How to Be a
Good Software Engineer

not only a Programmer

Dr. Umit Yalcinalp

Umit Yalcinalp

- Software Architect, Researcher
  - Adobe; Oracle;Sun Microsystems; SAP; ...
- Technologist
  - Java, XML, Web Standards; Cloud Computing; Infrastructure; Metaprogramming, ...
- Tech Education and Evangelism
  - Salesforce; SAP;
  - Mills College;
  - Conferences
- Startup

What is this talk about

- Coding is fashionable ! So what is the problem?
  “*Be a software engineer in 6 weeks!*”
- Some Distilled Lessons learned in 20+ yrs
- How to stay alive and thrive
- What to look for in a working environment

Lets look at a typical problem

Problem:

There are student records with following information:
`id, firstname, lastname, street, city, state, zipcode, year_started, status.`

Data:

```
10, mary, jones, 24 Bellweather street, San Francisco, California, 94114, 2014, G
130, bill, pecke, 1000 Morane Street Apt 14, Palo Alto, California, 94303, 2013, G
132, jack, drumms, 343 Marylane, Boston, MA, 23030, 2010, S
```

We need them sorted by last name.

Discussion

How about this?

```
sort -f -b –k=3 –t=“,” studentfile
```

You can get a lot done in Unix/Linux/Shell scripts

(Yep!) Many Approaches:

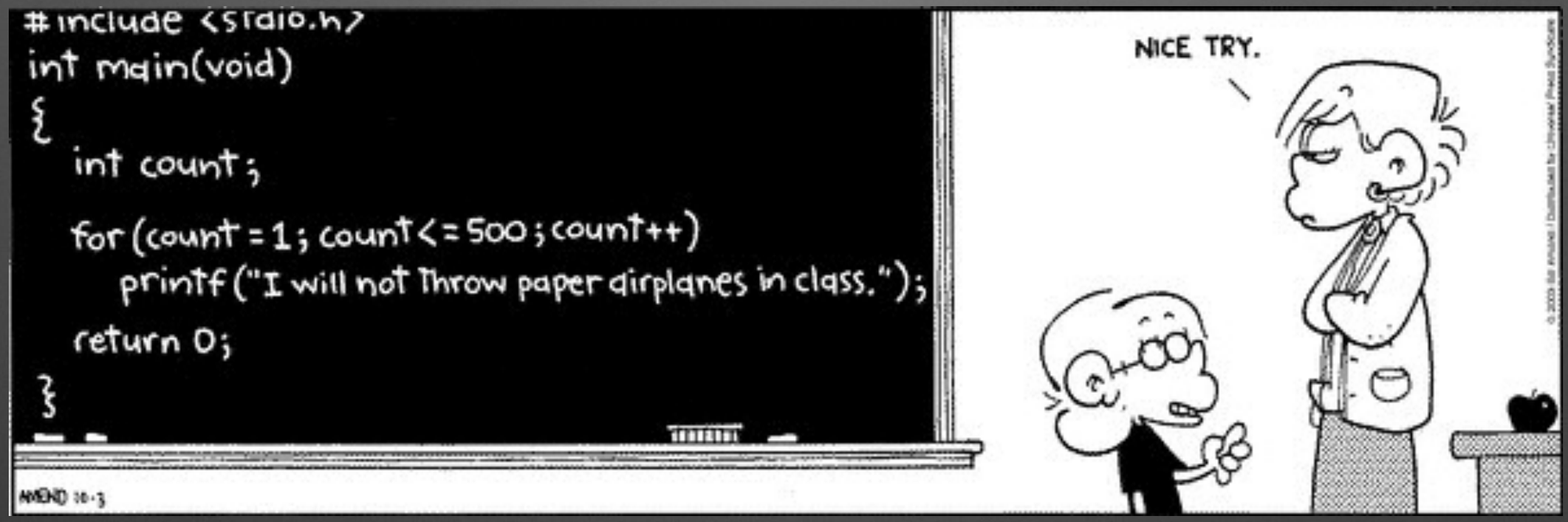
- Write a program in my favorite dev environment (Java, Python,…)
- Use a sorting library I found in open source
- Write a UNIX shell script that uses a sort utility
- Import them in a database & use the database language (SQL, …)
- Develop a web service that will expose student records in different ways
- Develop a system that will enable us to examine all the student records with a nice graphical interface
- …

Before writing Code

- What are the use cases? Are there more than one?
- How and where will this be used?
  - Ways of input/output, Size of input/output
  - Are there restrictions on usage?
  - What should this be?
    Some Function in a library
    Do we need an exposed API
    Should it be a Service, …

What is the problem we are solving?

Photo: http://www.sheknows.com/parenting/articles/804453/why-mommy-why


#include <stdio.h>
int main(void)
{
int count;
for (count = 1; count <= 500; count++)
printf ("I will not throw paper airplanes in class.");
return 0;
}
NICE TRY.

Rule #1

A Solution is

as good as

the assumptions you make

&

use cases

you have addressed

When you are interviewing

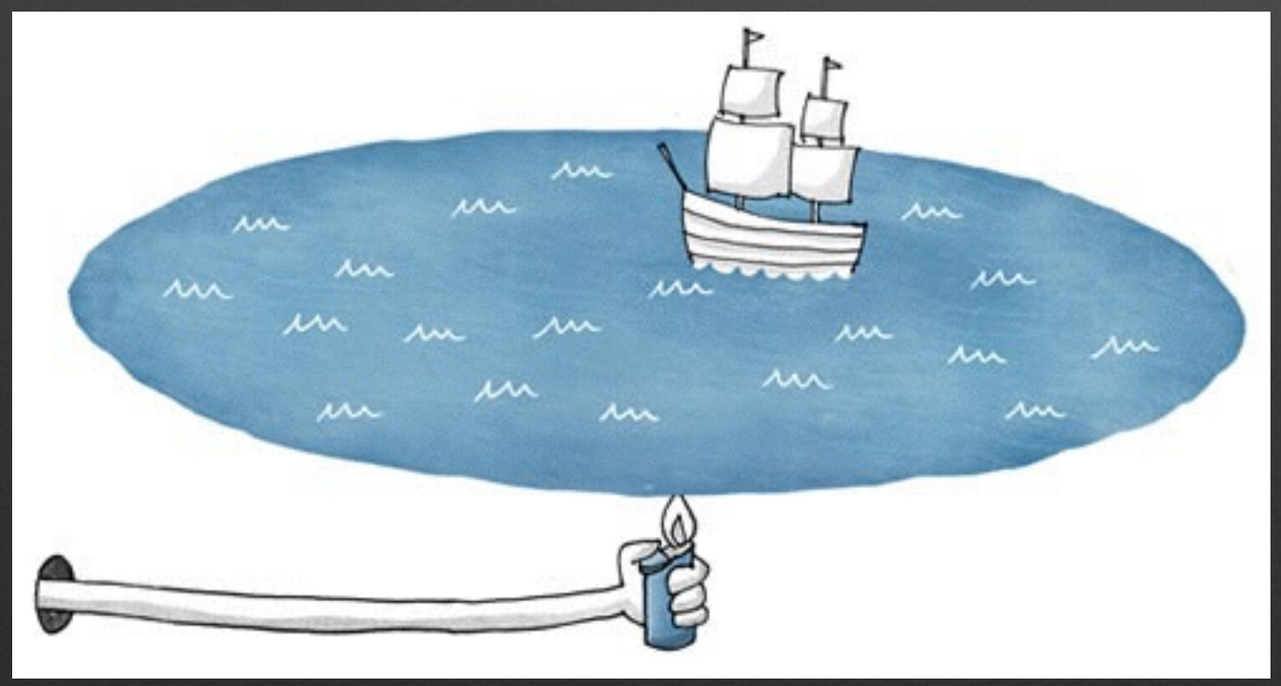
- Know your algorithms, complexity etc.
- You may illustrate more than one approach and explain why
- State your assumptions explicitly
- Always ask questions to understand the use case(s)

When you are developing

- Always ask questions to understand the use case(s)
- Always document assumptions and use cases
  - Use your dev environments coding practices
    - Don’t write cryptic code
      not `x = x+y` but `miles = miles + bonus`
- Always develop tests for your code
- Integrate with or contribute to end2end use case testing

Methodology Issues

- Keep it too small —> Bottom up
- Boil the ocean —> Top down



Be #SmartAgile

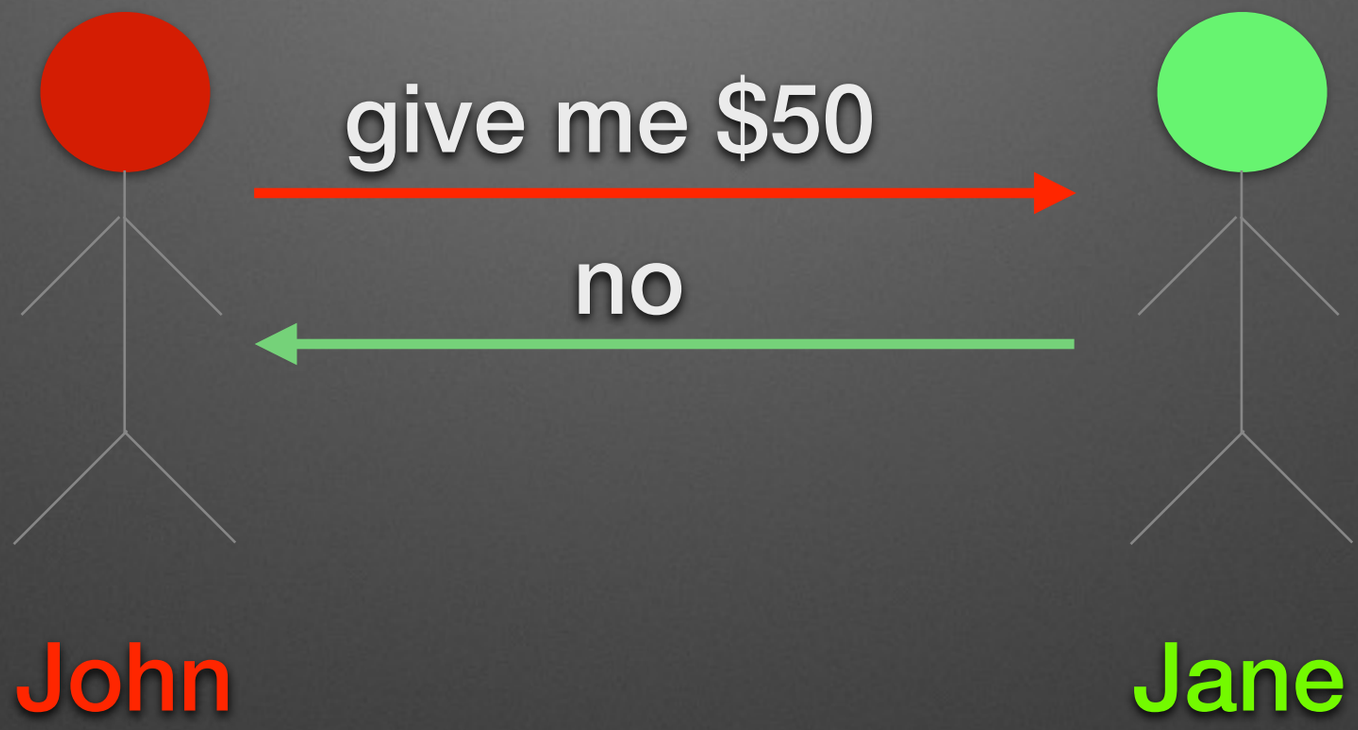
- Start Small
- Think Big
- Iterate, Iterate, Iterate
- Change Course with new findings at each step

(also lean principles)

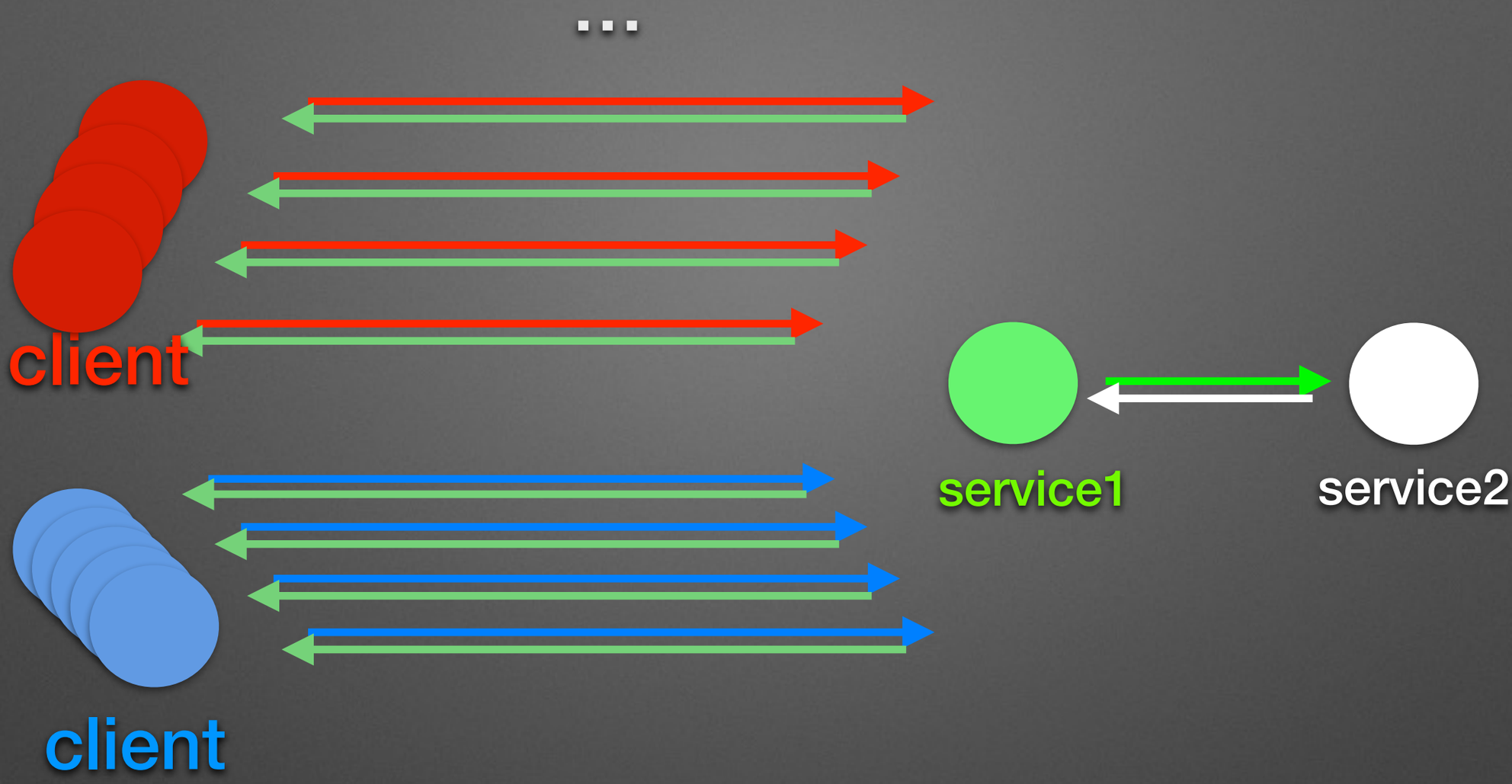
Rule #2:

Think Systems and Architecture not App

RESOURCE
HTML
HTTP
REACT.JS
PYTHON
Servlet
JSON
XML
JAVA
ANGULAR
Javascript
Jade
TAGLIBRARY
DOM
NODE.JS
JEE
EJB
JQUERY
MONGO
Angular.js
JADE
HTML
HTTP
JS
SQL
Mongo
TAGLIBRARY
REACT JS
JAVASCRIPT
EJB
JSON
SERVLET
JQUERY
JSON
MONGO
RESOURCE
JAVA
Angular.js


give me $50
no
John
Jane

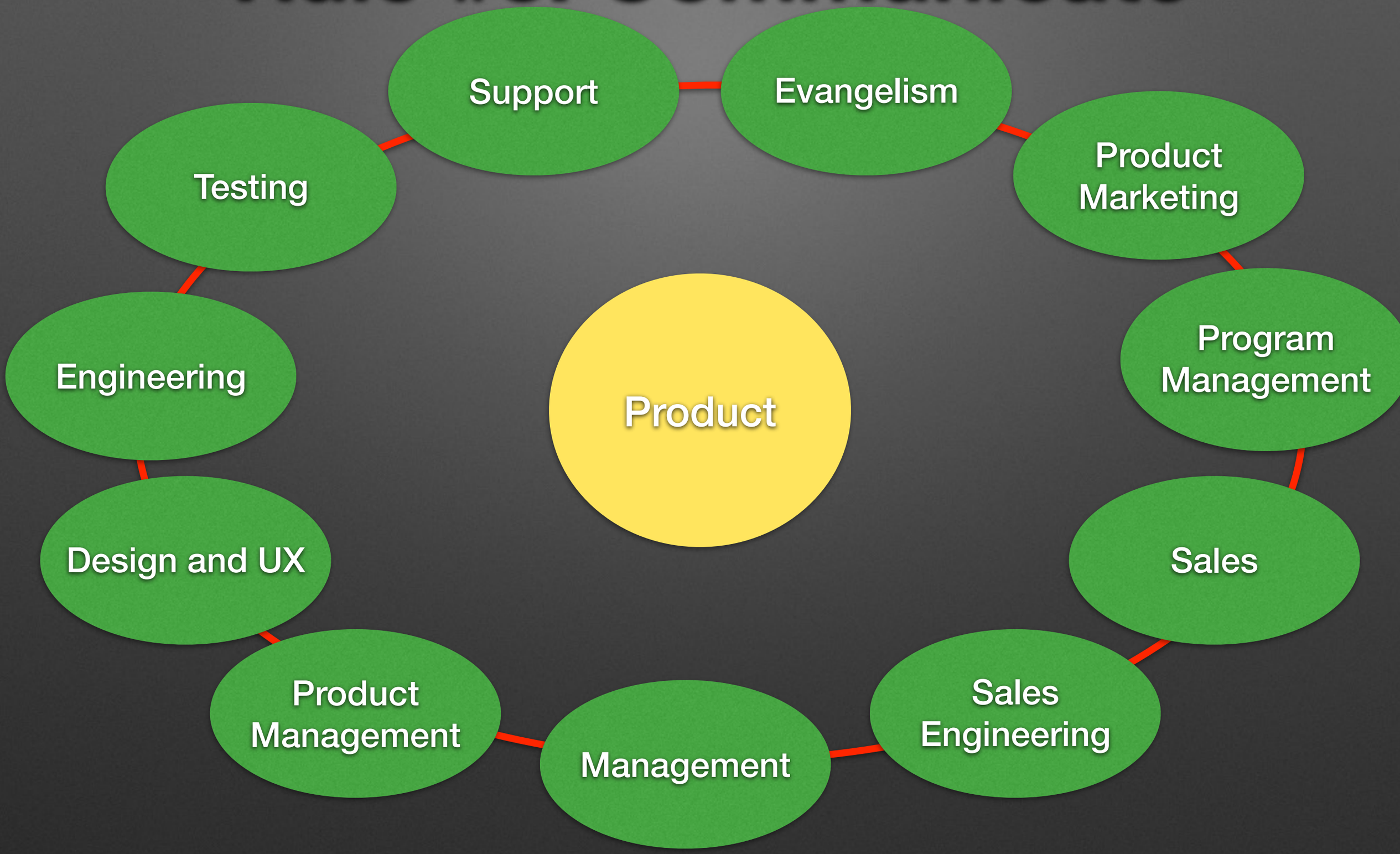

give me $50
who are you
Turkce konusun!
John
Jane



- Protocols
- Types of clients
- Responsibilities of each service and component in each service = Stack Architecture
- Security

Build on what you learn in this course…

Rule #3: Communicate



Strategy Issues

- Building things no one wants
- Building with bad software design

Rule #4: self.update();

- static final boolean CHANGE = true;
- Spend 1 hr each day minimum to learn something new
- Free Options:
  - Meetups and hackathons
    (Amazon, salesforce.com, Google, …)
  - Subscribe to articles (InfoQ)
  - Seminars in your school and elsewhere

Thank you!

Last words: Always think of how with why