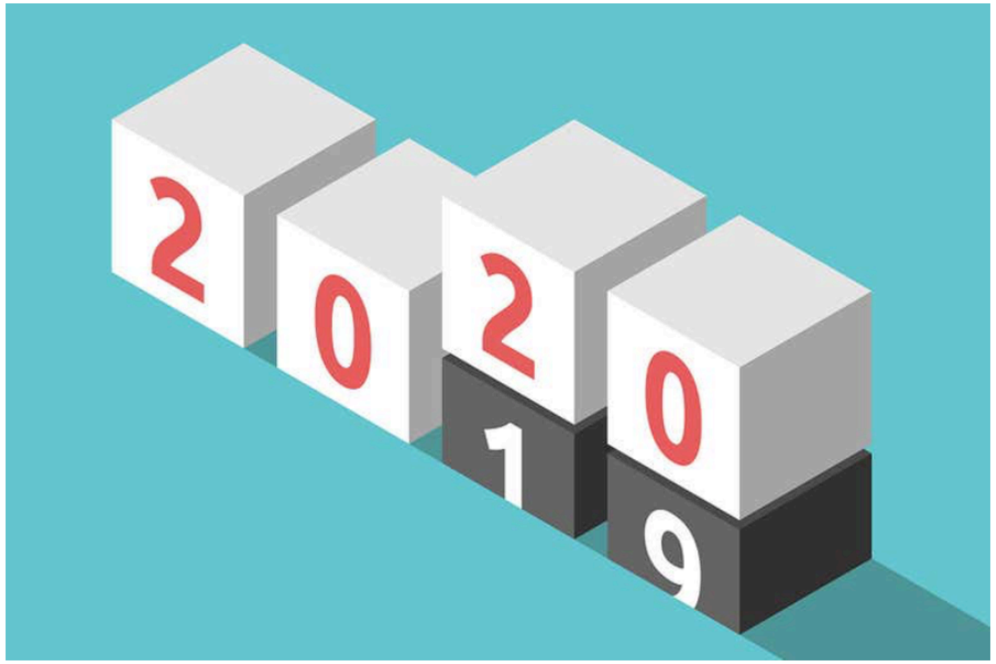
anti-Patterns



A lazy fix 20 years ago means the Y2K bug is taking down computers now

TECHNOLOGY 7 January 2020

By Chris Stokel-Walker

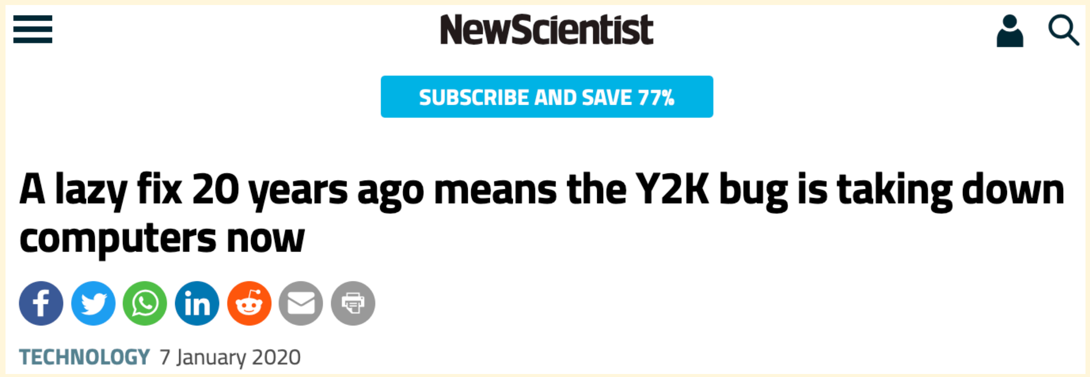


The change in year has caused a few issues
Dmitrii_Guzhanin/Getty

Parking meters, cash registers and a professional wrestling video game have fallen foul of a computer glitch related to the Y2K bug.

The Y2020 bug, which has taken many payment and computer systems offline, is a long-lingering side effect of attempts to fix the Y2K, or millennium bug.

Both stem from the way computers store dates. Many older systems express years using two numbers – 98, for instance, for 1998 – in an effort to save memory. The Y2K bug was a fear that computers would treat 00 as 1900, rather than 2000.

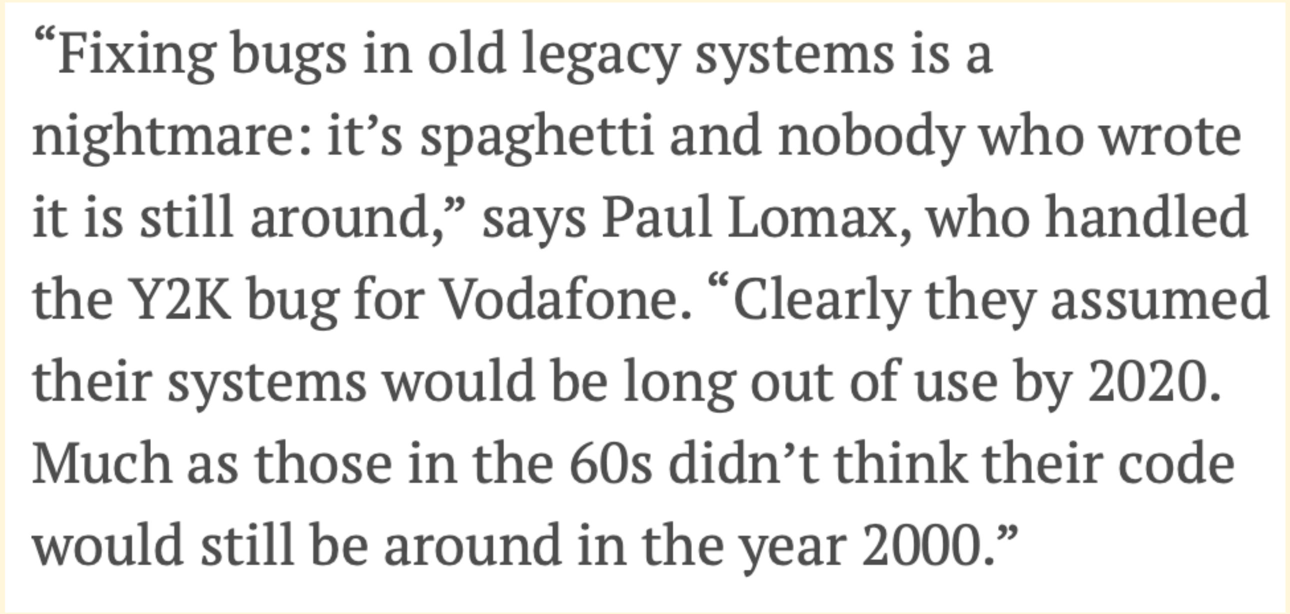


NewScientist

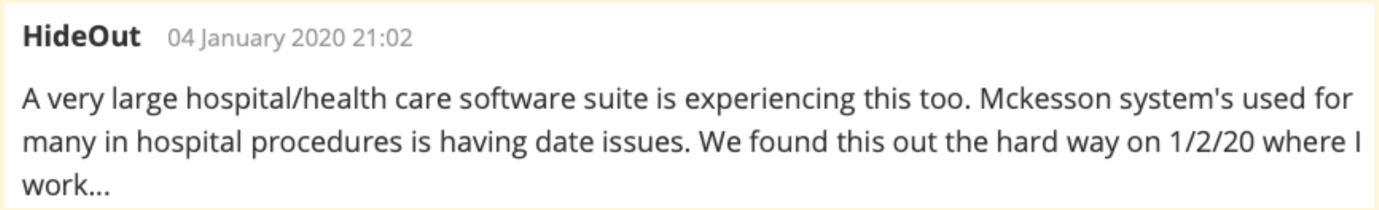
SUBSCRIBE AND SAVE 77%

A lazy fix 20 years ago means the Y2K bug is taking down computers now

TECHNOLOGY 7 January 2020

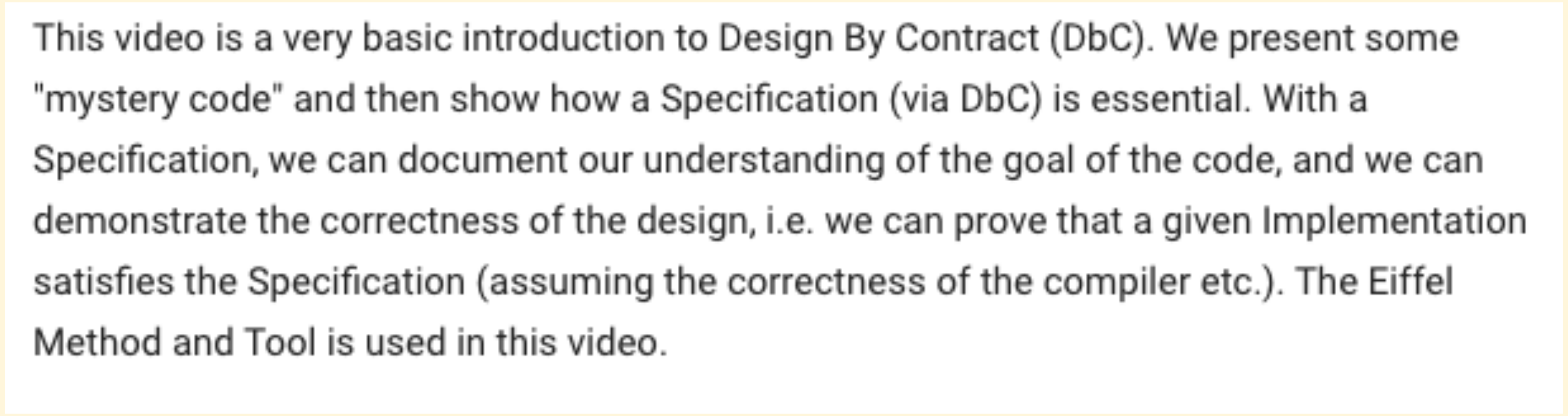


"Fixing bugs in old legacy systems is a nightmare: it's spaghetti and nobody who wrote it is still around," says Paul Lomax, who handled the Y2K bug for Vodafone. "Clearly they assumed their systems would be long out of use by 2020. Much as those in the 60s didn't think their code would still be around in the year 2000."



HideOut 04 January 2020 21:02

A very large hospital/health care software suite is experiencing this too. Mckesson system's used for many in hospital procedures is having date issues. We found this out the hard way on 1/2/20 where I work...


This video is a very basic introduction to Design By Contract (DbC). We present some "mystery code" and then show how a Specification (via DbC) is essential. With a Specification, we can document our understanding of the goal of the code, and we can demonstrate the correctness of the design, i.e. we can prove that a given Implementation satisfies the Specification (assuming the correctness of the compiler etc.). The Eiffel Method and Tool is used in this video.

See next video

02 DbC Specification Mystery

http://seldoc.eecs.yorku.ca/doku.php/eiffel/why/start