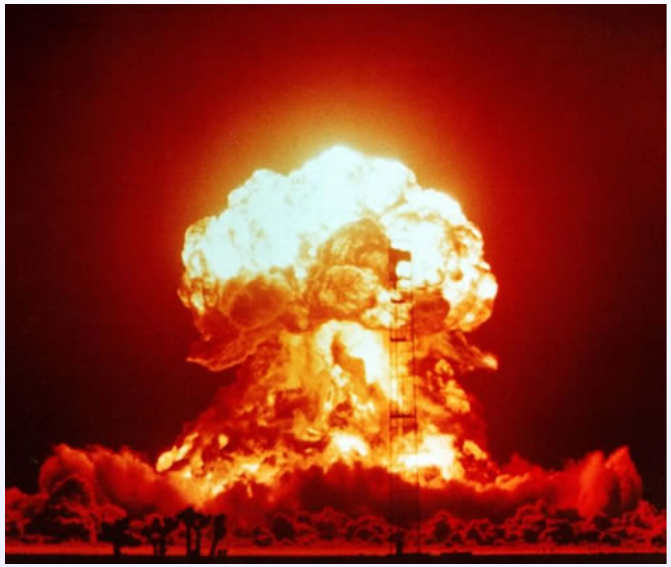
O Głośnych Pluskwach

Michał Ziemba

31 marca 2010

Plan prezentacji



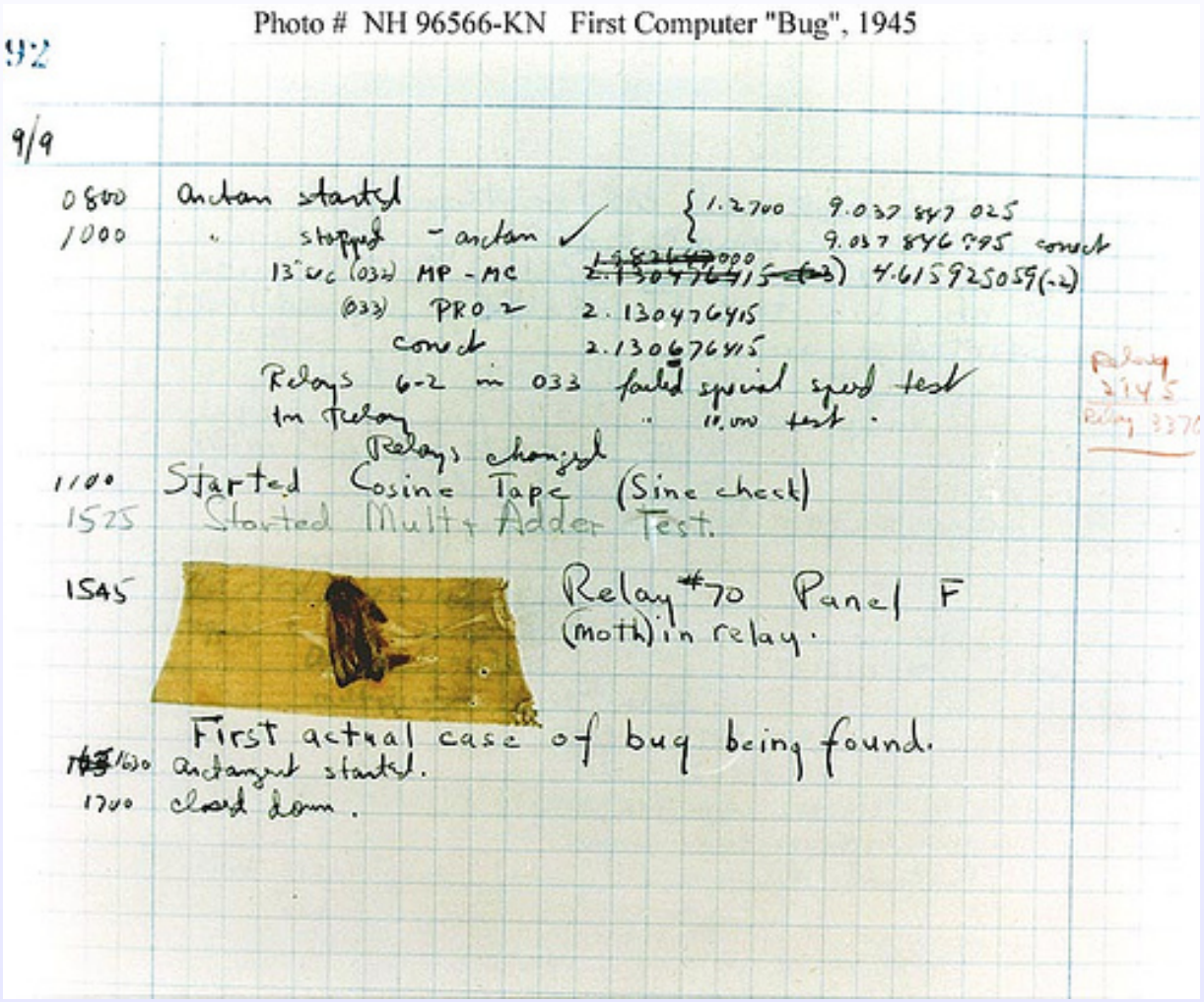
"To err is human, but to really foul things up you need a computer" – Paul Ehrlich

Pierwszy komputerowy bug, 1942

- ćma, znaleziona wewnątrz Mark II Aiken Relay Calculator na Harvardzie
- zespół, któremu przewodziła Grace Hopper
- owad znalazł się pomiędzy punktami przekaźnika #70 w panelu F
- 9 września 1942
- został umieszczony w logu z podpisem "First actual case of bug being found"

Grace Hopper

Pierwszy komputerowy bug



Mariner 1, 1962

- część programu Mariner, który składał się z 10 rakiet
- celem misji było wykonanie lotu w okolice Wenus
- start miał miejsce 22 lipca 1962
- na wniosek oficera nadzorującego lot, który zaobserwował nieoczekiwaną zmianę kursu, po 294.5 sekundach nastąpiła autodestrukcja

Mariner 1 – możliwe przyczyny

- przyczyny awarii rakiety Mariner 1 nie zostały w pełni wyjaśnione
- przez wiele lat krążyły legendy o rzekomym bugu w fortranowym kodzie odpowiadającym za sterowanie
- NASA jako przyczynę podaje niewłaściwe przeniesienie do kodu matematycznego symbolu

Mariner 1 – błąd w fortranowej pętli DO?

- bardzo częsty błąd popełniany w Fortranie
- użycie kropki zamiast przecinka w pętli DO

$$\text{DO 17 I} = 1, 10 \qquad \text{DO17I} = 1.10$$

- według osób pracujących w latach 60-tych w NASA, bug taki rzeczywiście wystąpił, ale w programie Mercury i został znaleziony zanim spowodował jakiekolwiek konsekwencje
- bug został znaleziony po wielotygodniowych testach numerycznych

```
1       IF (TVAL .LT. 0.2E-2) GOTO 40
2       DO 40 M = 1, 3
3       W0 = (M-1)*0.5
4       X = H*1.74533E-2*W0
5       DO 20 N0 = 1, 8
6       EPS = 5.0*10.0**(N0-7)
7       CALL BESJ(X, 0, B0, EPS, IER)
8       IF (IER .EQ. 0) GOTO 10
9    20 CONTINUE
10      DO 5 K = 1. 3
11      T(K) = W0
12      Z = 1.0/(X**2)*B1**2+3.0977E-4*B0**2
13      D(K) = 3.076E-2*2.0*(1.0/X*B0*B1+3.0977E-4*
14     *(B0**2-X*B0*B1))/Z
15      E(K) = H**2*93.2943*W0/SIN(W0)*Z
16      H = D(K)-E(K)
17    5 CONTINUE
18   10 CONTINUE
19      Y = H/W0-1
20   40 CONTINUE
```

Mariner 1 – brakujący symbol

$$\bar{\dot{R}}_n$$

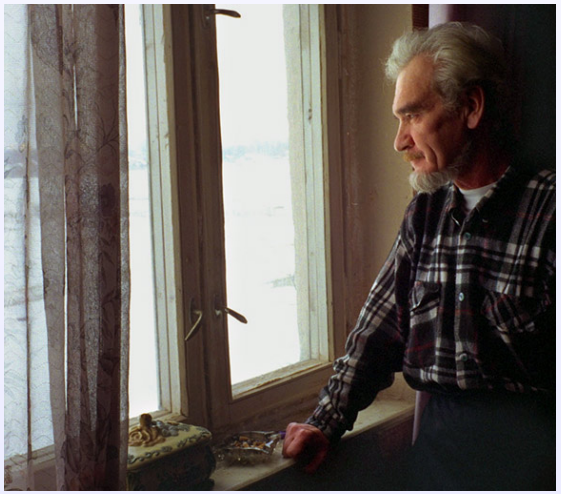
- symbol oznacza wartość zaokrąglonej pochodnej n-tego promienia R_n po czasie
- brak kreski nad R oznaczał brak wygładzenia pochodnej
- program bez wygładzenia każdą minimalną zmianę wartości prędkości traktował jako bardzo poważną, stąd seria gwałtownych manewrów która spowodowała zejście rakiety z kursu

Soviet gas pipeline, 1982

- w latach 70-tych władze rosyjskie powołały specjalną jednostkę KGB, której zadaniem było wykradanie zachodnich technologii
- w 1981 roku CIA otrzymało dokumenty przekazane przez pułkownika Vladimira Vetrova
- dokumenty zawierały listę przyszłych zakupów Związku Radzieckiego, na której znalazły się sprzęt i oprogramowanie
- prawdopodobnie oprogramowanie mające zarządzać rurociągiem gazowym zostało zabugowane tak, by przeszło proste testy, a następnie trafiło w ręce Rosjan
- być może bug był przyczyną największej nienuklearnej eksplozji w historii, została ona zauważona na zdjęciach satelitarnych Syberii

Stanislav Petrov ratuje świat, 1983

- podpułkownik Stanislav Petrov należał do załogi podmoskiewskiego bunkra, monitorującego radziecki system ostrzegania satelitarnego
- 26 września 1983 r. system pokazał 5 rakiet wystrzelonych w kierunku ZSRR
- Petrov, jak później stwierdził, "miał przeczucie"że to fałszywy alarm, taki też zdał raport
- śledztwo wykazało, że oprogramowanie niepoprawnie rozpoznało odbicie promieni słonecznych jako rakiety



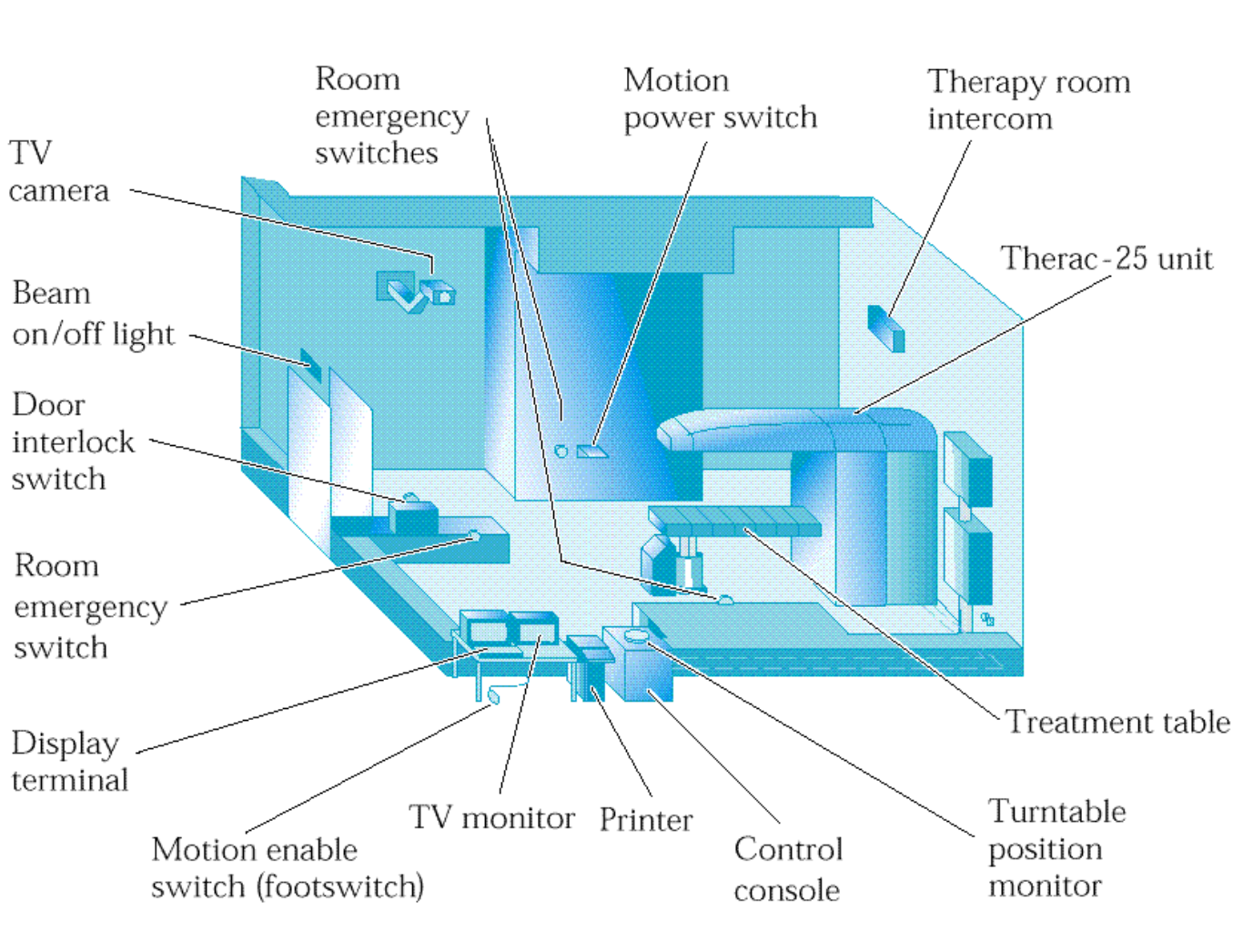
Therac-25, 1985-87

- Therac-25 był maszyną przeznaczoną do radioterapii (metody leczenia polegającej na użyciu promieniowania jonizującego do kontroli komórek nowotworowych)
- wyprodukowany przed Atomic Energy of Canada Limited (AECL)
- w latach 1985-1987 doszło do co najmniej sześciu przypadków nadmiernego napromieniowania, w wyniku czego trzy osoby zginęły
- osoby te otrzymały ponad 100 razy większą dawkę promieniowania niż powinny

Therac–25 – przyczyny tragedii

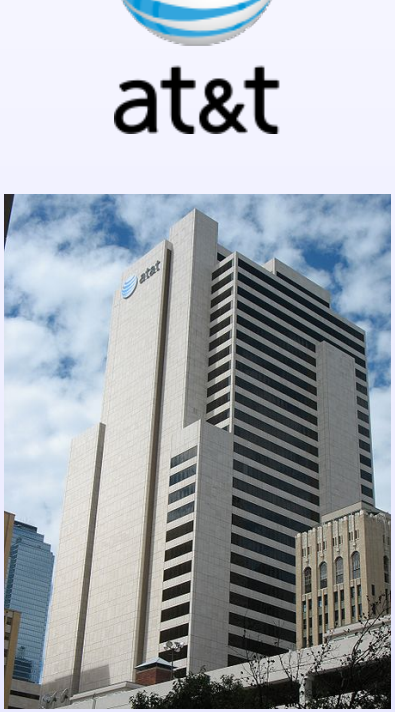
- Therac-25 obsługiwał dwa tryby działania: krótkotrwałe promieniowanie wysokoenergetyczną wiązką elektronów, lub użycie promieniowania rentgenowskiego, powstałego z wiązki elektronów
- przyczyną nadmiernego napromieniowania było użycie przez maszynę wysokoenergetycznej wiązki zamiast niskoenergetycznego promieniowania rentgenowskiego
- badania wykazały, że proces obsługujący sprzęt i proces obsługujący interfejs operatora nie były zsynchronizowane, co powodowało wystąpienie race conditions
- wykorzystywano oprogramowanie z wcześniejszych modeli (Therac–20)

Therac–25


Room emergency switches
Motion power switch
Therapy room intercom
TV camera
Therac-25 unit
Beam on/off light
Door interlock switch
Room emergency switch
Treatment table
Display terminal
TV monitor
Printer
Turntable position monitor
Motion enable switch (footswitch)
Control console

AT&T network collapse, 1990

- AT&T jest największym operatorem telefonicznym w USA
- w roku 1990 obsługiwało 70% rozmów długodystansowych
- stworzono sieć 114 switchów rozsianych po całych USA, realizujących połączenia telefoniczne
- switche obsługiwały ponad 115 milionów rozmów dziennie
- błąd w oprogramowaniu switchów spowodował paraliż sieci na 9 godzin, blokując 50 milionów rozmów



AT&T – przyczyny awarii

- 15 stycznia 1990 switch w Nowym Jorku wykonał planowy reset, po którym wysłał informację o wznowieniu działania, a następnie zajął się obsługą żądań, które pojawiły się w czasie resetu
- niektóre z pozostałych switchów otrzymały dwie wiadomości z Nowego Jorku, w odstępie 4 milisekund
- bug w oprogramowaniu switcha spowodował, że druga wiadomość, zamiast zostać zapisana w buforze, nadpisała ważne dane komunikacyjne
- to z kolei wywołało reset kolejnego switcha, i propagację kolejnej pary następujących zaraz po sobie wiadomości do kolejnych węzłów sieci

AT&T – przyczyny awarii

```
1  while (ring receive buffer not empty and side buffer not empty) do
2      Initialize pointer to first message in side buffer or ring receive buffer
3      get copy of buffer
4      switch (message)
5          case (incoming_message):
6              if (sending switch is out of service) do
7                  if (ring write buffer is empty) do
8                      send "in service" to status map
9                  else
10                     break
11                 END IF
12                 process incoming message, set up pointers to optional parameters
13                 break
14     END SWITCH
15 do optional parameter work
```

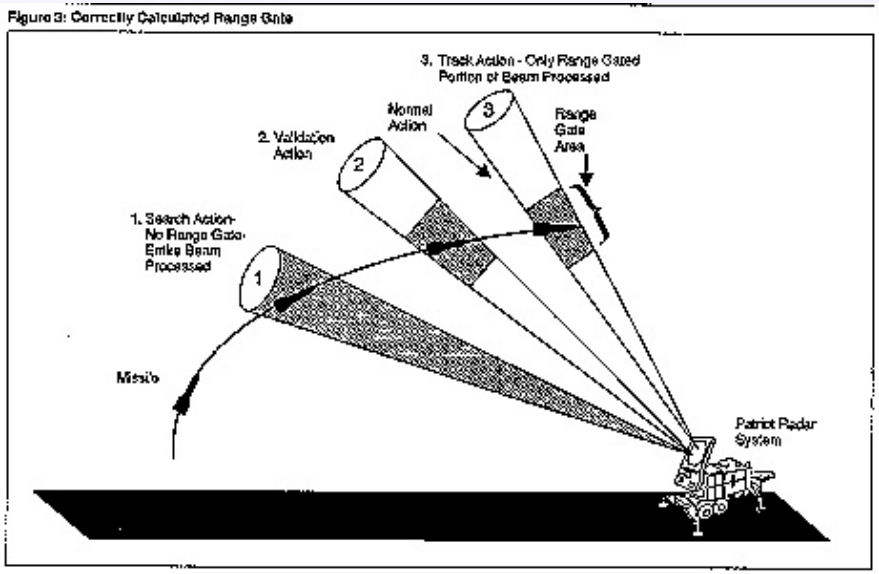
Patriot missile failure, 1991

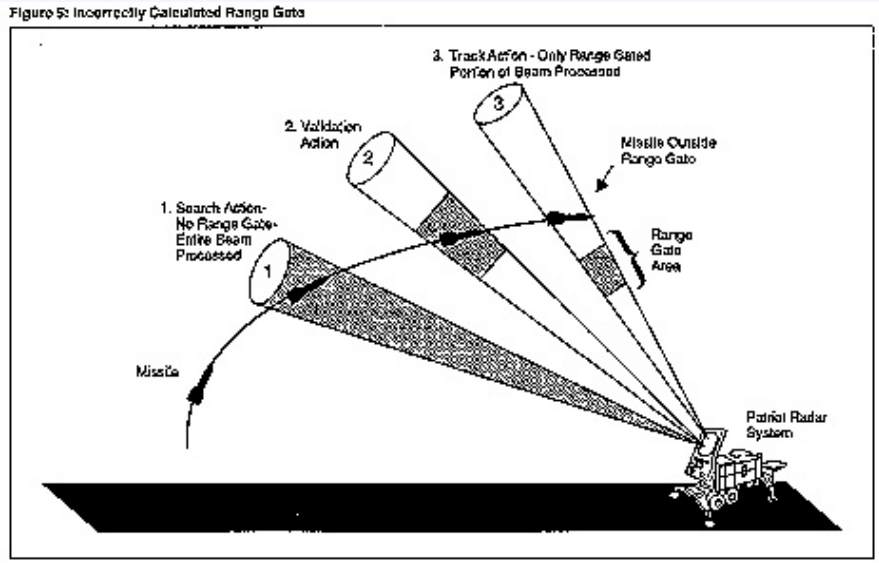
- 25 lutego 1991, podczas Wojny w Zatoce Perskiej system obrony przeciwrakietowej Patriot przepuścił nadlatującą rakietę wystrzeloną przez Irakijczyków w Dharan w Arabii Saudyjskiej
- trafiła ona w amerykańskie baraki, zabijając 28 żołnierzy, a raniąc kolejnych 98
- powodem był błąd numeryczny w oprogramowaniu

Patriot – przyczyny tragedii

- błąd wystąpił w algorytmie obliczania przyszłej lokalizacji rakiety na podstawie jej prędkości i ostatniego odczytu radaru
- komputer obsługujący system Patriot miał rejestry 24-bitowe
- czas był liczony w dziesiętnych sekundy, reprezentacja binarna 0.1 jest nieskończona, zatem obcinano rozwinięcie po 24 bitach
- niedokładność była tym większa, im dłużej system działał, zalecało się wykonywać restart co 8 godzin
- system który spowodował katastrofę działał przez 100 godzin, co spowodowało niedokładność 0.34 sekundy

Patriot – przyczyny tragedii





Intel floating point divide, 1993

- bug w procesorach Intela, powodujący niewłaściwe wyniki przy dzieleniu zmiennoprzecinkowym
- odkrył go Thomas Nicely, profesor w Lynchburg College w Virginii, w październiku 1994, wykonując obliczenia związane z liczbami pierwszymi
- w mailu do Intela prof. Nicely opisał problem, a także podał przykłady działań dających nieprawidłowe wyniki (np. 1/824633702441.0)
- wkrótce potem wiele ludzi potwierdziło w Internecie istnienie buga

Intel FDIV – przyczyny

- procesory Intela wykorzystywały algorytm dzielenia SRT (Sweeney, Robertson, Tocher)
- bug był spowodowany nieprawidłowymi wpisami w look-up table, która jest generowana na podstawie dzielnej i dzielnika i służy do zgadywania kolejnych cyfr ilorazu
- błędny skrypt kopiujący look-up table na sprzętowy PLA (Programmable Lookup Array)
- błąd dokładności występował średnio raz na 9 miliardów operacji dzielenia
- największa stwierdzona niedokładność – czwarte miejsce po przecinku

Intel FDIV – krytyka

- Intel przyznał, że problem istnieje, ale stwierdził że nie jest poważny i nie ma wpływu na większość użytkowników (przeciętny użytkownik doświadczy błędu raz na 27000 lat)
- dano możliwość wymiany procesora tylko w przypadku, jeśli udowodni się że jest on wadliwy
- spowodowało to ostrą reakcję społeczeństwa, do której przyłączyły się także duże firmy, jak na przykład IBM
- pod wpływem nacisków Intel zgodził się wymieniać wszystkie wadliwe procesory
- koszty związane z wymianą procesorów wyniosły 475 milionów dolarów

Ping of Death, 1995-1996

- bug w implementacji TCP-IP, umożliwiający atak sieciowy, powodujący reboot, zawieszenie lub błąd systemu
- polegał na wysłaniu nieprawidłowego pakietu IP przy pomocy programu ping
- w większości systemów implementacja programu ping uniemożliwiała wysłanie błędnego datagramu, wyjątkami były Windows '95 i NT
- wiele systemów okazało się podatnych na działanie POD, jednak szybko powstały patche (dla Linuxa po trzech godzinach od ujawnienia buga)

Ping of Death – przyczyny

- maksymalna długość pakietu IP wynosi 65535 bajtów (RFC 791), w tym co nazjwyżej 65507 bajtów danych
- większe pakiety ulegają fragmentacji, każda część zawiera w nagłówku offset umozliwiający ponowne połączenie fragmentów
- istnieje możliwość, że dla ostatniego fragmentu wysyłanego pakietu zajdzie $(offset + dane) > 65535$, co może spowodować przepełnienie bufora na odbierającej maszynie

```
ping -l 65510 <adres ip>
```

Ariane 5, 1996

- 4 czerwca 1996 bezzałogowa rakieta Ariane 5 została wystrzelona przez Europejską Agencję Kosmiczną z Kourou w Gujanie Francuskiej
- niespełna 40 sekund po starcie, na wysokości 3700 m, doszło do autodestrukcji
- program wart 7 miliardów dolarów, sama rakieta (z czterema satelitami na podładzie) – 500 milionów
- dwa tygodnie po awarii powstał raport

Ariane 5 – przyczyny

- sterowaniem rakietą zajmował się główny komputer pokładowy, na podstawie danych dostarczanych przez Inertial Reference System (SRI)
- bug pojawił się w funkcji odpowiedzialnej za wyrównywanie jednej z platform podczas startu
- konwersja 64-bitowego floata do 16-bitowego signed integera (kod pisany w Adzie) spowodowała nieobsłużony wyjątek

Ariane 5 – błędy programistów

- wcześniejsza wersja rakiety, oznaczona numerem 4, wykorzystywała to samo oprogramowanie
- w Ariane 4 bug nie występował, ponieważ trajektoria wczesnej fazy lotu różniła się od Ariane 5, a to oznacza mniejsze możliwe wartości przyspieszenia
- przy podobnych konwersjach w innych miejscach kodu sprawdzano możliwość rzucenia wyjątku

Mars Pathfinder, 1997

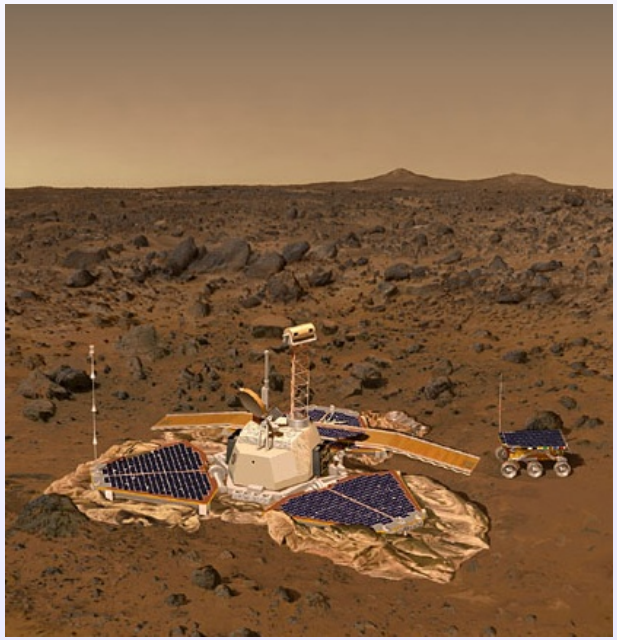
- sonda kosmiczna NASA, przeznaczona do badań Marsa, składająca się z dwóch części: lądownika i pojazdu o nazwie Sojourner
- start nastąpił 4 grudnia 1996 z przylądka Canaveral na Florydzie, lądowanie na Marsie – 4 lipca 1997
- niedługo po wylądowaniu system zaczął nieoczekiwanie się resetować, powodując utratę danych
- mimo to misja zakończyła się sukcesem

Mars Pathfinder – priority inversion

- Pathfinder używał VxWorks – jądra systemu czasu rzeczywistego, stworzonego przez Wind River Systems
- VxWorks używał schedulera wykorzystującego priorytety, przypisywane poszczególnym procesom
- za synchronizację procesów odpowiadały muteksy
- istniała możliwość, że przerwanie spowoduje wybranie do wykonania procesu komunikacyjnego (średni priorytet) w czasie, gdy proces obsługujący szynę danych (wysoki priorytet) czekał na proces zbierający dane pogodowe (niski priorytet)
- po jakimś czasie system stwierdził, że proces obsługujący szynę danych dawno się nie wykonywał, więc musiało się stać coś złego, zatem wykonano reset systemu

Mars Pathfinder – szukanie błędu

- VxWorks może być uruchomione w trybie, gdy zapisywane są informacje o zmianach kontekstu, przerwaniach itp. (trace)
- godzinami inżynierowie w JPL badali zachowanie systemu z włączonym tracingiem na replice Pathfindera, szukając przyczyny resetów
- po całonocnej pracy, następnego dnia nad ranem udało się zasymulować błędne działanie systemu, dzięki czemu znaleziono błąd

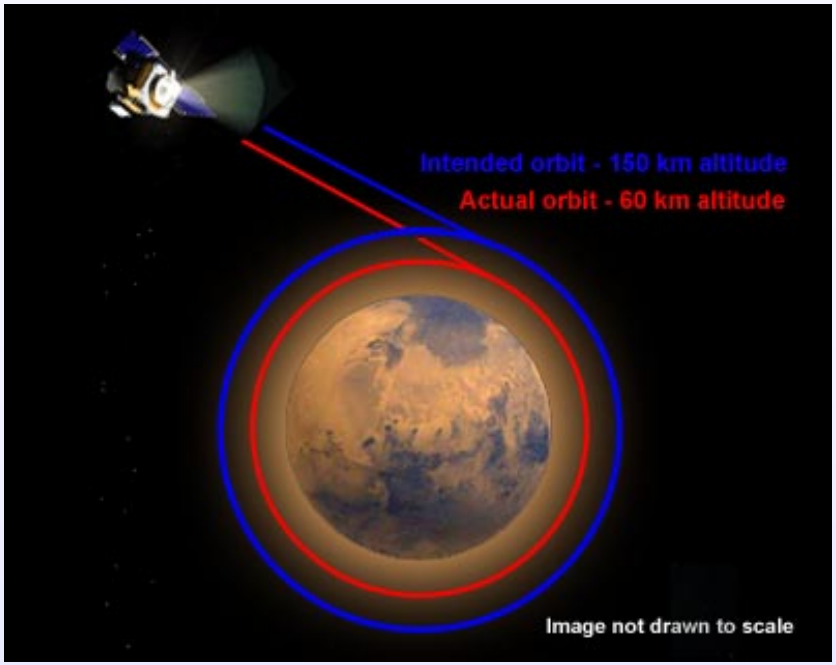


Mars Climate Orbiter, 1998

- część programu Mars Surveyor '98, razem z Mars Polar Lander
- celem misji było poznawanie klimatu, pogody, zawartości wody i CO2 na Marsie
- start odbył się 11 grudnia 1998 z przylądka Canaveral na Florydzie
- MCO miał wejść w orbitę na wysokości 140-150 km, w rzeczywistości było to 57 km
- MCO uległ zniszczeniu w wyniku działania zbyt dużego ciśnienia 23 września 1999

Mars Climate Orbiter – przyczyny awarii

- przyczyną było użycie niewłaściwych jednostek miary
- system sterujący rakietą oczekiwał wartości nacisku w Newtonach (system metryczny), a otrzymał w funtach (amerykański system miar)
- niewłaściwe wartości siły nacisku spowodowały wyznaczenie niewłaściwej trajektorii lotu



Mars Climate Orbiter – szukanie błędu

- 15 października 1999 powołano specjalną komisję, której zadaniem było zbadanie przyczyn niepowodzenia misji
- komisja (MCO Mishap Investigation Board) składała się z 12 osób, oraz 8 zewnętrznych ekspertów
- metoda działania opisana jako “process that alternated between individual brainstorming and group discussion”
- oprócz wadliwego kodu który spowodował niewłaściwy tor lotu, komisja znalazła wiele mniejszych uchybień projektowych, które miały być poprawione w Mars Polar Landerze

USS Yorktown, 1998

- krążownik amerykańskiej Marynarki Wojennej, aktywny w latach 1984-2004
- od 1997 roku funkcjonował jako poligon doświadczalny, testowano na nim oprogramowanie Smart Ship (zarządzające siecią, złożoną z 27 komputerów, odpowiadające m.in. za monitorowanie stanu paliwa, pracy silników oraz sterowanie statkiem)
- 21 września 1997 jeden z członków załogi umieścił w bazie danych liczbę 0, co spowodowało awarię całego systemu (błąd dzielenia przez 0)
- mimo że bug nie miał związku z systemem operacyjnym, sytuacja ta była zalążkiem do dyskusji nad słusznością użycia Windowsa NT, zamiast bardziej stabilnego Unixa

Podsumowanie

- błędy w oprogramowaniu były przyczyną wielu historycznych katastrof, spowodowały straty liczone w miliardach dolarów
- systemy często nie są odpowiednio testowane, brak testów w przypadku wdrożenia nowszych wersji oprogramowania
- w przypadku głośnych bugów niekiedy trudno jest o dokładną dokumentację tego, co się stało
- najczęściej stosowaną metodą szukania bugów jest symulowanie zachowania systemu które doprowadziło do danej awarii i analiza kodu przez powołaną do tego grupę ekspertów
- większość głośnych bugów miała miejsce przed rokiem 2000, można zatem sądzić że w nowych systemach kładzie się większy nacisk na testowanie oprogramowania

Bibliografia

- http://www5.in.tum.de/~huckle/bugse.html
- http://users.csc.calpoly.edu/~jdalbey/SWE/Papers/att_collapse.html
- ftp://ftp.hq.nasa.gov/pub/pao/reports/1999/MCO_report.pdf
- http://catless.ncl.ac.uk/Risks/
- http://www.absoluteastronomy.com/topics/Therac-25
- http://www.intel.com/support/processors/pentium/fdiv/wp/
- http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html
- http://www.ima.umn.edu/~arnold/disasters/patriot.html

Dziękuję za uwagę