COM S 362
Object-Oriented Analysis & Design

OOAD for short

Analysis and Design

- Analysis is the process of *understanding* a problem and its potential solutions.
- Design is the act of choosing a structure within which to organize the components of the solution.

Why design?

- A correct, but uninformative answer is that this is a required class, because design is a core programming skill.
- To really understand why design is important, we need to understand
  - what programmers *do,*
  - what challenges programmers face, and
  - how good design changes the challenge or contributes to the work.

What do Developers Really Do?

- What percent of time are developers creating new code vs. maintaining existing code?
- What percent of time do developers spend reading and trying to understand existing code?

Quiz

- What percent of time is writing new code?

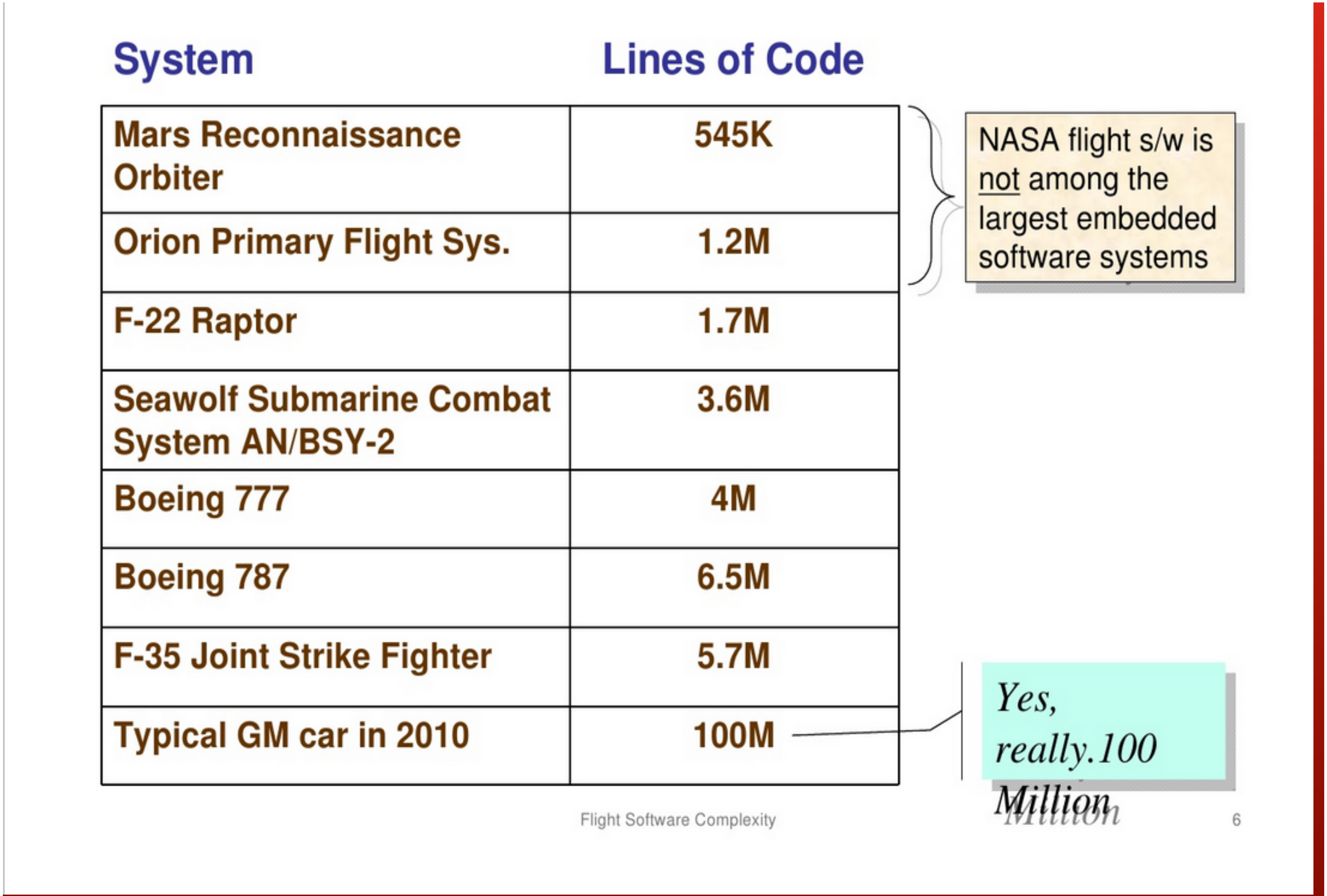
The Reality of Programming

- Programmers spend 80% of their time modifying existing code.
- Software systems are enormous.
- Software systems are enormously complex.
- The rate of change in software systems is rapidly accelerating.
- The risk of failure in large projects is significant.
- Software structure (design) often determines the difficulty of making a particular change.

Analysis and Design skills are critically important to programming success.

How Hard can it Be?

- What is the largest project you have completed?
- How many lines of code are in the software on a car?



System	Lines of Code
Mars Reconnaissance Orbiter	545K
Orion Primary Flight Sys.	1.2M
F-22 Raptor	1.7M
Seawolf Submarine Combat System AN/BSY-2	3.6M
Boeing 777	4M
Boeing 787	6.5M
F-35 Joint Strike Fighter	5.7M
Typical GM car in 2010	100M

NASA flight s/w is not among the largest embedded software systems

Yes, really.100 Million

Flight Software Complexity

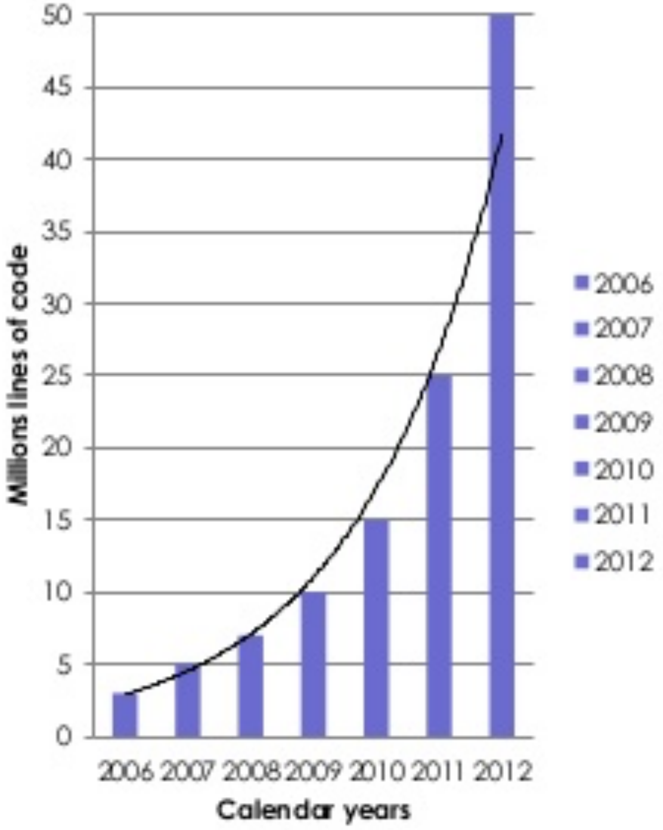
6

From: https://www.slideshare.net/NASAPMC/dvorakdan visure

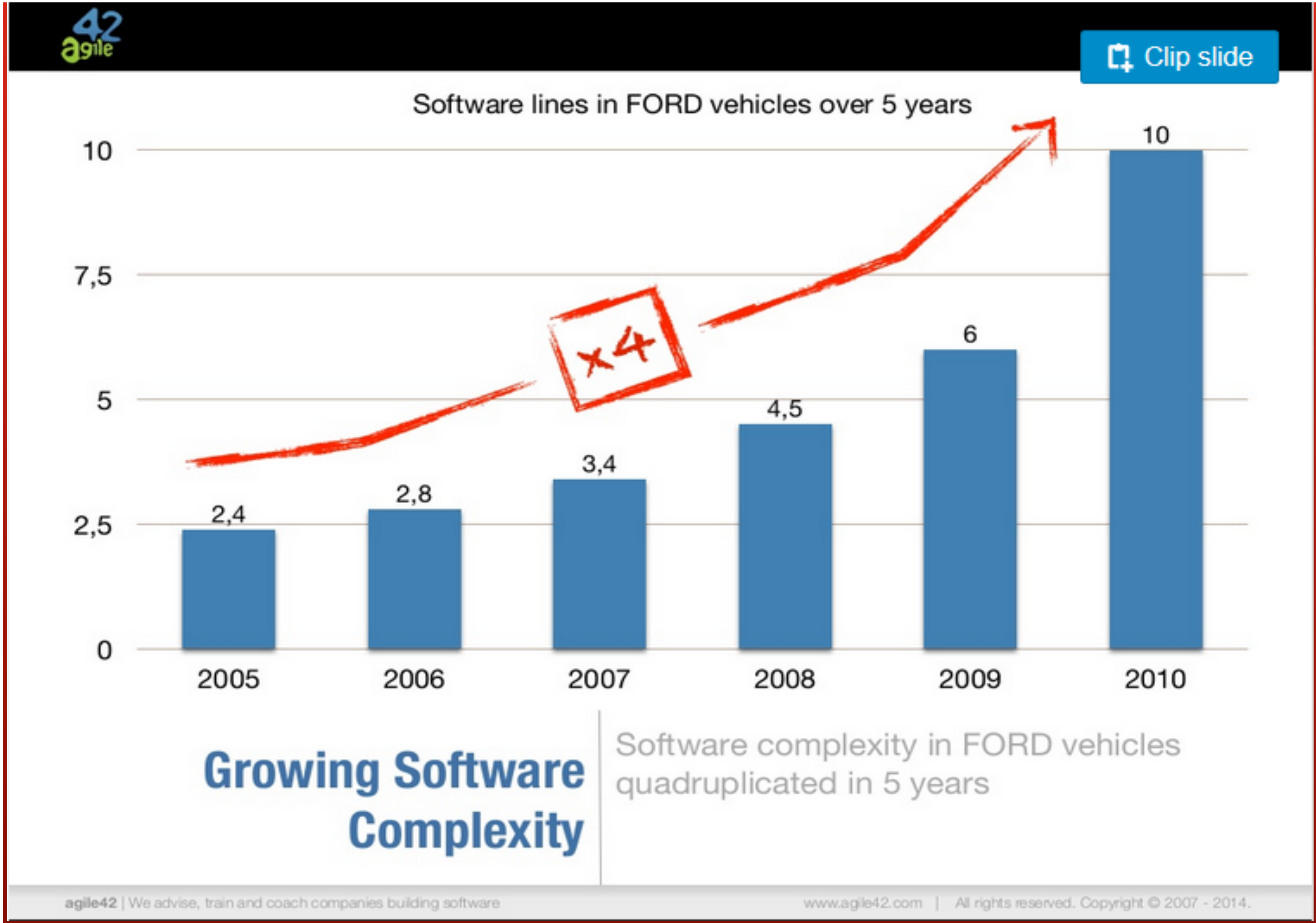
Our View of the Market: Market Pains 1

Growing Complexity of Software and Product Variants

- **Key issues:**
  - Relative growth of software, increase of 33% compared to mechanics and electronics
  - Complexity of product lines and large-scale systems
- **Examples:**
  - Aircraft carrier, 1,000,000 requirements
  - Hospital information system: 90,000 requirements
  - Train control system: 14,000 contract requirements



•Ford Motor Company's Prediction of Growth of Software Content in Automobile embedded software.
•CPDA, The Requirements Engineering Process - Panel Discussion of Critical Issues in Requirements Management
•Nationale Roadmap Embedded Systems, ZVEI, www.zvei.org, 2010
•Requirements Engineering: A Roadmap, Bashar Nuseibeh & Steve Easterbrook, 2000



Growing Software Complexity

Software complexity in FORD vehicles quadruplicated in 5 years

agile42 | We advise, train and coach companies building software

www.agile42.com |

From: https://www.slideshare.net/tumma72/from-a-product-vision-to-a-running-software-and-back-again-and-agile-coach-story-from-andrea-tomasini

Software Challenges

Programming is code evolution

60%-80% of development is maintenance/evolution of existing products.

- How do developers understand million line programs well enough to reliably modify or extend them?
- How do we preserve design integrity over the life of a product? How do future maintainers acquire an understanding of the designer’s intent?

A 1992 Bell Labs study found that new project members spend 60%-80% of their time understanding code.[1]

A 2007 survey of 780 Microsoft developers (Cherubini) found that 95% agreed that understanding existing code is a significant part of their job.[1]

[1] Reported in http://blog.architexa.com/2010/05/no-you-are-not-dumb-programmers-do-spend-a-lot-of-time-understanding-code/

What About This?

```java
import java.util.Scanner;

public class TicTacToe {

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        char[] mark ={'X','O'};
        char[] b = {' ',' ',' ',' ',' ',' ',' ',' ',' '};
        int move = 0;
        for (int turn = 0; turn < 9; turn ++){
            //draw the board
            System.out.println("   "+b[0]+" | "+b[1]+ " | " +b[2]);
            System.out.println("  -----------");
            System.out.println("   "+b[3]+" | "+b[4]+ " | " +b[5]);
            System.out.println("  -----------");
            System.out.println("   "+b[6]+" | "+b[7]+ " | " +b[8]);
            // get next move
            System.out.format("\nPlayer %d select cell (1 - 9)%n", (turn%2)+1);
            move = scan.nextInt();
            if (b[move-1] != ' ') {
                throw new IllegalStateException("That cell is taken! you lose");
            }
            b[move-1] = mark[turn%2];
        }
        System.out.println("Game Over");
        scan.close();
    }
}
```

What is design?

Design assumes we know the problem and that we believe the problem to be solvable and that we intend to solve it.

Given that, Design answers the question: what code should we write? How should it be organized? What is our strategy for ensuring it meets our business objectives?

Design is about *where* each piece of implementation will live and why. Design is about how each design decision is expressed and why.

What Design is *not*

- Design is not style, not readability.
- Design is not about *when* code will be executed, or about whether it the code behaves correctly.
- Design is not about efficiency nor aesthetics.
- Design is not algorithms or computability – not time complexity nor decidability, or whether one problem can be reduced to another, etc.
- Design is not how to write code – not about language syntax or semantics, nor even about good style.
- Design is not about how to manage a project – whether you should work iteratively or sequentially, co-located or distributed.

The Goals of Good Design

- To facilitate change
- To facilitate understanding
- To facilitate testing
- To facilitate development

All through a focus on creating structure within the code – often this code structure mirrors the designer’s understanding of the problem’s intrinsic structure.