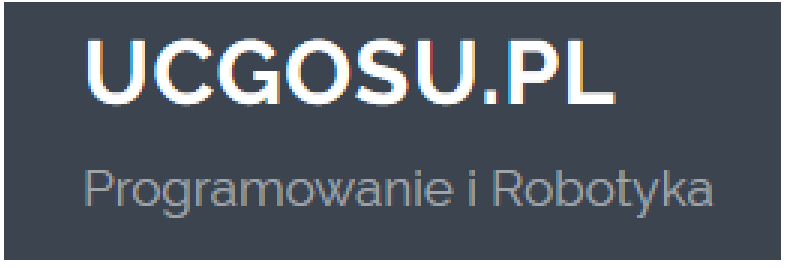
What could we learn from high-level programmers?

Maciek Gajdzica

About me

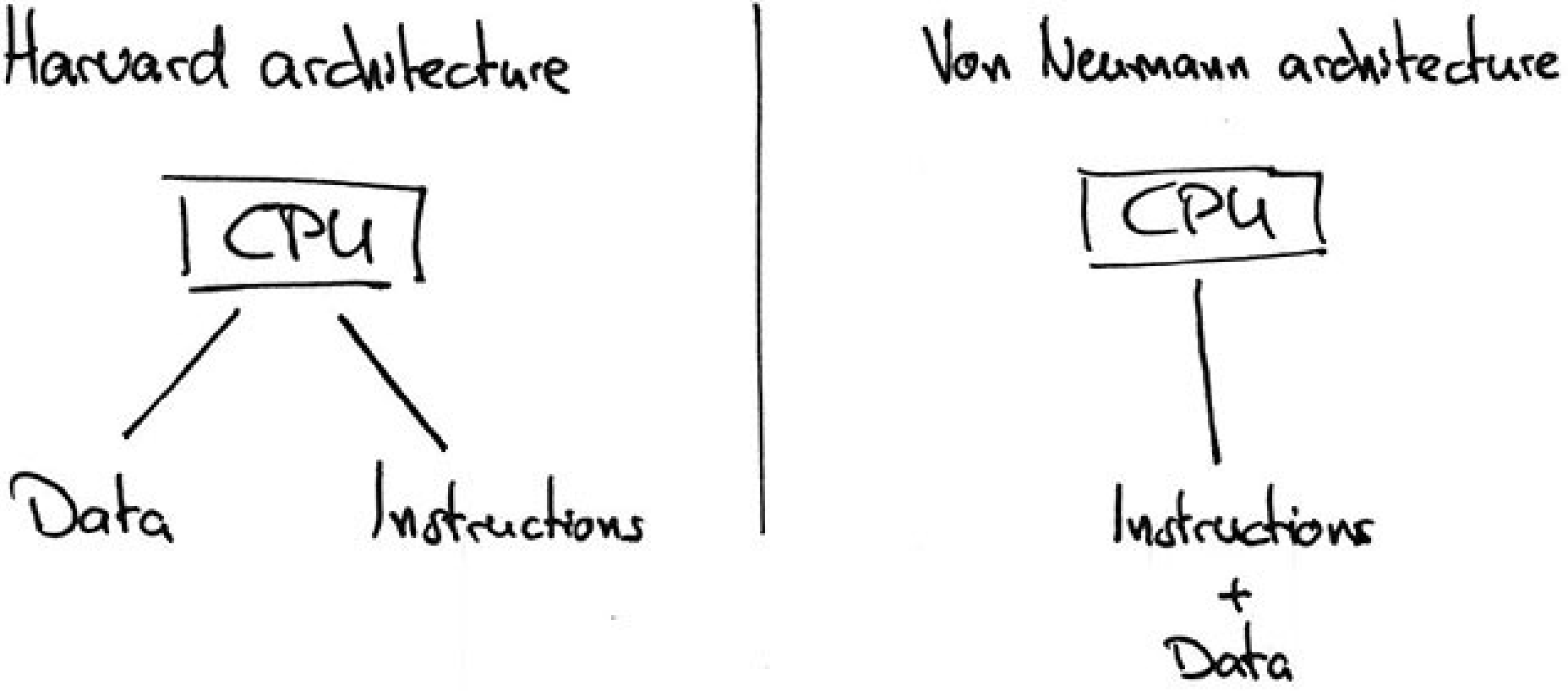
- Embedded dev for 7 years
- Railway, automotive, medical, IoT, home automation
- Organizing Gdańsk Embedded Meetup
- Blog: ucgosu.pl (in Polish)
- Twitter: @MaciekGajdzica



Atmega32

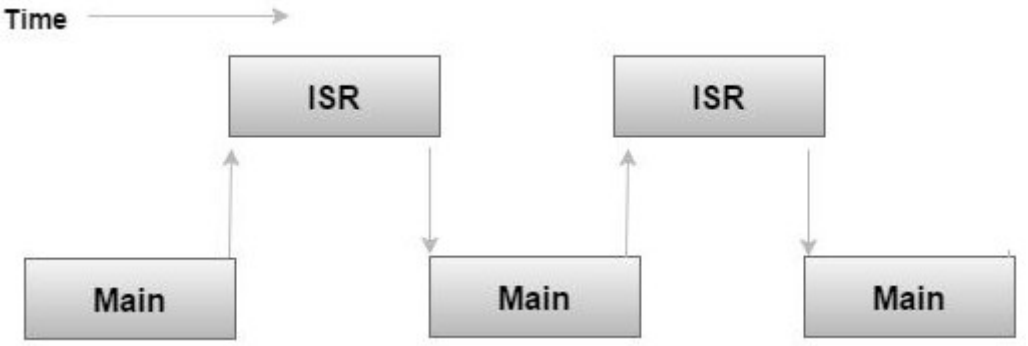
- Clock 16 MHz
- FLASH 32 kB
- RAM 2 kB

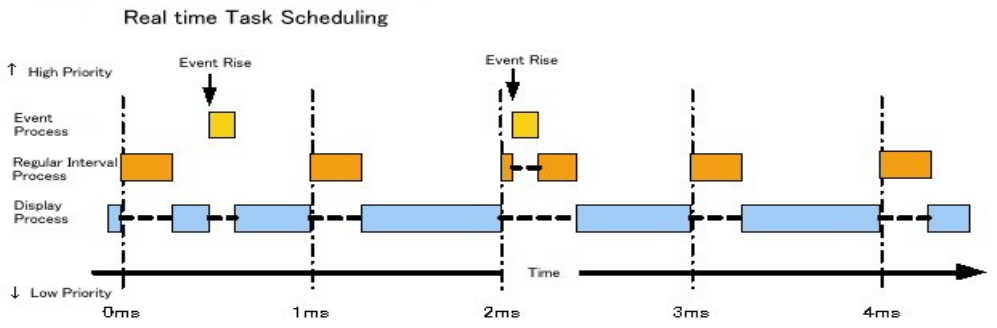
Embedded architecture?



Embedded architecture?

```
#include <avr/io.h>

int main(void)
{
   DDRD  = 0xff;

   DDRC  = 0x00;
   PORTC = 0x03;

   while(1)
   {
     if(!(PINC & 0x01))
     {
         PORTD = 0xF0;
     }

     if(!(PINC & 0x02))
     {
         PORTD = 0x0F;
     }
   }
}
```

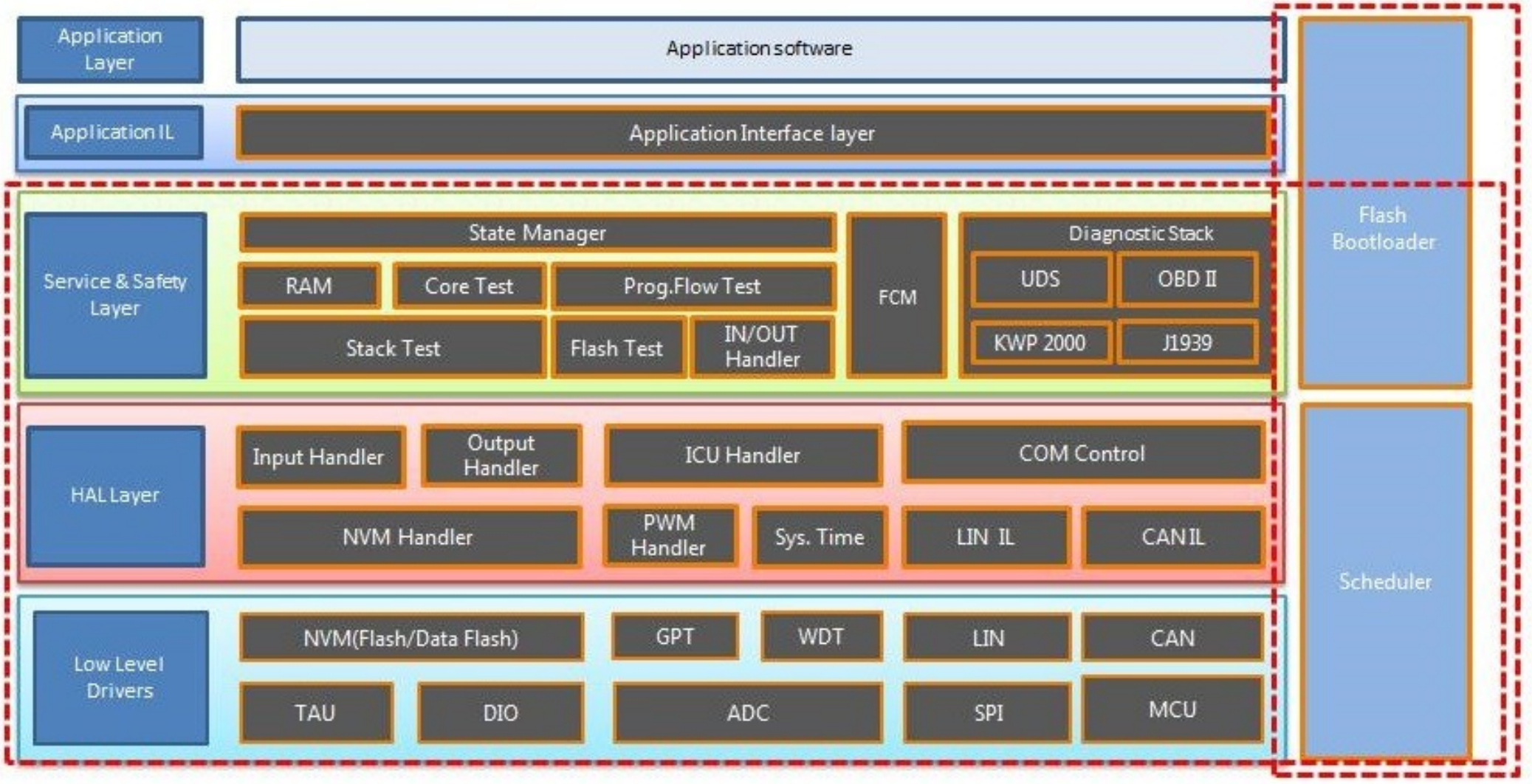


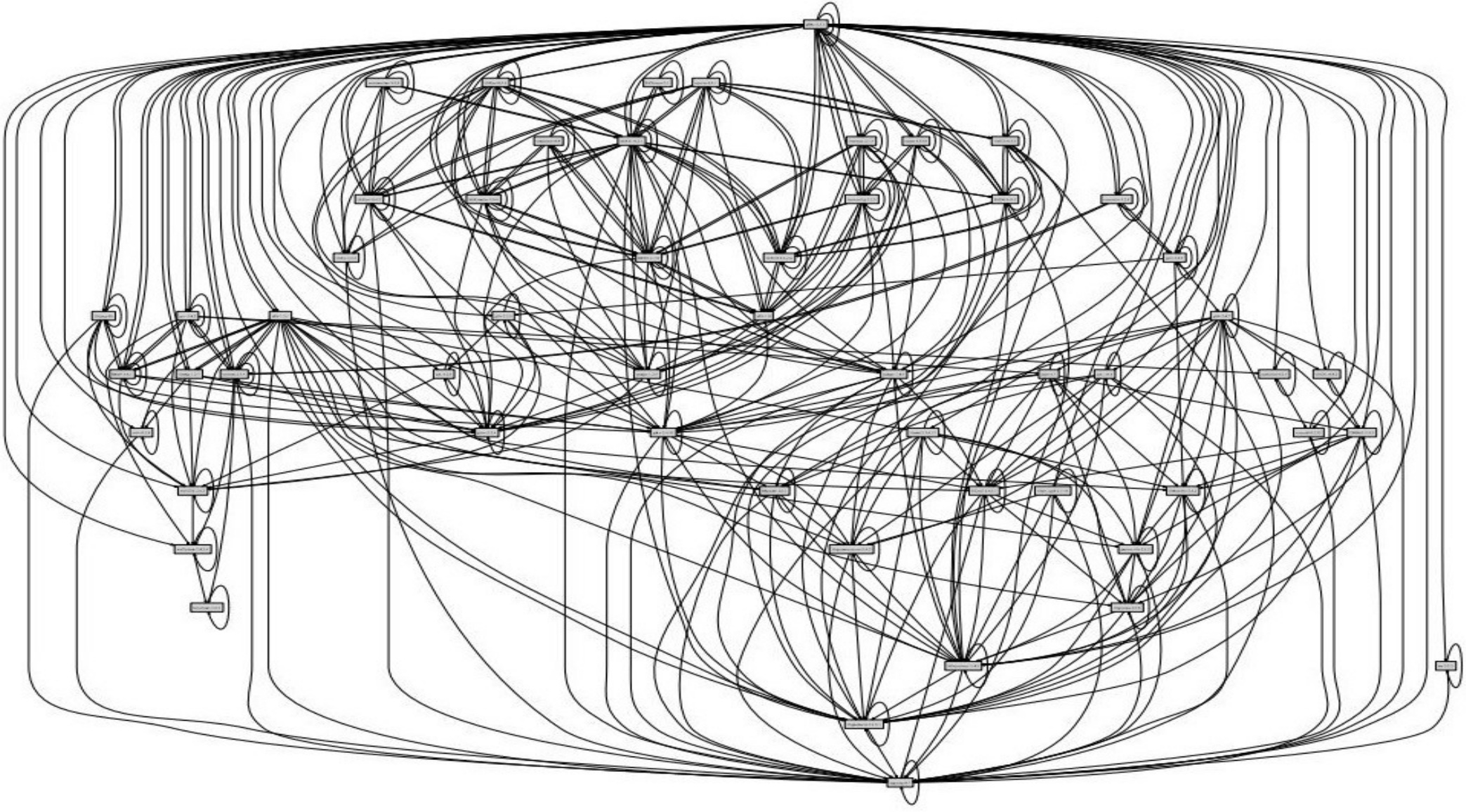


STM32

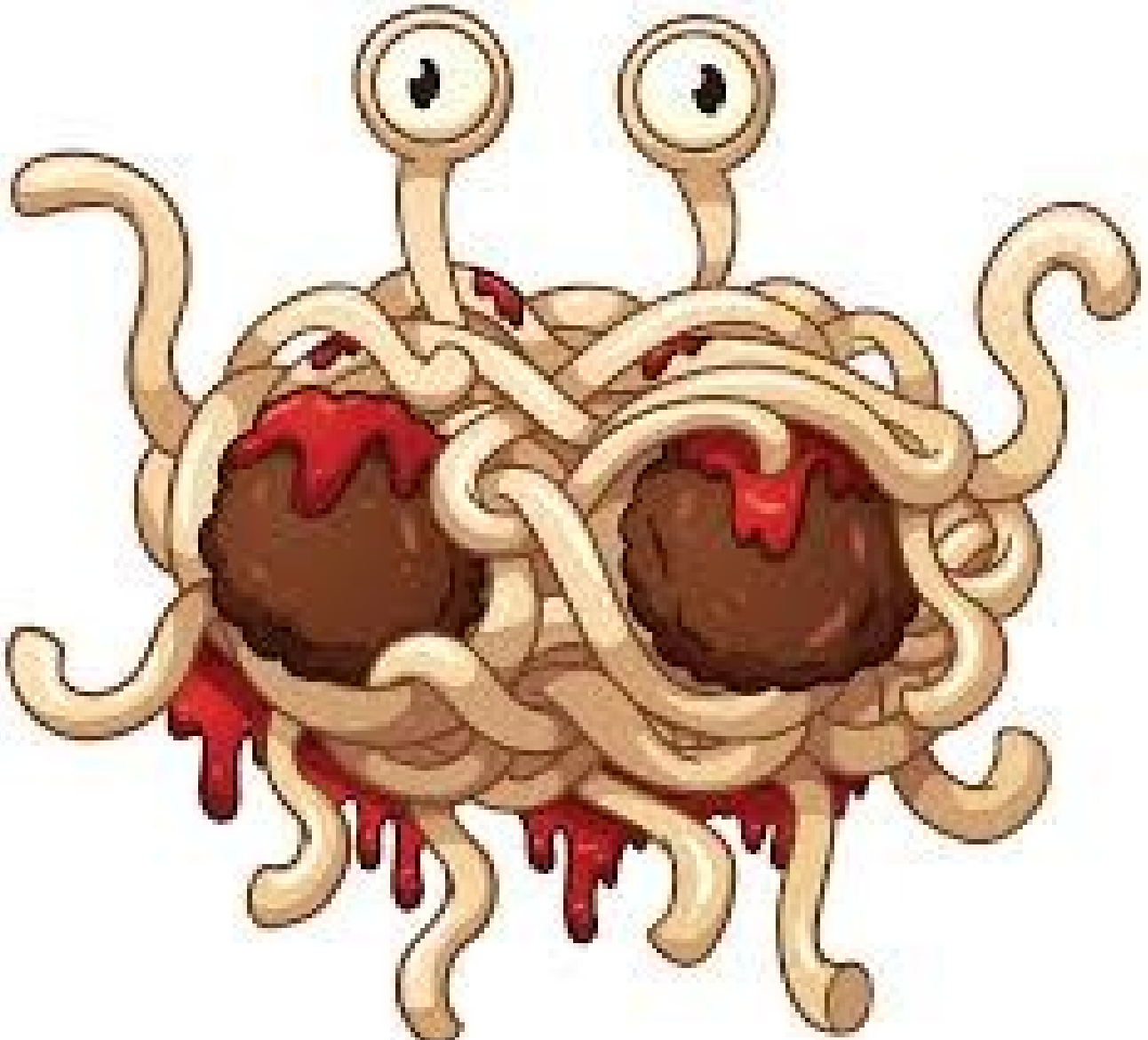
- Clock 180 MHz
- FLASH 2048 kB
- RAM 256 kB

Software Architecture

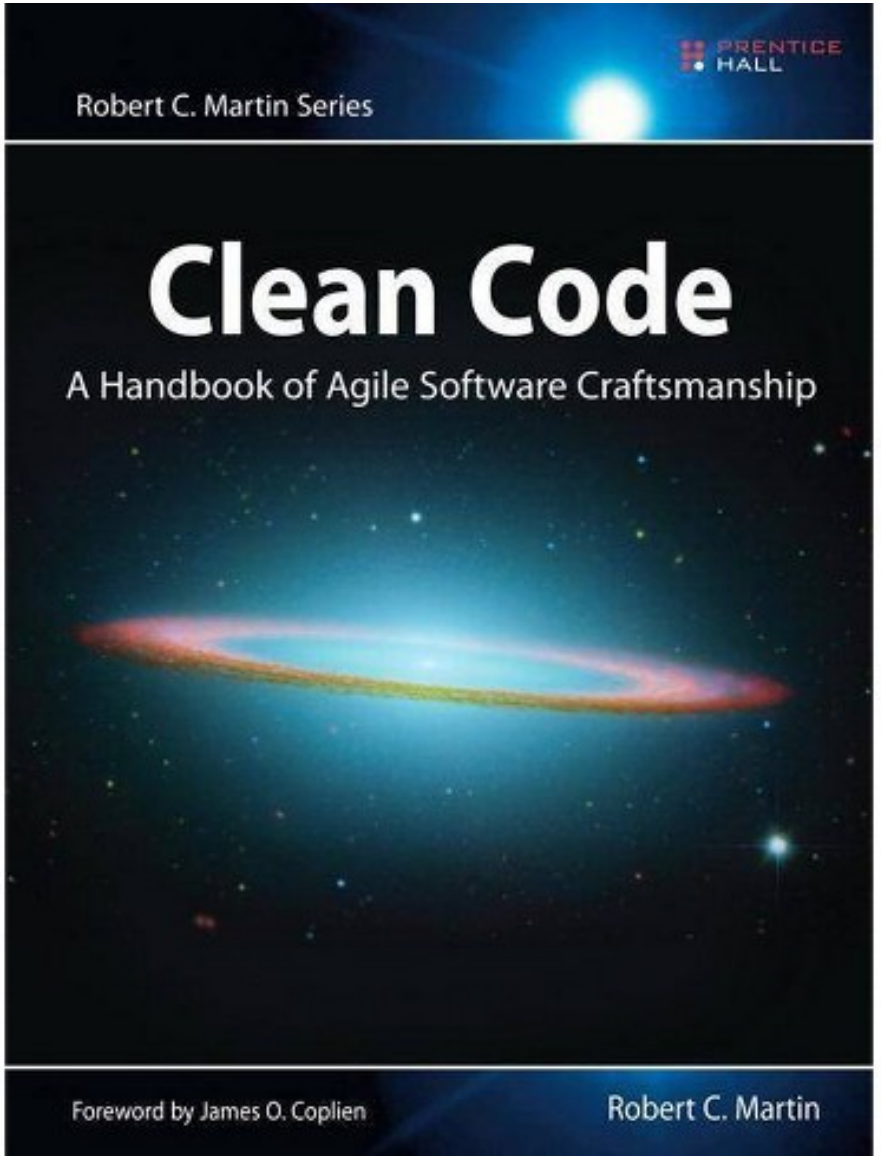




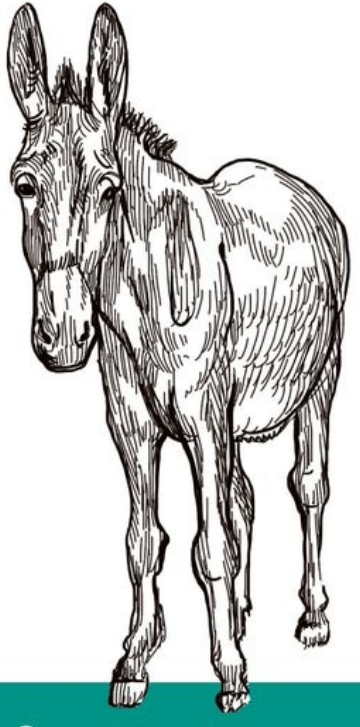

UniCredit
SONY
Ford
PS3
UEFA.com
POR QUÉ?



```
void  Sim80x_SetPower(bool TurnOn)
{
  if(TurnOn==true)
  {
    if(Sim80x_SendAtCommand("AT\r\n",200,1,"AT\r\r\nOK\r\n") == 1)
    {
      osDelay(100);
      #if (_SIM80X_DEBUG==1)
      printf("\r\nSim80x_SetPower(ON) ---> OK\r\n");
      #endif
      Sim80x.Status.Power=1;
      Sim80x_InitValue();
    }
    else
    {
      #if (_SIM80X_USE_POWER_KEY==1)
      HAL_GPIO_WritePin(_SIM80X_POWER_KEY_GPIO,_SIM80X_POWER_KEY_PIN,GPIO_PIN_RESET);
      osDelay(1200);
      HAL_GPIO_WritePin(_SIM80X_POWER_KEY_GPIO,_SIM80X_POWER_KEY_PIN,GPIO_PIN_SET);
      #endif
      osDelay(3000);
      if(Sim80x_SendAtCommand("AT\r\n",200,1,"AT\r\r\nOK\r\n") == 1)
      {
        osDelay(10000);
        #if (_SIM80X_DEBUG==1)
        printf("\r\nSim80x_SetPower(ON) ---> OK\r\n");
        #endif
        Sim80x.Status.Power=1;
        Sim80x_InitValue();
      }
```


Robert C. Martin Series
PRENTICE HALL
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin

Where's the fun in just knowing what the code is supposed to do?



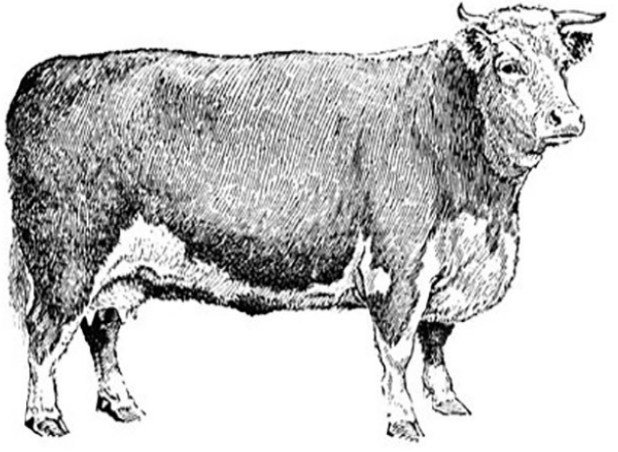
Essential

Excuses for Not Writing Documentation

O RLY?

@ThePracticalDev

No comments, no documentation but 20 tickets



The Guy Who Wrote This Is Gone

It's running everywhere

O RLY?

FML

Documentation

- Developer’s Torment: The Documentation - Jakub Marchwicki, Zbyszko Papierski
- Architecture Decision Records

Debugging

“Ok it compiles, so let’s get to the real work – debugging”

Dan Saks – Meeting Embedded 2018

Advanced and expensive HW debuggers:

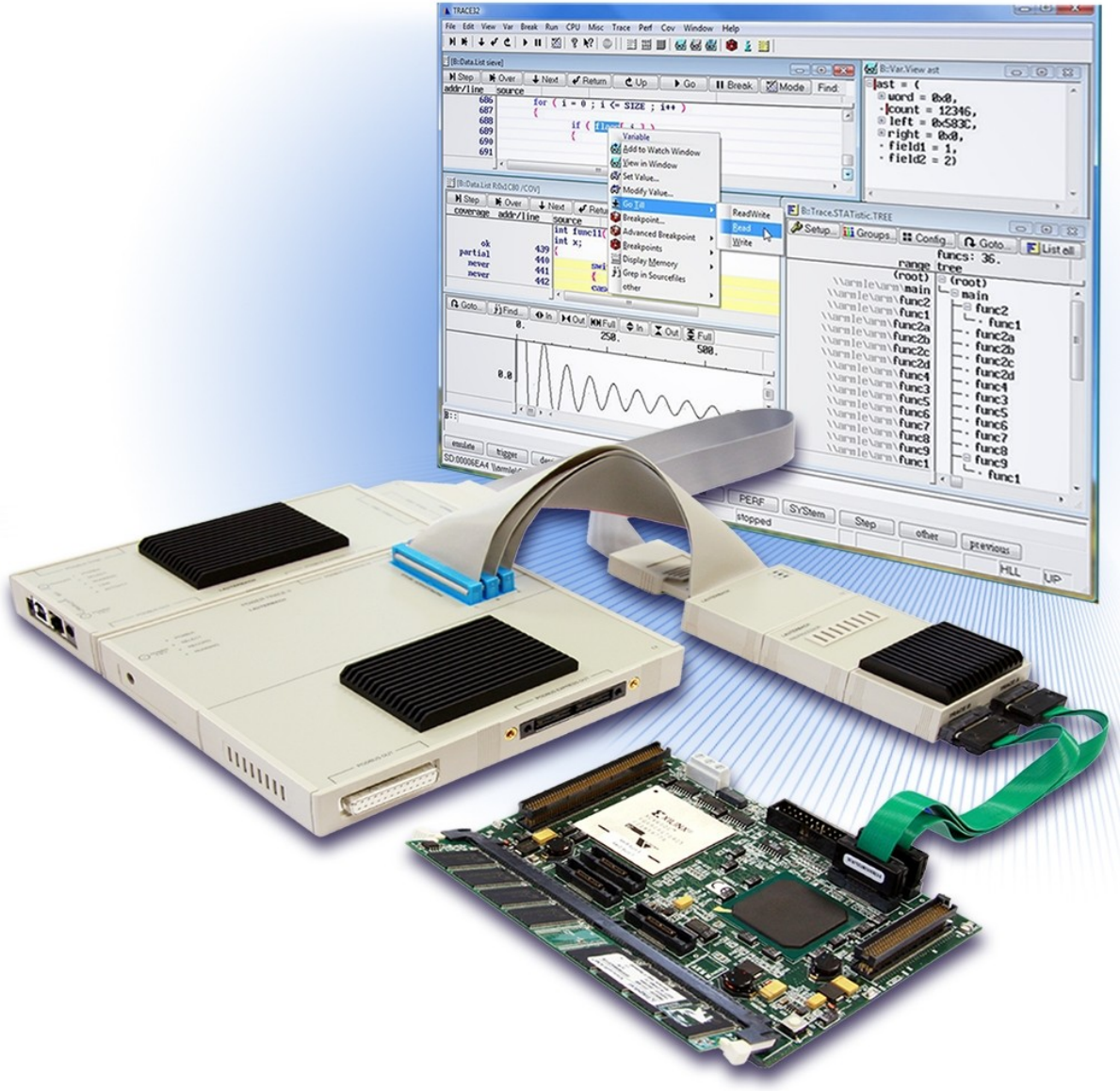
- Call stack
- Trace
- Step back
- Print log

Modern compilers:

- Overcoming C limitations
- Additional optimizations and options
- MISRA support
- Additional warnings

External devices:

- Oscilloscope
- Logic analyzers
- Communication protocol sniffers
- Signal generators



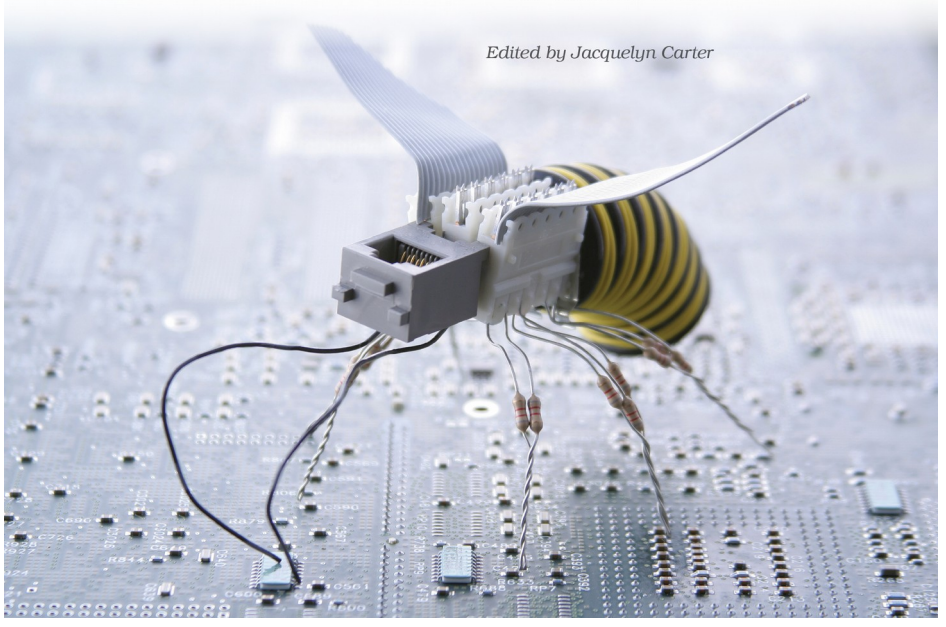
The Pragmatic Programmers

Test-Driven Development for Embedded C

James W. Grenning

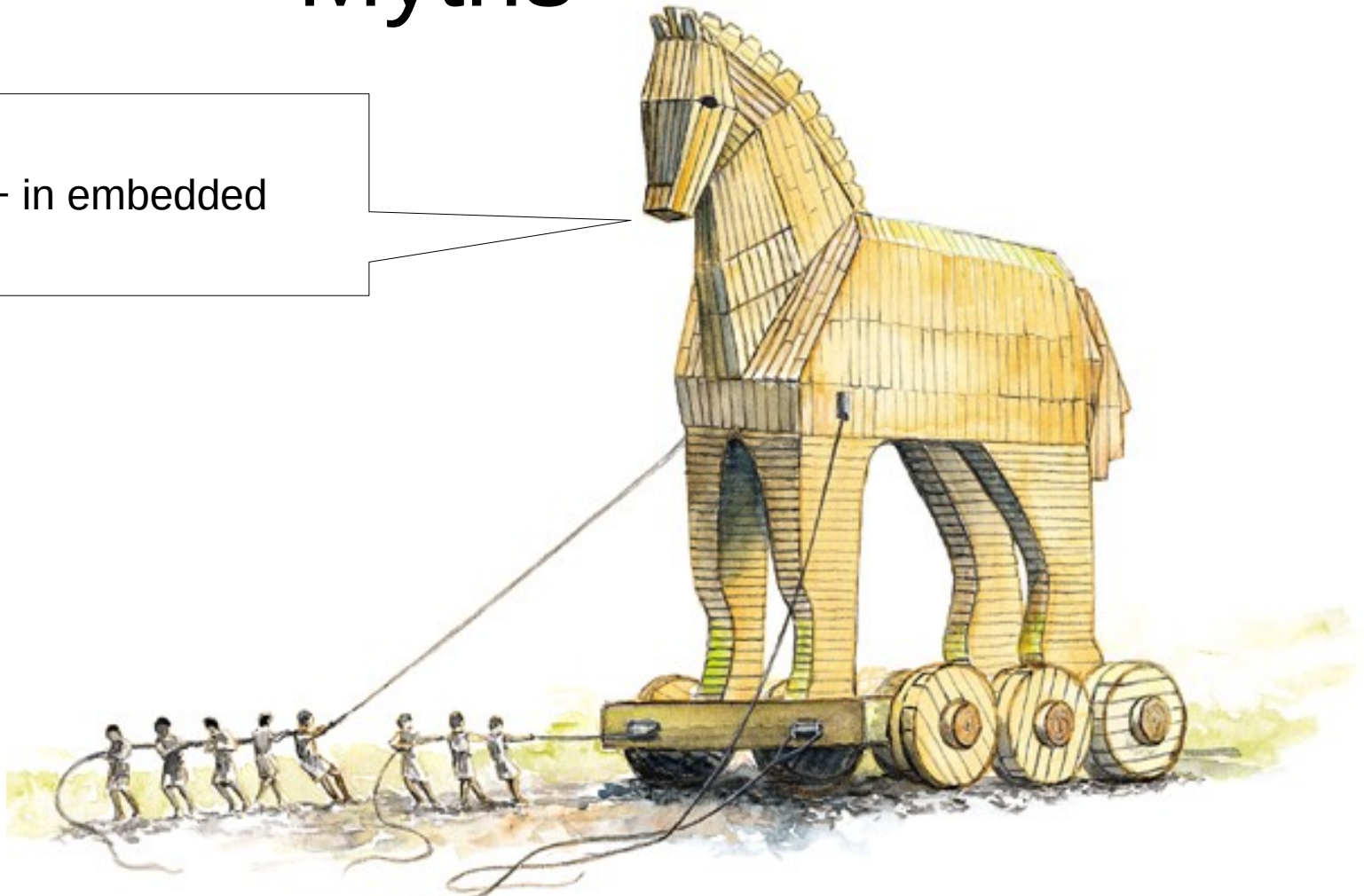
Forewords by Jack Ganssle and Robert C. Martin

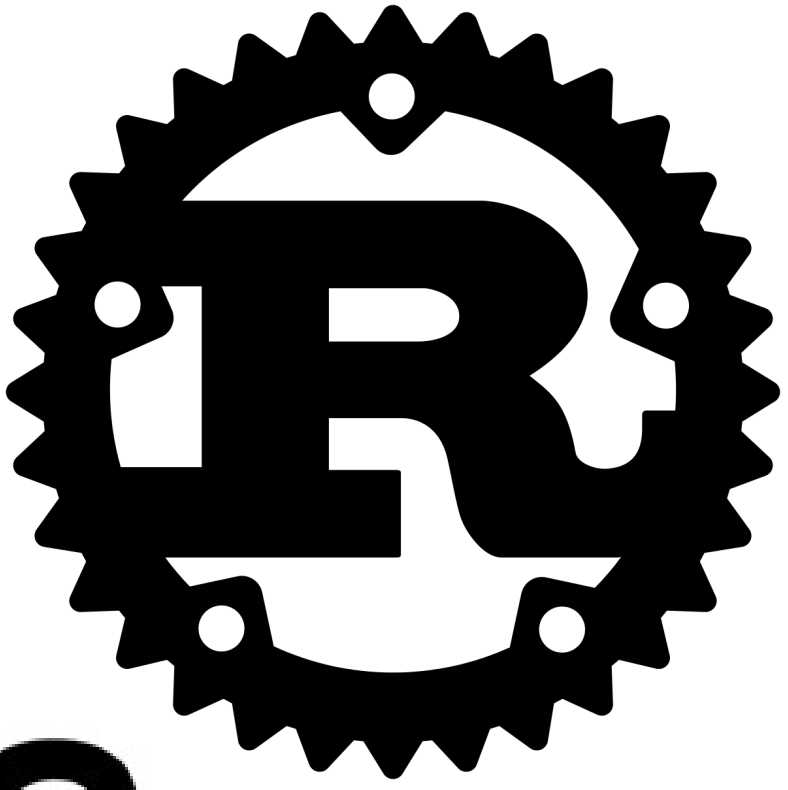
Edited by Jacquelyn Carter



Myths

Don’t use C++ in embedded






Ada
2012

MORE TOOLS!!!

Event 2			
Parameter	Type	Actual Value	Expected Value
Stub called: manager.c			
Subprogram: Get_Table_Record			

Event 3			
Parameter	Type	Actual Value	Expected Value
Stub called: manager.c			
Subprogram: Update_Table_Record			
Data			
Is_Occupied	enum	v_true	match
Number_In_Party	unsigned short	1	match
Check_Total	float	14	match

Event 4			
Parameter	Type	Actual Value	Expected Value
Returned from UUT: manager.c			
Subprogram: Place_Order			
Table	unsigned short	1	
Seat	unsigned short	1	
Order			
Entree	enum	STEAK	

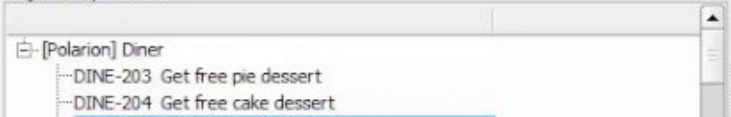
-- End of Test Case --

Test Status	
Expected Results matched 100%	**(3/3) PASS**
Test Status	**PASS**

Col: 0 Line: 1

✓ Parameter Tree | Control Flow | ✓ Execution Report

Project Requirements

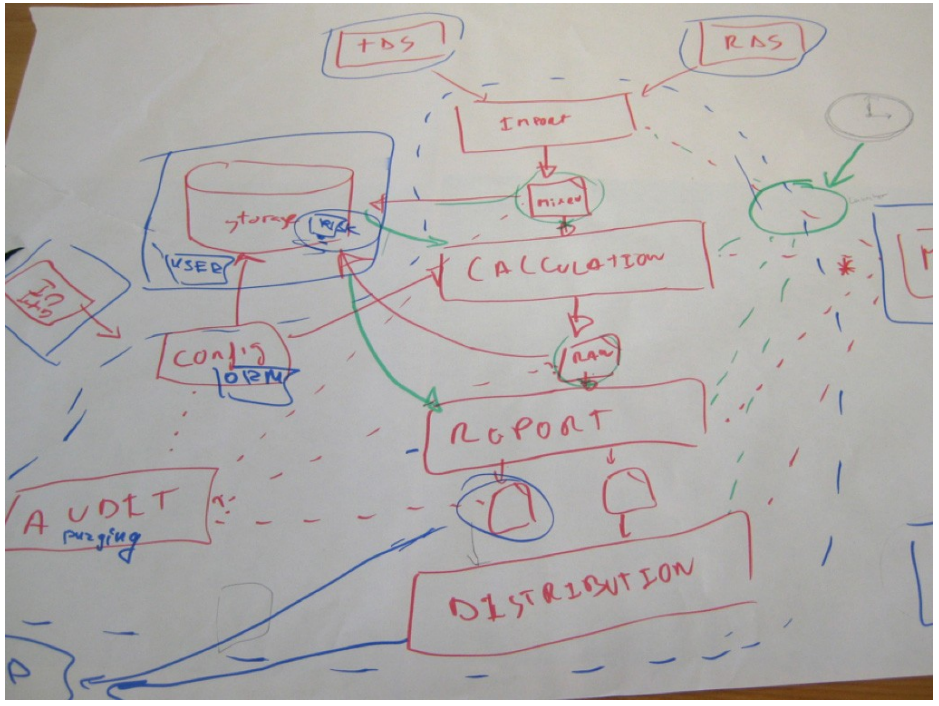


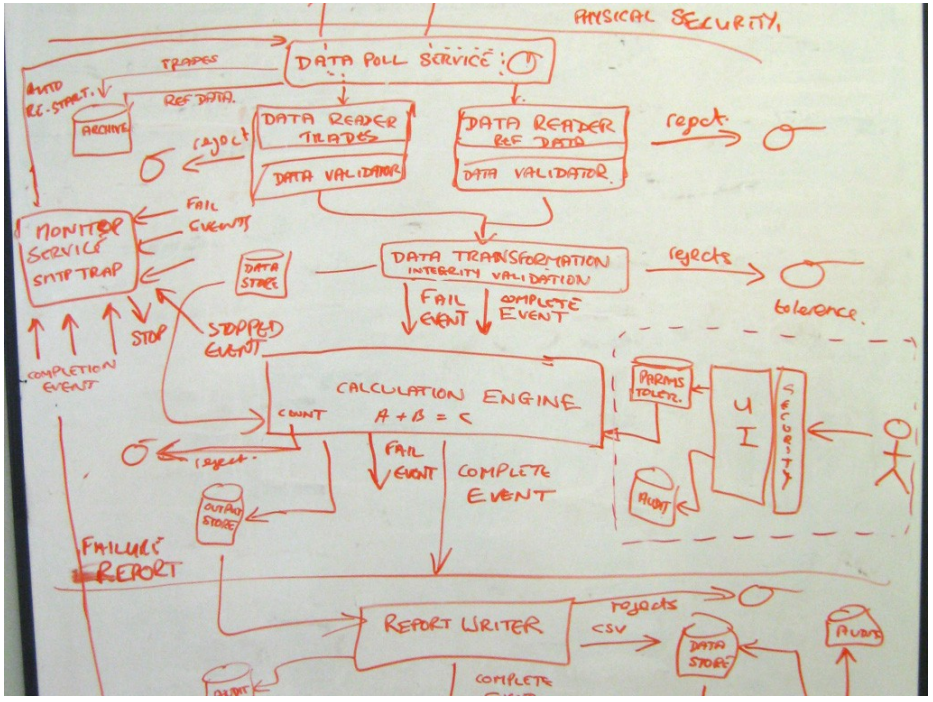
Test Case Requirements

ID	Title
DINE-79	Placing an order updates occupied status
DINE-80	Placing an order updates number in party
DINE-205	Steak Order is $14

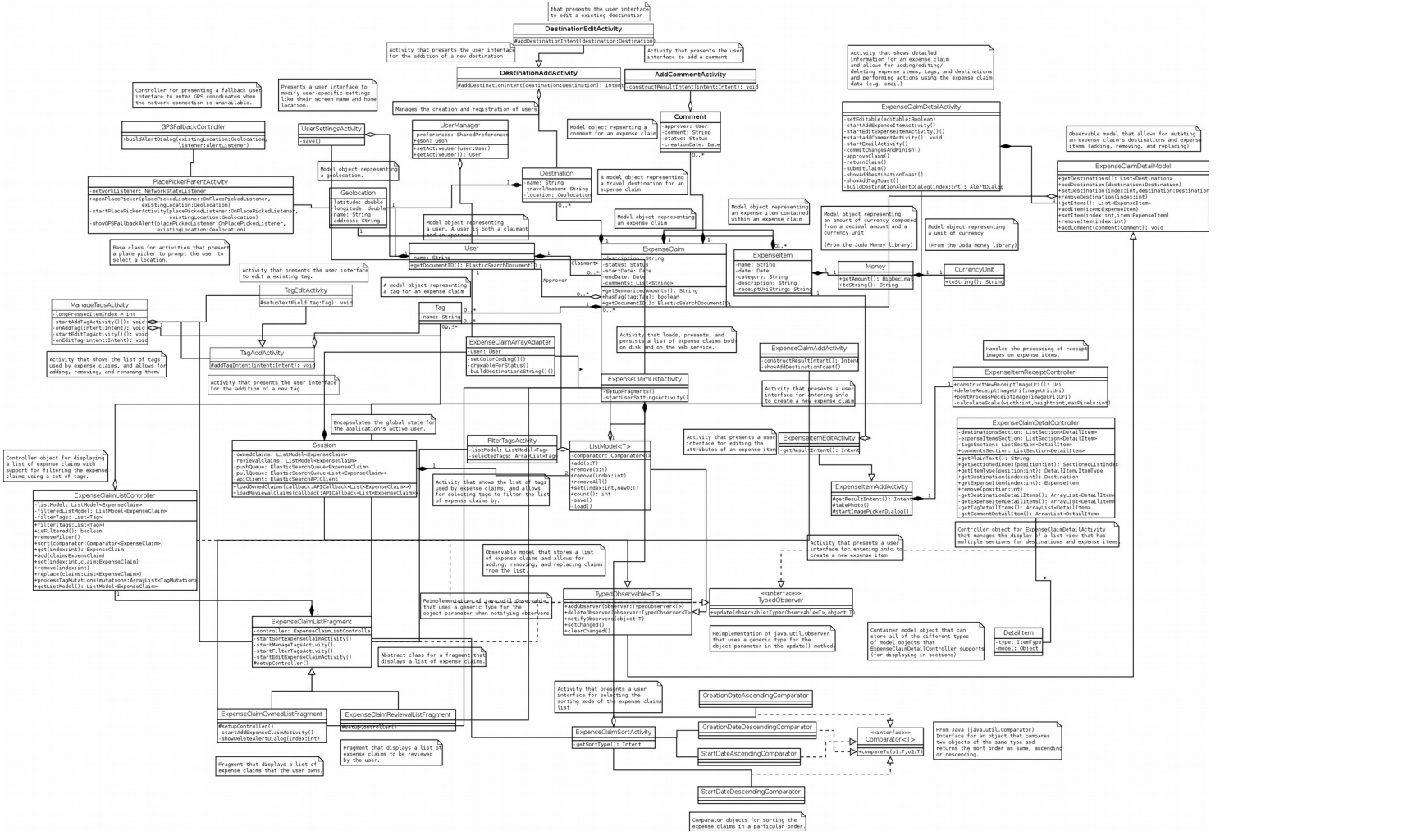

B1
B2
B3

Visualizing architecture





Visualizing architecture



The C4 model

System Context

The system plus users and system dependencies

Containers

The overall shape of the architecture and technology choices

Components

Components and their interactions within a container

Classes (or Code)

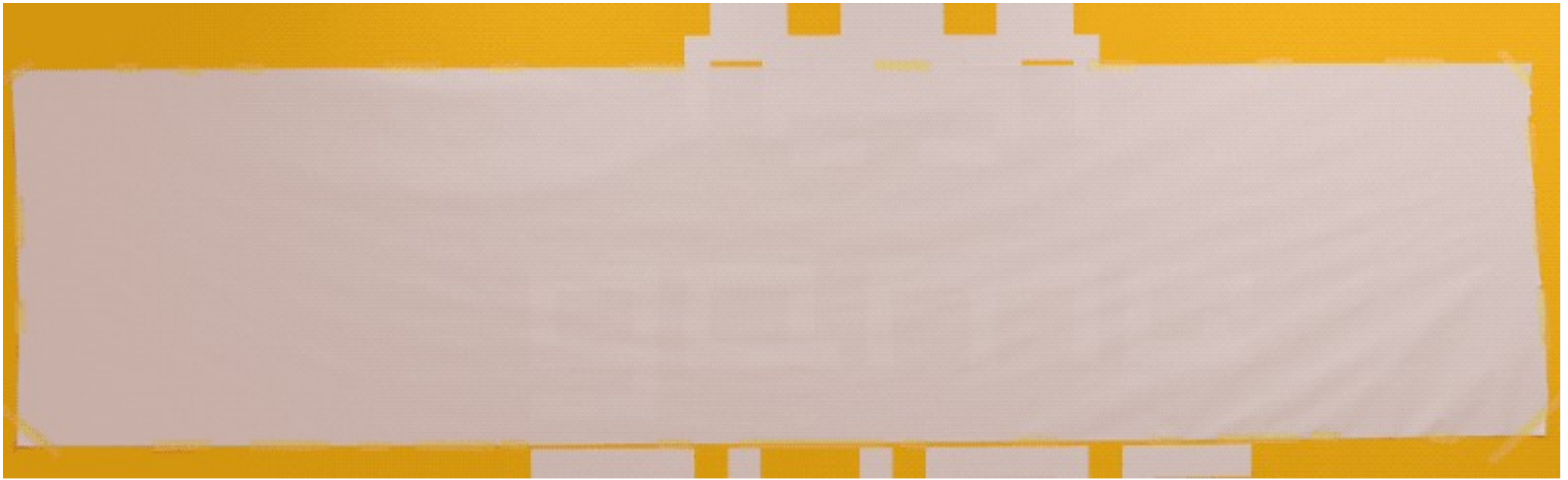
Component implementation details

C4 Model

- https://c4model.com/
- Visualise, document and explore your software architecture – Simon Brown
- Visualising software architecture with the C4 model – Simon Brown

Event Storming

Event Storming

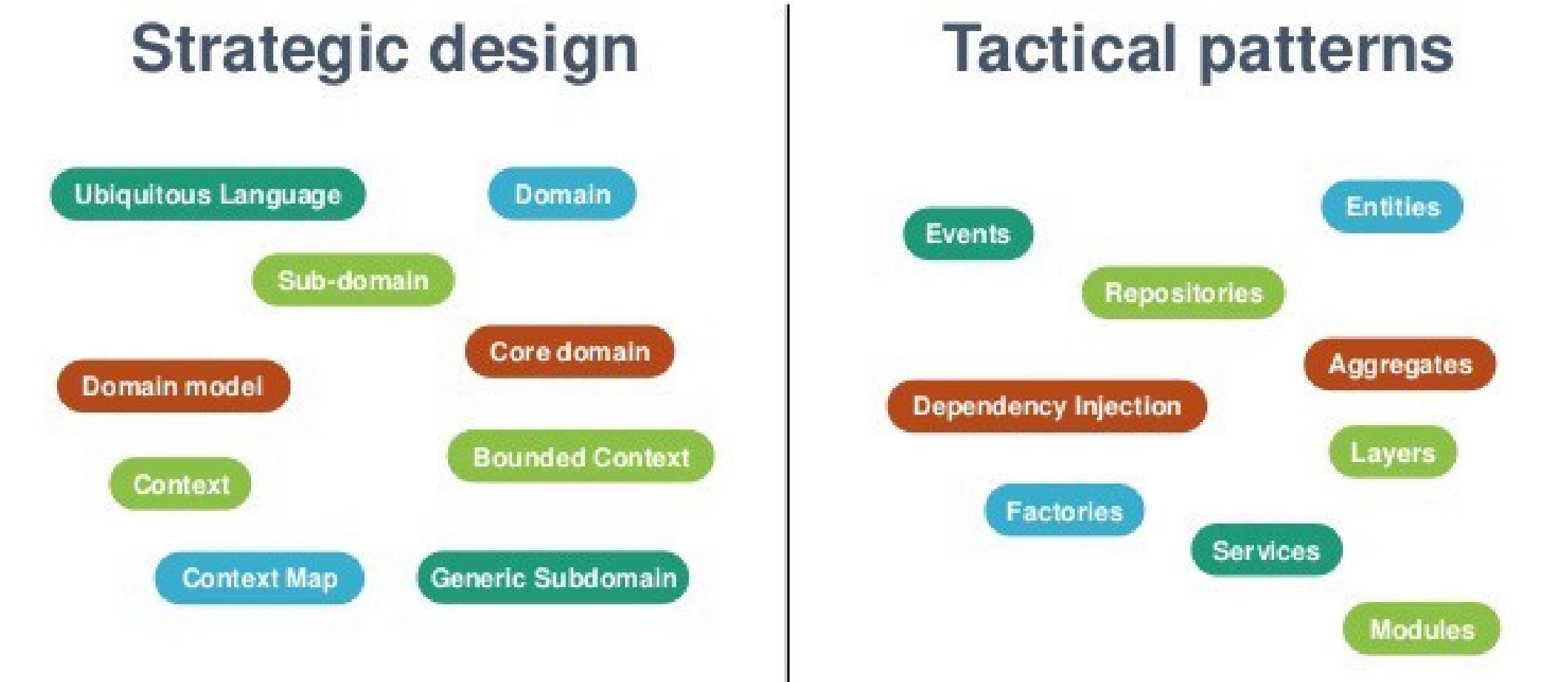


Source:
https://raw.githubusercontent.com/mariuszgil/awesome-eventstorming/master/assets/timelapses/timelapse-1.gif

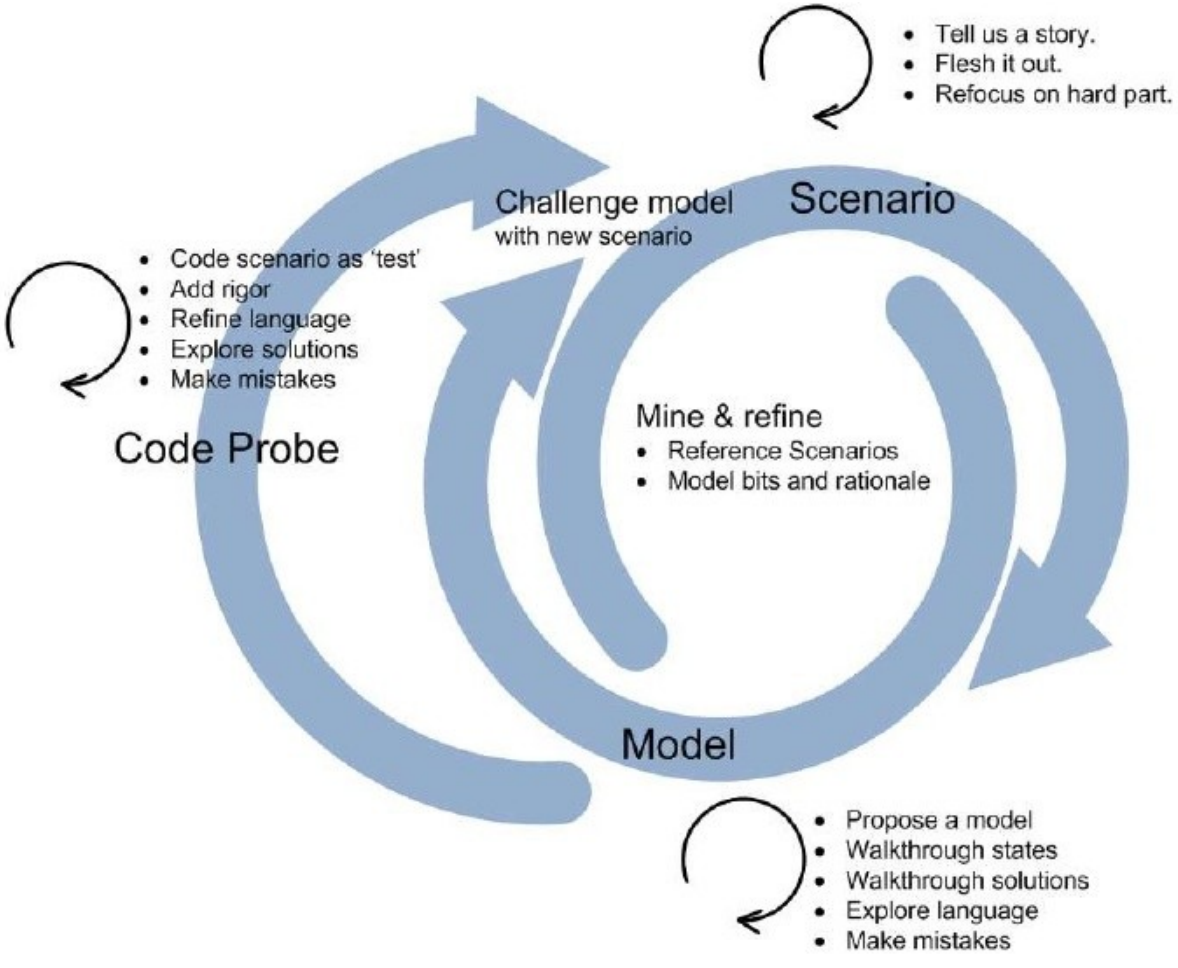
Event Storming

- Event Storming book – Alberto Brandolini
- Github – Awesome Event Storming
- Discovering unknown domain with Event Storming – Mariusz Gil

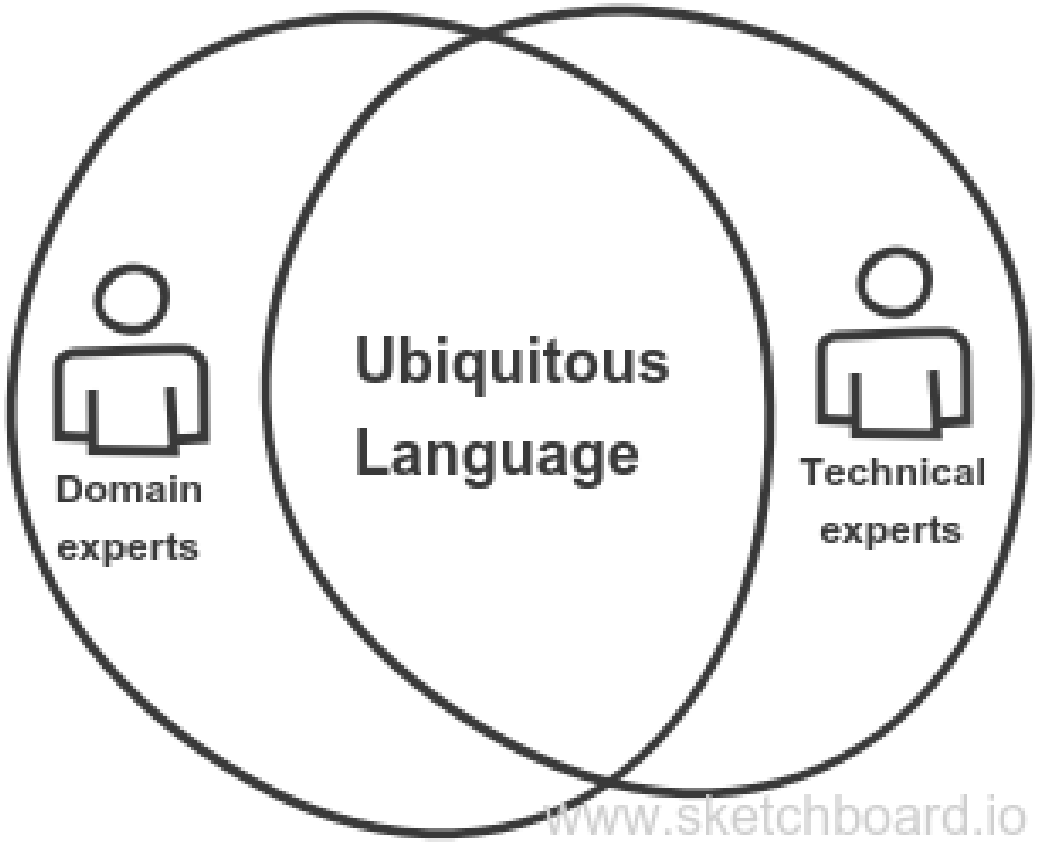
Domain Driven Design



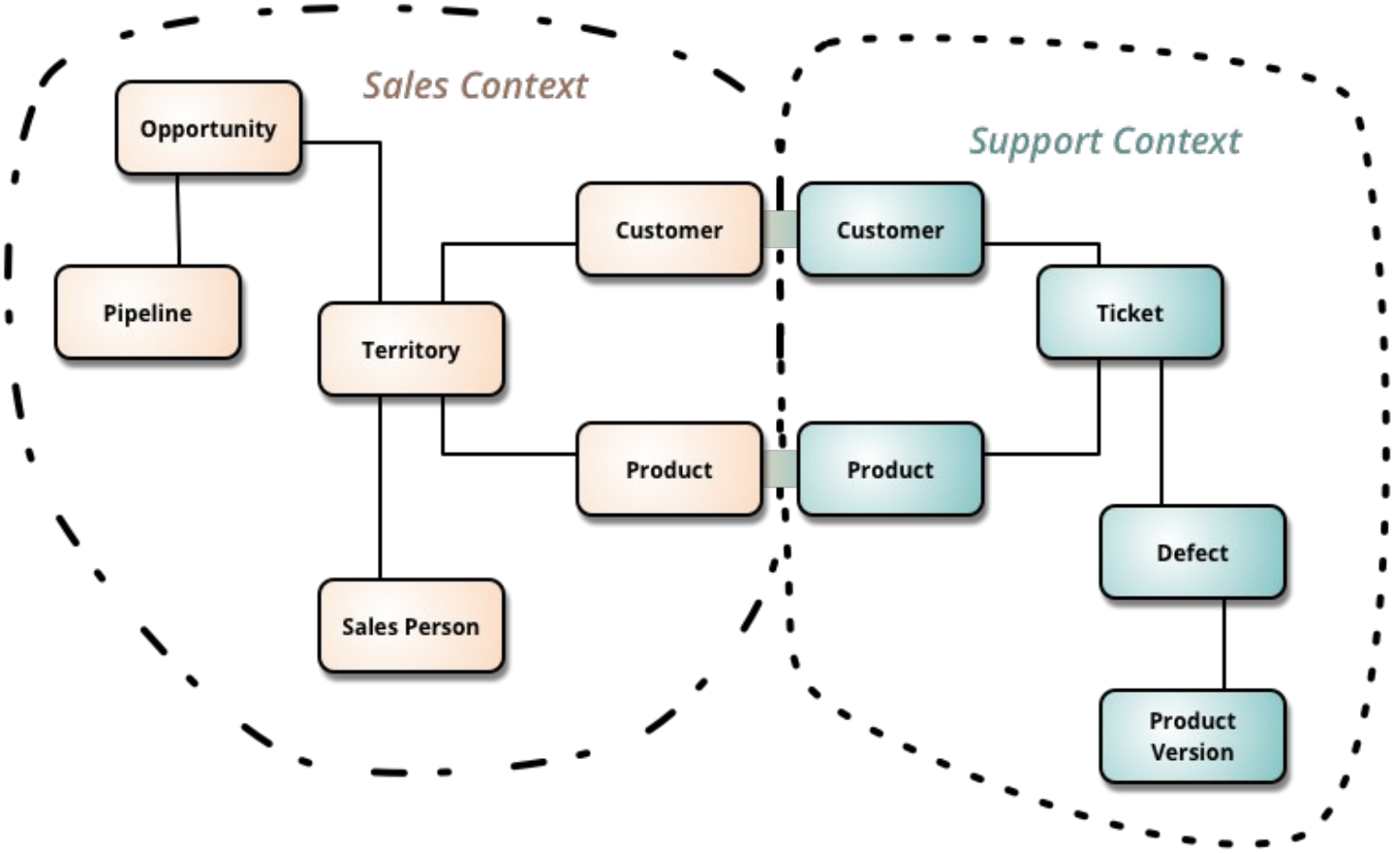
DDD – Design Whirlpool



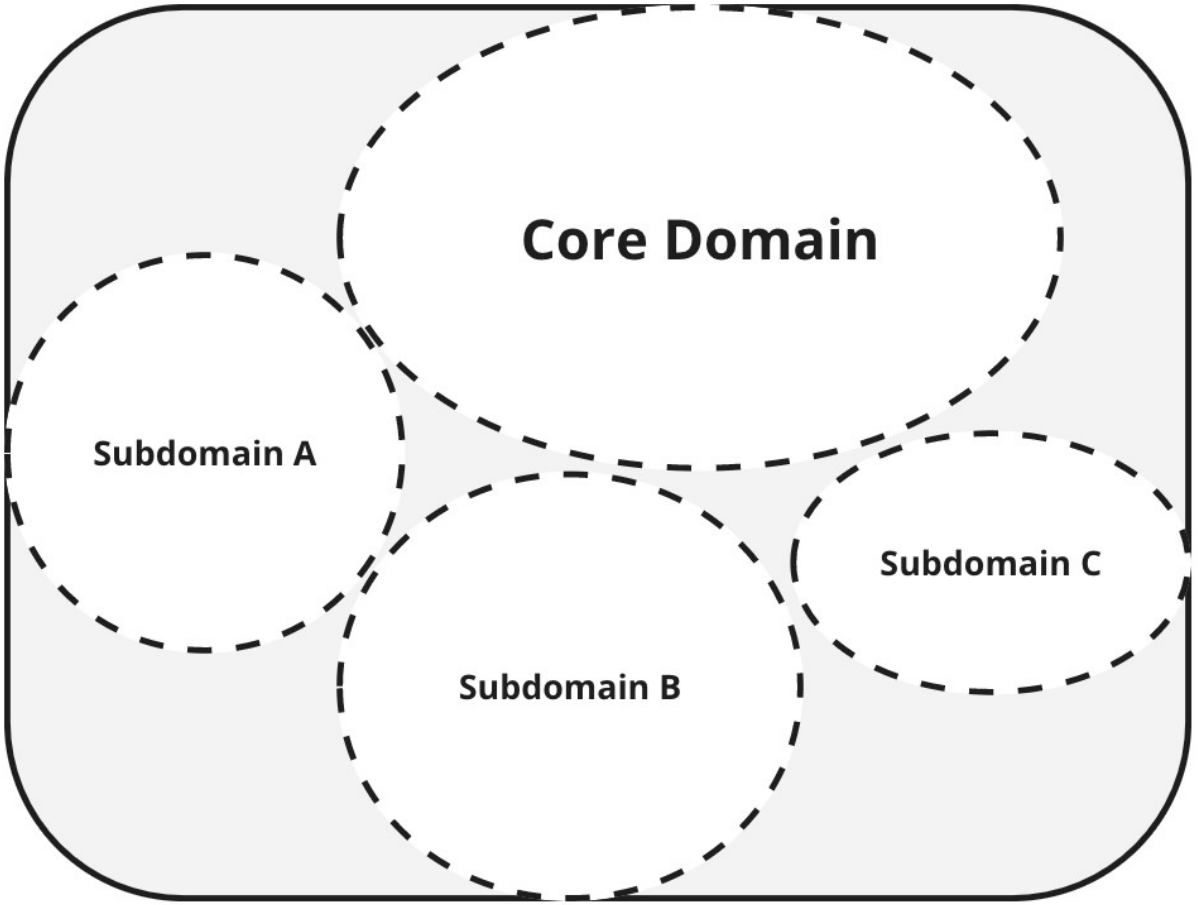
DDD – Ubiquituous Language

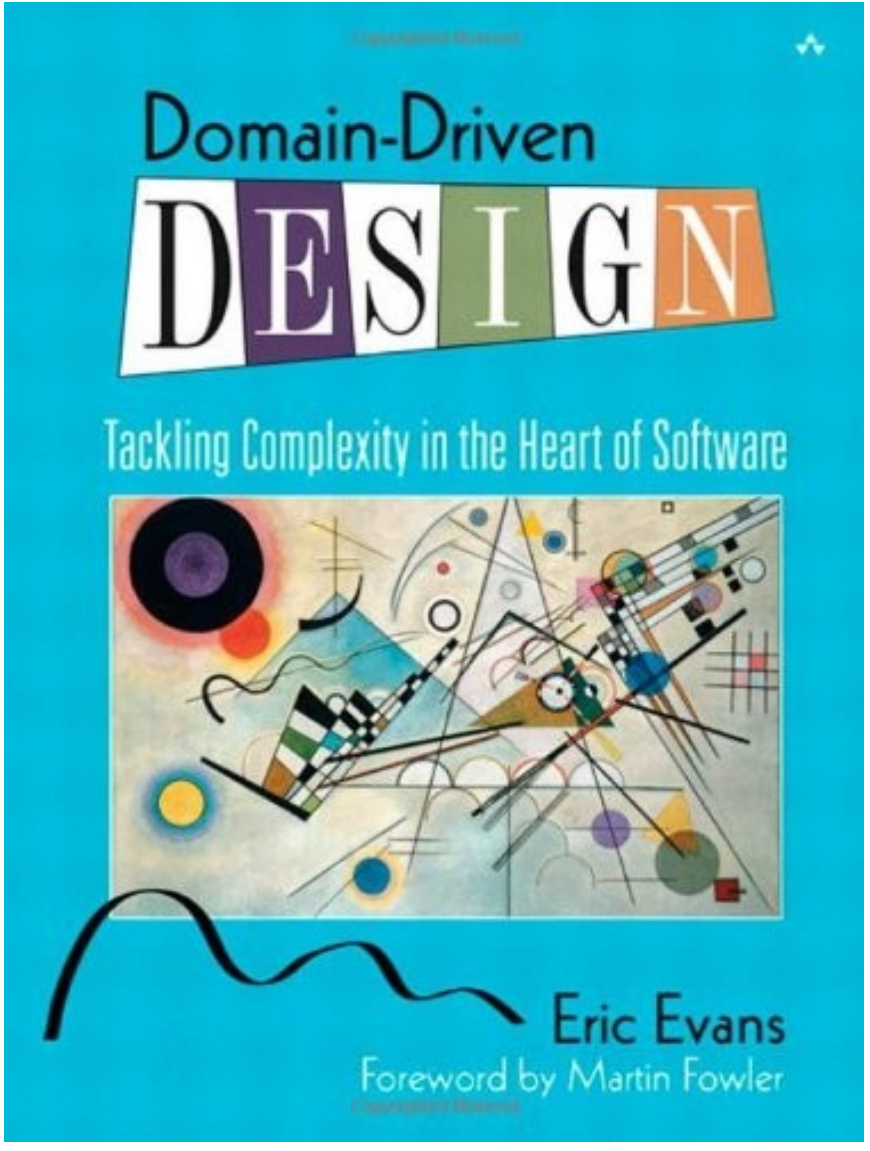


DDD – bounded context

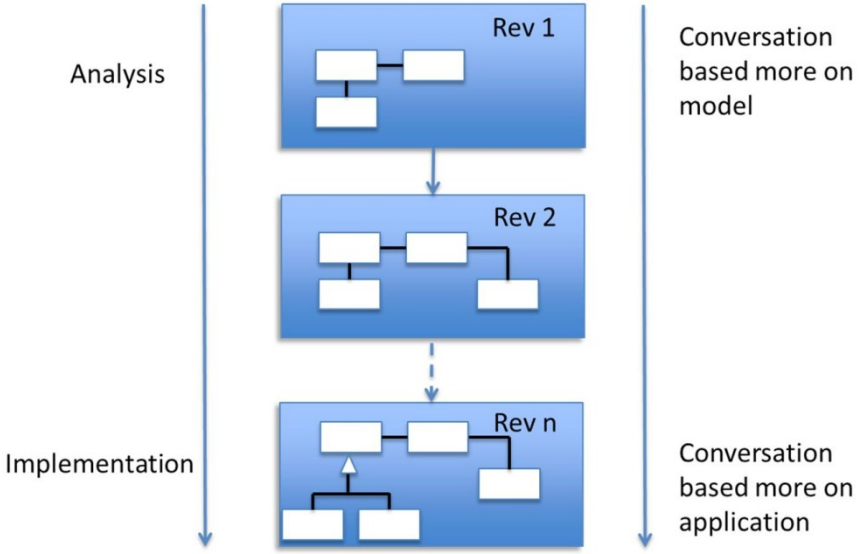


DDD – Core domain




Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler

DDD in safety-critical



Source:
https://www.researchgate.net/publication/308089490_Experience_from_integrating_Domain_Driven_Software_System_Design_into_a_Systems_Engineering_Organization


VARIANT A
VARIANT B


The End