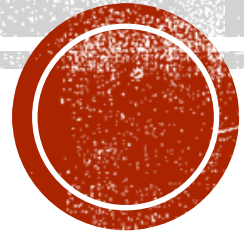
РАЗВОЈ СОФТВЕРА 2

Софтверске грешке



ГРЕШКЕ, ОТКАЗИ, КВАРОВИ

ERRORS AND HUMANS

"To err is human; *to forgive, divine*."

" …but to really foul up requires a computer.."

"From novice to the best, from the inexperienced to the experienced, from young to old, all designers/programmers make errors."

ERROR: A DEFINITION

“mistake, error, fault”

“ a wrong action attributable to bad judgment or ignorance or inattention; "he made a bad mistake"; "she was quick to point out my errors"; "I could understand his English in spite of his grammatical faults"

ERROR, FAULT, FAILURE

Analyst/Designer/Programmer makes a mistake.

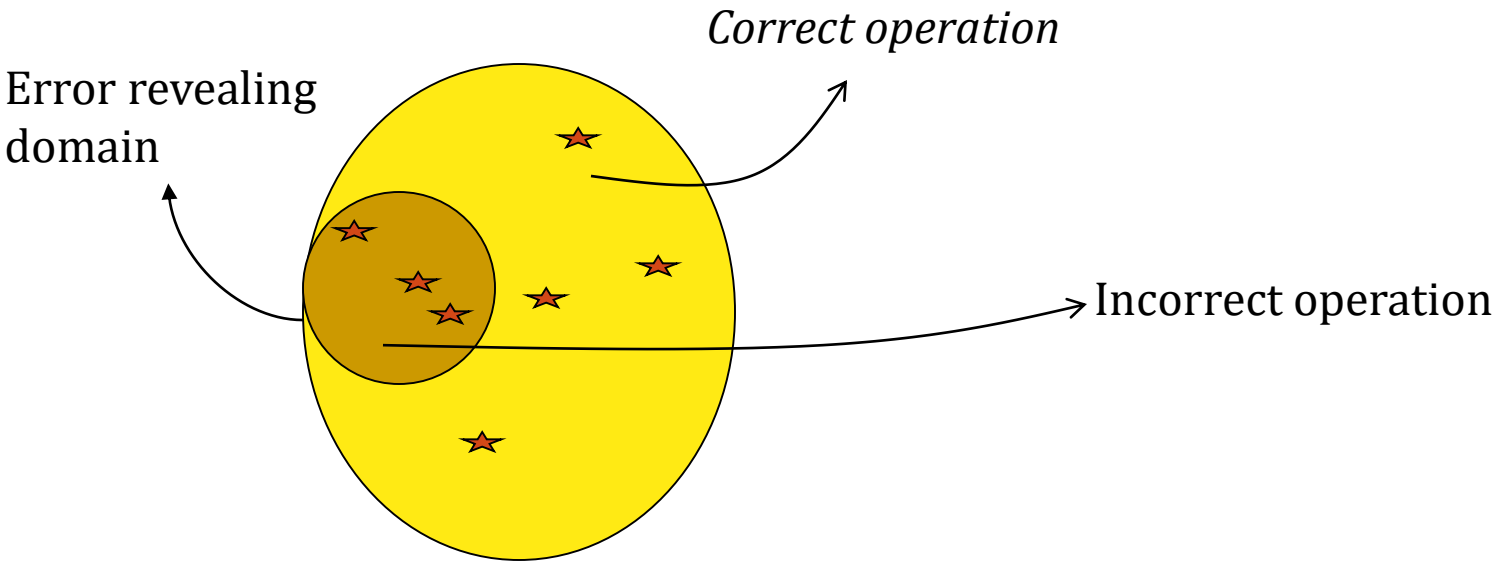
Fault appears in the program.

Fault remains undetected during testing.

The program fails during execution i.e. it behaves unexpectedly.

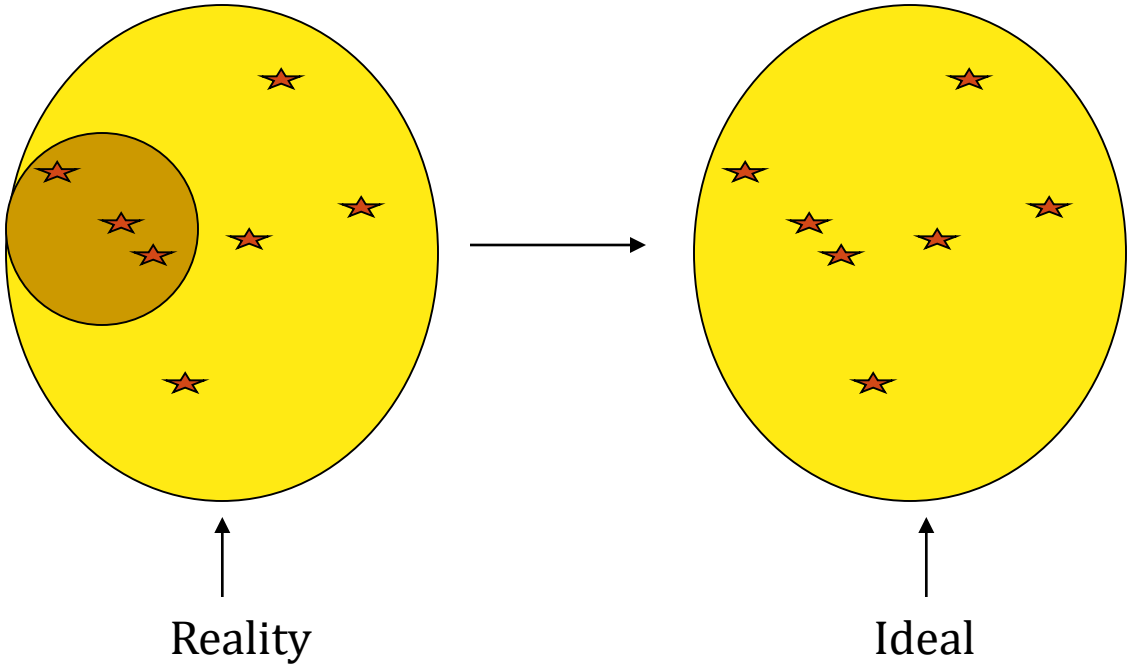
ERRORS AND INPUT

Set of inputs, also known as the Input domain



FINDING AND FIXING ERRORS

Goal of testing and debugging: Reduce the set of failure causing inputs to null.



IMPACT OF ERRORS

- Funny: Will likely generate laughter.
- Inconvenience: May require machine reboot.
- Disaster: May cause loss of property.
- Fatal: May cause death.

ПРИМЕРИ „СМЕШНИХ“ ГРЕШКИ

FUNNY: LA TOSCA (GIACOMO PUCCINI)

- “The opera Tosca debuted just over one hundred years ago, at the Teatro Constanzi in Rome on January 14, 1900. Soon after its premiere, it became one of the most popular operas in the repertoire, and it remains so to this day.”

FUNNY: CANDLE BLOWING

- “It was the candelabra that played a prominent role in a San Diego performance of Tosca in 1956. The script called for Tosca to blow out the four candles in the candelabra before dramatically placing a candle on either side of Scarpia and a crucifix on his breast and exiting the stage.

- With modern fireproofing the fire-risk is usually considered small enough to permit the use of real candles.”

FUNNY: BLOW ORDER MISMATCH

- “In San Diego, however, the candles were not only electric, but the order of their going out was fixed on a computer tape along with all the rest of the lighting cues.

- The tape obeyed the stage manager's signal and snuffed the candles exactly as Tosca blew them out - except that on this occasion the programming was wrong and it blew them out in a different order from hers.”

FUNNY: OOPS!

- She blew to the right, the candle on the left went out, she blew the back one, the one in front went out!

- To further confuse the situation, as she began E avanti a lui tremava tutta Roma (And before him all Rome trembled), the electronic bleep for the curtain arrived too soon and the curtain shut with furious speed before she had finished.

ПРИМЕРИ ГРЕШКИ СА МАЊИМ ПОСЛЕДИЦАМА

INCONVENIENCES: MINOR AND MAJOR

- A nuclear reactor was shut down because a single line of code was coded as X = Y instead of X=ABS (Y) i.e. the absolute value of Y irrespective of whether Y was positive or negative.
- On July 1-2, 1991, computer-software collapses in telephone switching stations disrupted service in Washington DC, Pittsburgh, Los Angeles and San Francisco.
- In May 1992, Pepsi fan a promotion in the Philippines. It told customers they could win a million pesos (approx. $40,000) if they bought a bottle of Pepsi and found number 349 stamped on the underside of the bottle cap. Unfortunately, due to a software error, 800,000 bottle caps were produced with number 349 instead of one, which was an equivalent of $42 billion in prize money. It cost the company dearly as some people pursued their claims through the courts and Pepsi paid out millions of dollars in compensation.

INCONVENIENCES: MINOR AND MAJOR

- February 1994. Chemical Bank managed to allow $15 million to be withdrawn incorrectly from 100,000 accounts - a single line error in the program caused every ATM on their network to process the transaction twice
- The Mars Climate Orbiter crashed in September 1999 because of a "silly mistake": wrong units in a program.
- A cat was registered as a voter to demonstrate risks (no photograph required).
- Blue Cross of Wisconsin installed a new $200m claims processing system - it sent out $60 million in unwarranted and duplicate payments. Also, when a data entry clerk typed 'none' in the town field the system sent hundreds of checks to the non-existent town of 'NONE' in Wisconsin

ПРИМЕРИ ГРЕШКИ СА ВЕЛИКИМ ПОСЛЕДИЦАМА

DISASTER: ROCKET LAUNCH

- French rocket Ariane 501 was scheduled to launch on the morning of June 4, 1996, from the launch site in Kourou, French Guiana.

- Failure of the launch due to the un-manned rocket exploding after 42 seconds from the time of the launch.

DISASTER: THE ERROR

- An internal software exception was caused during execution of a data conversion from a 64-bit floating-point number to a 16-bit signed integer value.
- The value of the floating-point number was greater than what could be represented by a 16-bit signed integer (e.g. 43,445).
- The data conversion instructions were not protected from causing operand errors.

DISASTER: INADEQUATE TESTING

- “The guidance system for Ariane 5 was an upgrade from Ariane 4. But Ariane 5 had a different trajectory which pushed one of the numerical parameters out of bounds.”
- “ This would have been caught in testing, but only if the testing specification were properly upgraded to work for Araine 5. That didn't happen although the specifications and development processes are at the same level as NASA uses. They still failed.”

ПРИМЕРИ ГРЕШКИ СА ФАТАЛНИМ ПОСЛЕДИЦАМА

FATAL: THERAC-25

“The Therac-25 was a computerized radiation therapy machine.”

“The Therac-20, a predecessor of the Therac-25, employed independent protective circuits and mechanical interlocks to protect against overdose. The Therac-25 relied more heavily on software.”

FATAL: NO HARDWARE INTERLOCKS

“The Therac-25 supported a multitasking environment, and the software allowed concurrent access to shared data. This precarious implementation caused program failure under certain (race) conditions.”

FATAL: OVERDOSE AND DEATHS

“The machine massively overdosed patients at least six times between June 1985 and January 1987. Each overdose was several times the normal therapeutic dose and resulted in the patient's severe injury or even death.”

ЗАКЉУЧЦИ

CONSIDERED PAPERS

- Robert Charette: Why Software Fails, IEEE Spectrum, September 2005, https://spectrum.ieee.org/why-software-fails

- Robert Charette, Joshua Romero: Lessons From a Decade of IT Failures, IEEE Spectrum, October 2015, https://spectrum.ieee.org/lessons-from-a-decade-of-it-failures

LESSONS

Lesson 1. The Staggering Impact of IT Systems Gone Wrong

- The world has relied on large-scale IT systems for decades, but we still haven't learned how to prevent and avoid major glitches and failures.
- When we looked back over the decade, several themes became impossible to ignore:
  - Modernizing IT Systems Is Difficult and Expensive
  - Digitizing Health Records Is Difficult and Expensive
  - Banks Rely on Unreliable Technology
  - Even Brief Stock-Exchange Glitches Are Costly
  - Even Brief Air-Travel Glitches Are Costly

LESSONS

Lesson 2: Overcomplexifying, Underdelivering

- It is hard to build IT systems, but it's arguably even more difficult to maintain them properly over time
- Very often, decades of neglect have resulted in a tangled mess of poorly understood and poorly implemented systems that limit operational effectiveness and efficiency
- In the past decade, we've seen numerous attempts to combine the functionality of such legacy systems into a single modern replacement system
- That's easier said than done

LESSONS

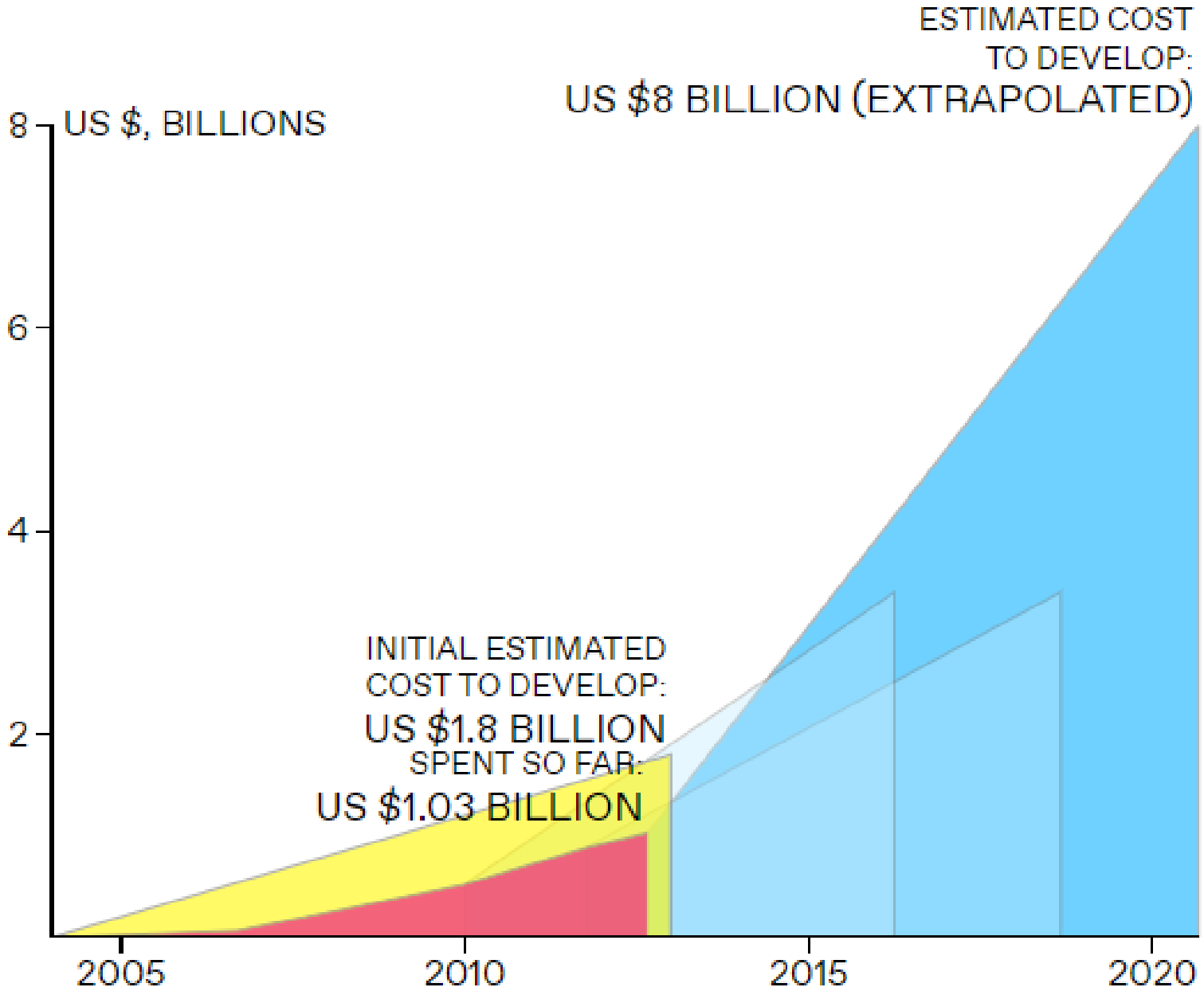
Lesson 3: The Life Cycles of Failed Projects

- IT projects rarely fail all at once
- Instead, these failures tend to snowball, growing larger and more hopeless as time goes on
- Along the way, the definition of success is prone to mutation, as deadlines get pushed back and budgets increase
- That's how a project can launch "ahead of schedule" even if it's more than three years late.

LESSONS

Lesson 3:

U.S. Air Force's Expeditionary Combat Support System



LESSONS

Lesson 4: The IT Failure Blame Game

- When IT systems fail, there's always a reason.
- If you dig deep enough, at the root of any problem are human decisions: sloppy code, insufficient testing, poorly understood dependencies, and incorrect assumptions.
- Yet when we read about (and report on) failures, the language we use tends to assign blame to inanimate technology that can't defend itself or get fired.

LESSONS

The cadavers of dead IT projects are buried under mounds of cash

One of them:

https://spectrum.ieee.org/uk-firecontrol-project-finally-axed

НАПОМЕНА

Највећи део материјала ове презентације је преузет из чланака:

- **Why Software Fails**, аутора Robert Charette:, IEEE Spectrum, September 2005, https://spectrum.ieee.org/why-software-fails
- **Lessons From a Decade of IT Failures** , аутора Robert Charette, Joshua Romero, IEEE Spectrum, October 2015, https://spectrum.ieee.org/lessons-from-a-decade-of-it-failures