Prevádzka a údržba globálneho softévru
Prípadová štúdia (Case Study)

Andrej Danko, PhD.
Máj 2011

Čo je to globálny softvér?
Východiskové podmienky a stratégia

Softvér na riadenie podnikových zdrojov

- Integrované riešenie s jednotným zdrojom údajov a kontrolnými manažérskymi nástrojmi pre riadenie všetkých kľúčových operácií podniku
- Finančné a nákupno-predajné procesy, sklady, správa zdrojov, vzťahov so zákazníkmi, účtovníctvo a výkazníctvo => *všetko vysoko legislatívne závislé*

Expanzia a globalizácia

- Úvodná podpora 3 krajín/lokalizácií (celkovo ~1 500 zákazníkov) v roku 2003
- Cieľ: expanzia do desiatok krajín celosvetovo a 10x znásobenie počtu zákazníkov do 5 rokov
- Priorita bola v čo najväčšej rýchlosti dodávky na trh
- Vytvorenie globálneho riešenia pre malé a stredné podniky

Softvérový pohľad

- **Jedna báza zdrojového** kódu (tzv. A vetva) v C/C++
- Všeobecná biznis logika a lokalizácia (krajinne-špecifická/závislá biznis logika) **v rámci jedného zdrojového kódu**
- Centralizované vývojové prostredie (z technického i organizačného pohľadu)

Stratégia globalizácie v roku 2003

- Rozdelenie zdrojového kódu na 3 vetvy/bázy:
  - A vetva: EU + Amerika (jadro, nové procesy, lok.)
  - B vetva: Ázia + ANZ (lokalizácia)
  - C vetva: východná Európa + Rusko (lokalizácia)
- Tri rôzne verzie (A; B a C, založené na A)
- 3 rôzne tímy: jeden pôvodný – A (tím v Izraeli), dva nové – B (tím v Ázii) a C (tím v Európe)

Implementácia globalizačnej stratégie a vplyv na údržbu

Problémy a výzvy (s čím „bojoval“ výrobca softvéru)

Štrukturálne

- Závislosť kódových báz (B a C na A)
  - Sekvenčná nadväznosť kódových báz a s tým spojené otázky projektového manažmentu verzií a dodávok
- Komplexnosť transportu *zmien, opráv a nových biznis procesov a funkcií* medzi jednotlivými vetvami (nové verzie, medzi verziami, opravami)
- Nedostatočná či úplne chýbajúca dokumentácia

Procesné a organizačné

- Nepostačujúce globálne procesy/vývojové štandardy a postupy pri tvorbe/zmene kódu
- Nevhodné nástroje na podporu, neexistujúce rozsiahle automatizované (regresné) testovanie
- 90% zmien v A vetve, dlhé prestoje dodávok v B a C
- 3 tímy v rôznych časových pásmach, zložitá synchronizácia, kulturálne rozdiely a chápanie

Počet opravných balíkov (angl. patches) a urgentných zmien (angl. emergency fix) v 2006

Version	Monthly patches	Additional emergency Fixes	Total
6,5	0	0	0
7.1 B	0	0	0
7,6	1	0	1
2004 A	13	1	14
2004 B	10	3	13
2004 2B	10	7	17
2004 C	11	3	14
2005 A	8	0	8
2005 A SP1	13	4	17
2005 B	3	1	4
2004 B Linux	6	0	6
2005 B B1BC	0	9	9
Total	**75**	**28**	**103**

Zákaznícky pohľad a interné dôsledky

Problémy a výzvy (s čím „bojovali“ zákazníci a dopady na architektúru softvéru a procesy)

Prevádzka softvéru u zákazníkov

- Vysoká chybovosť softvéru a následná nespokojnosť
- Dodávky nových funkcií a biznis procesov – *prispôsobenie a zlepšenie SW* – zaostávali za očakávaniami zákazníkov (časovo, obsahovo)
- Zákazníci na A vetve nemohli využiť zmeny v B a C vetvách (často krát generické, nie lokalizačné)
- Prechod na novú verziu (či opravný balík) v rámci vetvy bol komplikovaný a sprevádzaný regresiami

Podpora a riešenie problémov

- Komplikovaný proces podpory a dodávania opráv ak oprava požaduje zmenu v A vetve (90%) a prichádzala z B a C vetvy
- Nespokojnosť zákazníkov s kvalitou a časovými dodávkami opráv

„The only thing we maintain is user satisfaction.“ (M.M. Lehman)

“Spaghetti code”

- Výsledkom masívnej expanzie bol ťažko udržiavateľný a neprehľadný zdrojový kód
- Náklady na synchronizáciu a transport (angl. merge) nového vývoja i opráv z A do B a C boli enormné
- Mnoho realizovaných požiadaviek bolo chybne implementovaných (*návrh*), niektoré až skoro fatálne (*chyby v špecifikácii*) pre ďalší vývoj a existenciu samotného produktu

Organizačné limity

- Frustrácia vývojárov bola značná, keďže väčšinu času nerobili nový vývoj ale transport/merge – 70%
- Chýbajúce posilnenie kompetencií B a C tímov

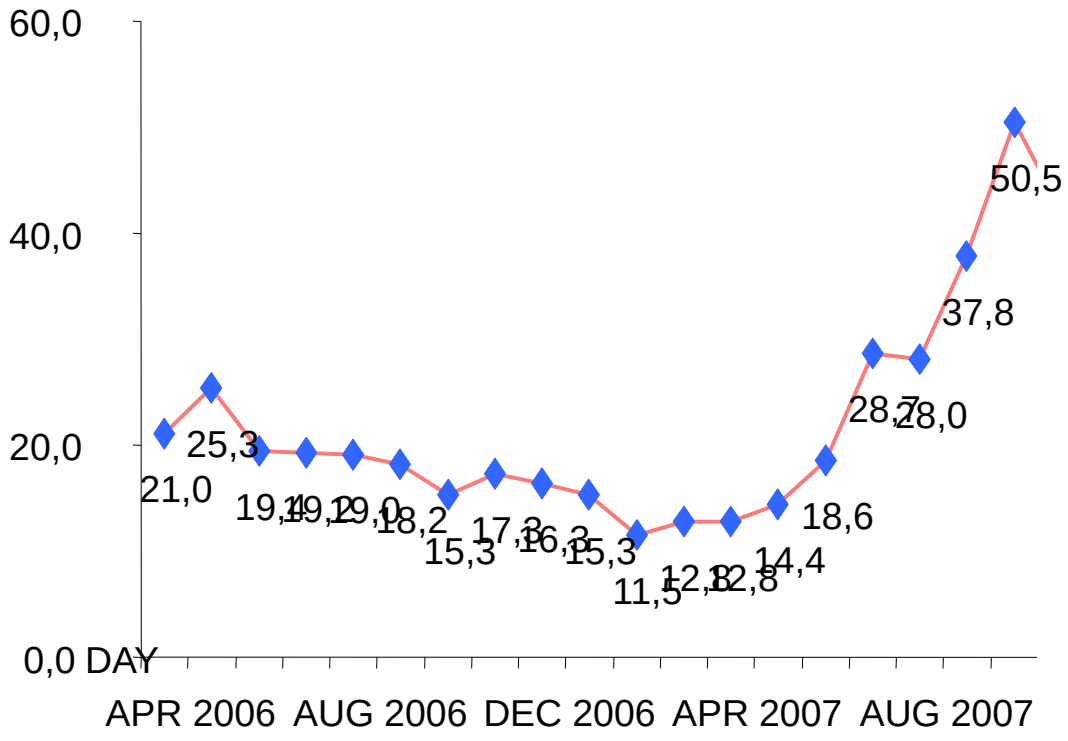
Front nespracovaných požiadaviek

- Legislatívne požiadavky, defekty, neúplné biznis procesy a funkcie systému, minimálne inovácie

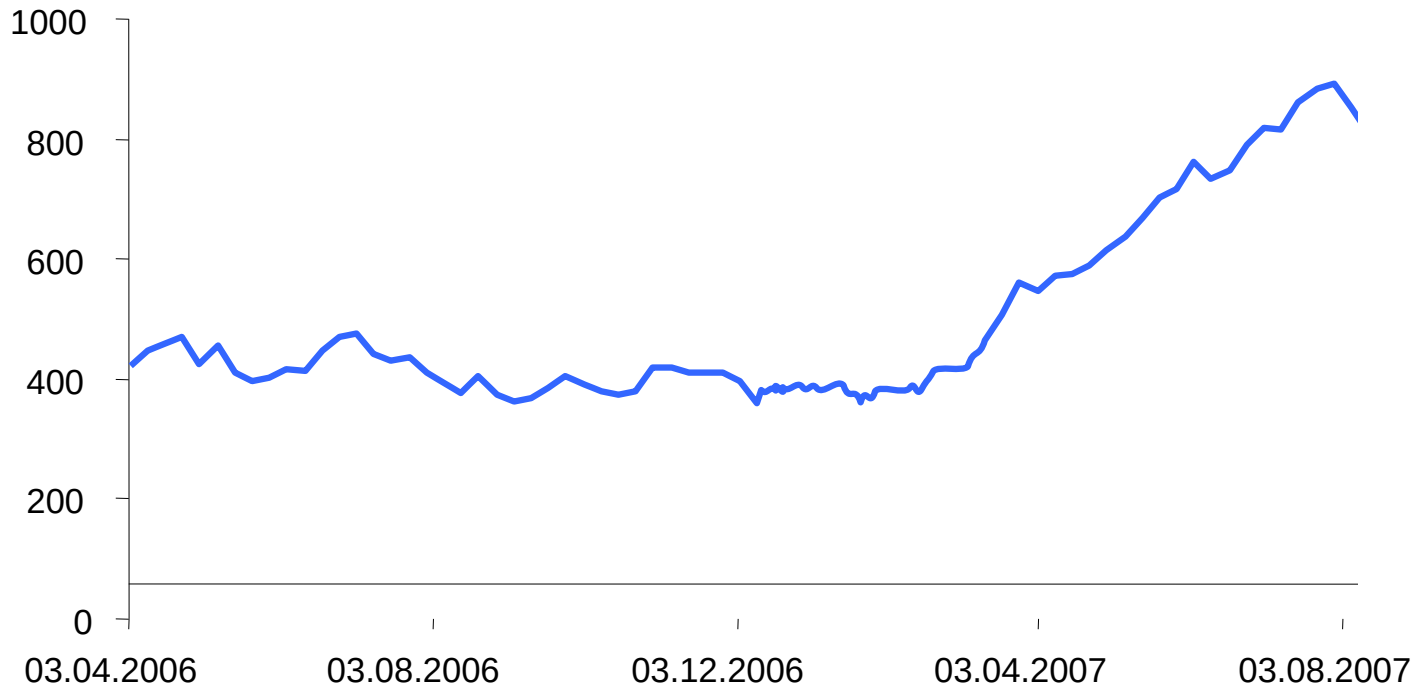
Ukazovatele priamo ovplyvňujúce spokojnosť zákazníkov

V rokoch 2006 a 2007 počas/po expanzii

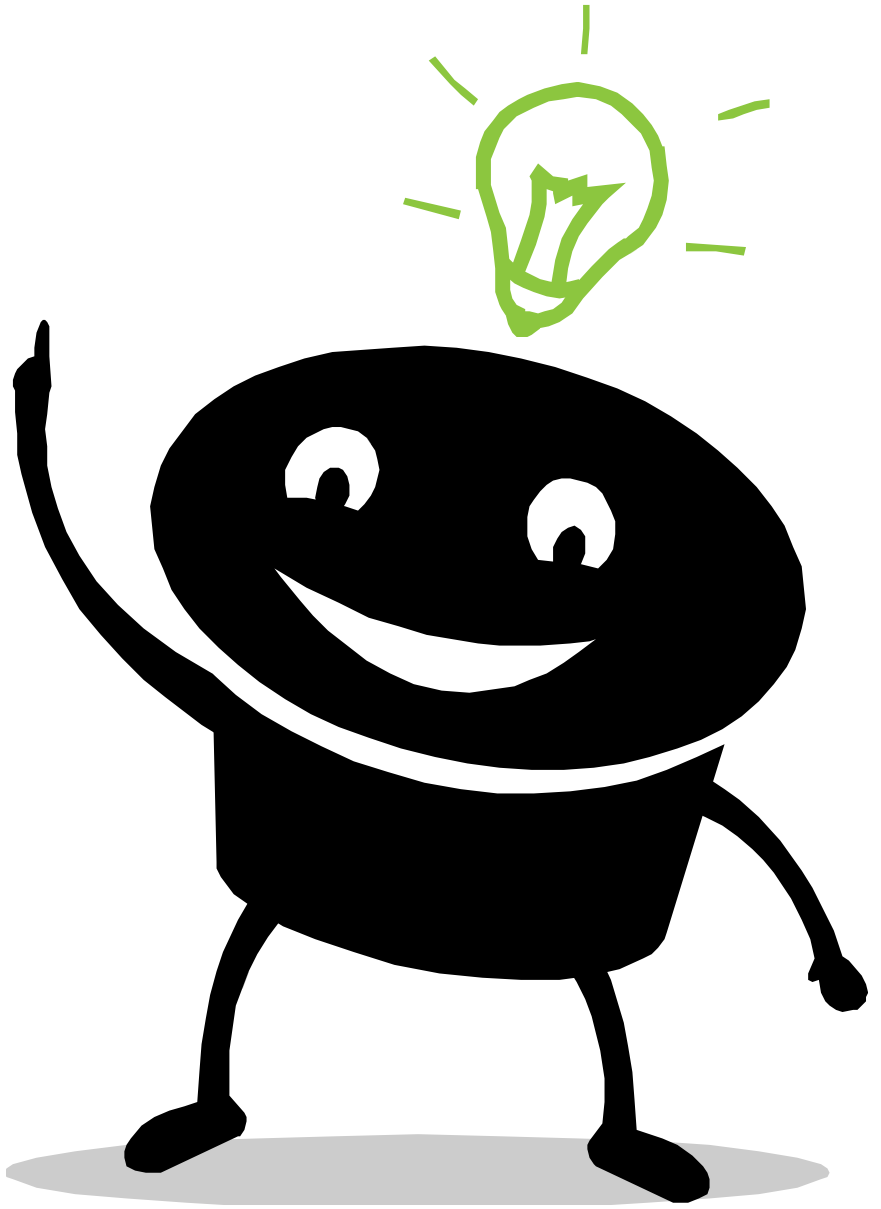
Priemerný čas (v dňoch) potrebný na vyriešenie jedného incidentu



Počet otvorených (nevyriešených incidentov - angl. Incident backlog)



Ako by ste riešili danú situáciu?



Stratégia a riešenie na ďalší udržateľný rast a spokojnosť zákazníkov

Od roku 2007

Spätná integrácia a reinžiniering

- Postupné spojenie vetiev a vytvorenie jednotnej kódovej bázy, najprv A+C => a následne + vetva B – 3 ročný proces (2007-2009)
- Obnovenie návrhu za účelom reštrukturalizácie systému (spojenie funkcií medzi vetvami, prechod na OO štruktúru, odstránenie návrhových defektov)
- Vytvorenie regresného testovania pomocou rozsiahleho automatizovaného testovacieho súboru

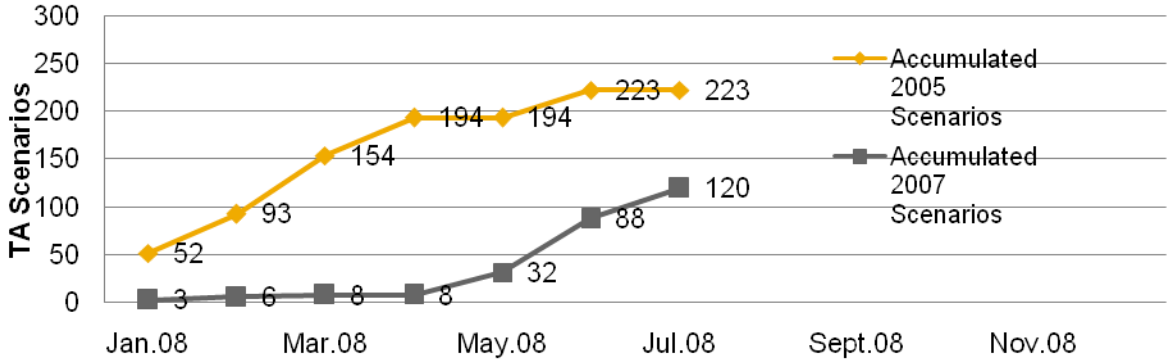
Systematické znižovanie TCO prevádzky softvéru

- Postupné spojenie vetiev a vytvorenie jednotnej bázy
- Automatizované nástroje na upgrade, zdravie systému

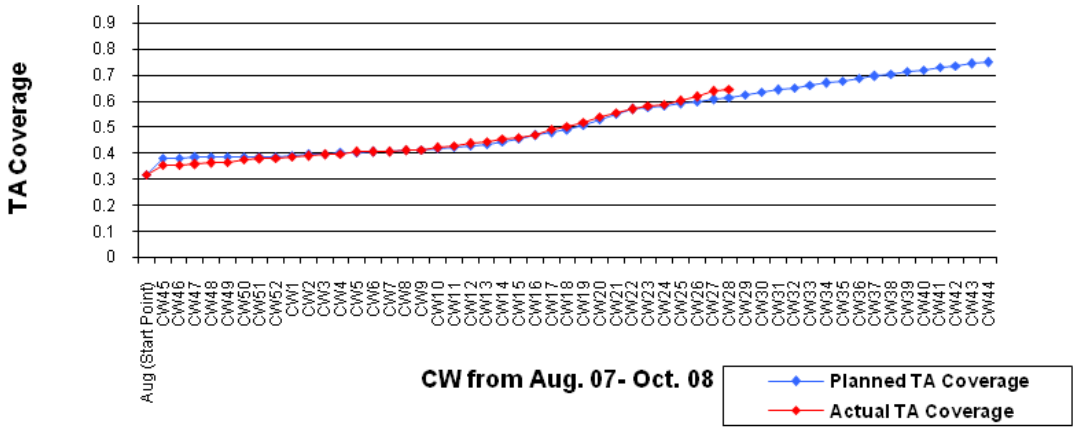
Globálna organizácia a procesy

- Vytvorenie globálnej vývojovej organizácie
- Vytvorenie globálnej organizácie údržby softvéru
- Reštrukturalizácia a zefektívnenie podporných procesov

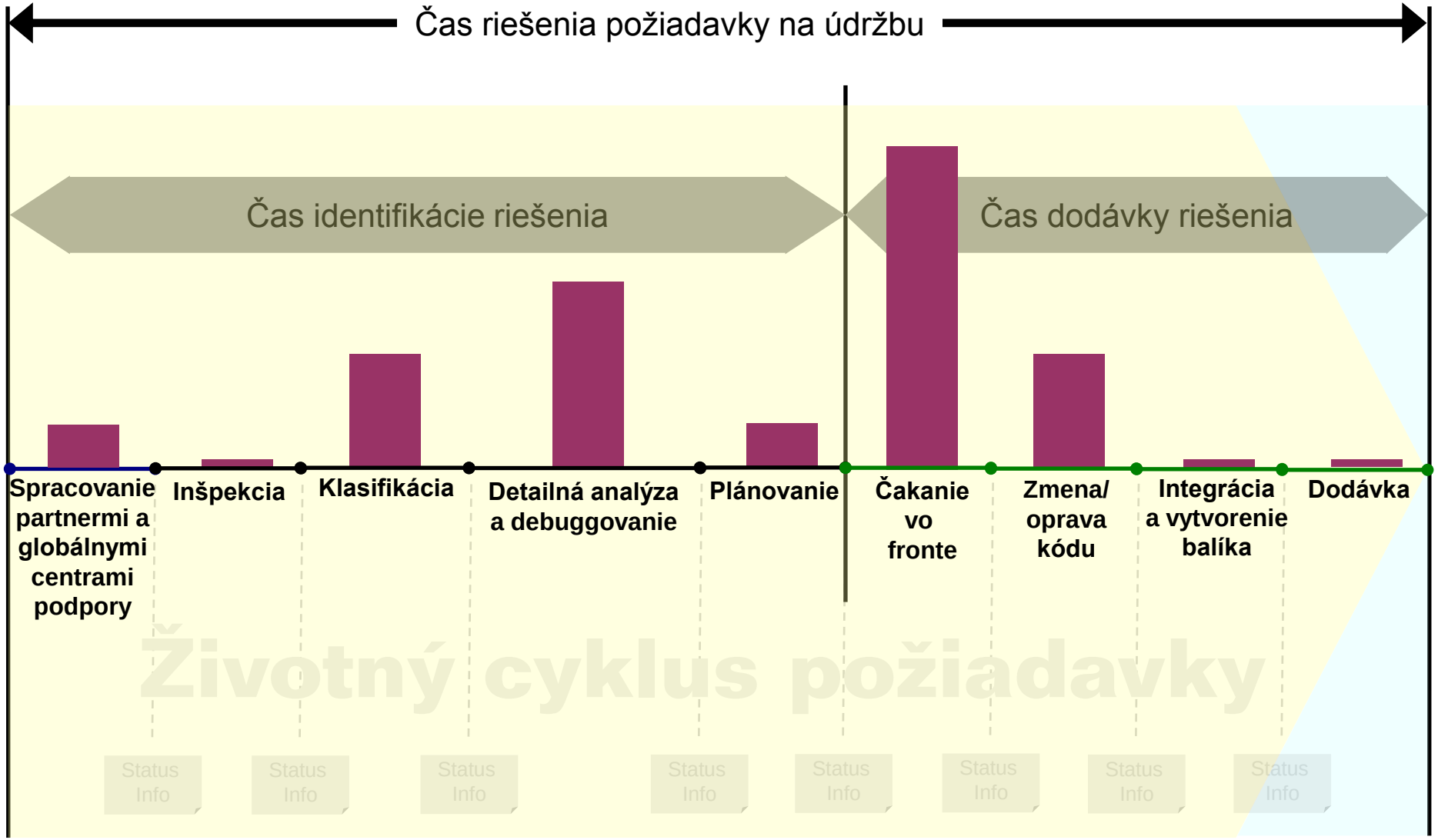
Celkový počet scenárov (založených na zákazníckych podnetoch) pokrytých automatizovaným testovaním



% pokrytie aplikácie automatizovaným testovaním



Globálny proces spracovania požiadaviek na údržbu



Výsledok implementácie upravenej stratégie
2008-2011

Z pohľadu architektúry softvéru
- Najnovšia verzia má jednu kódovú bázu (A+B+C)
- Mnohé kritické časti softvéru prešli reinžinieringom

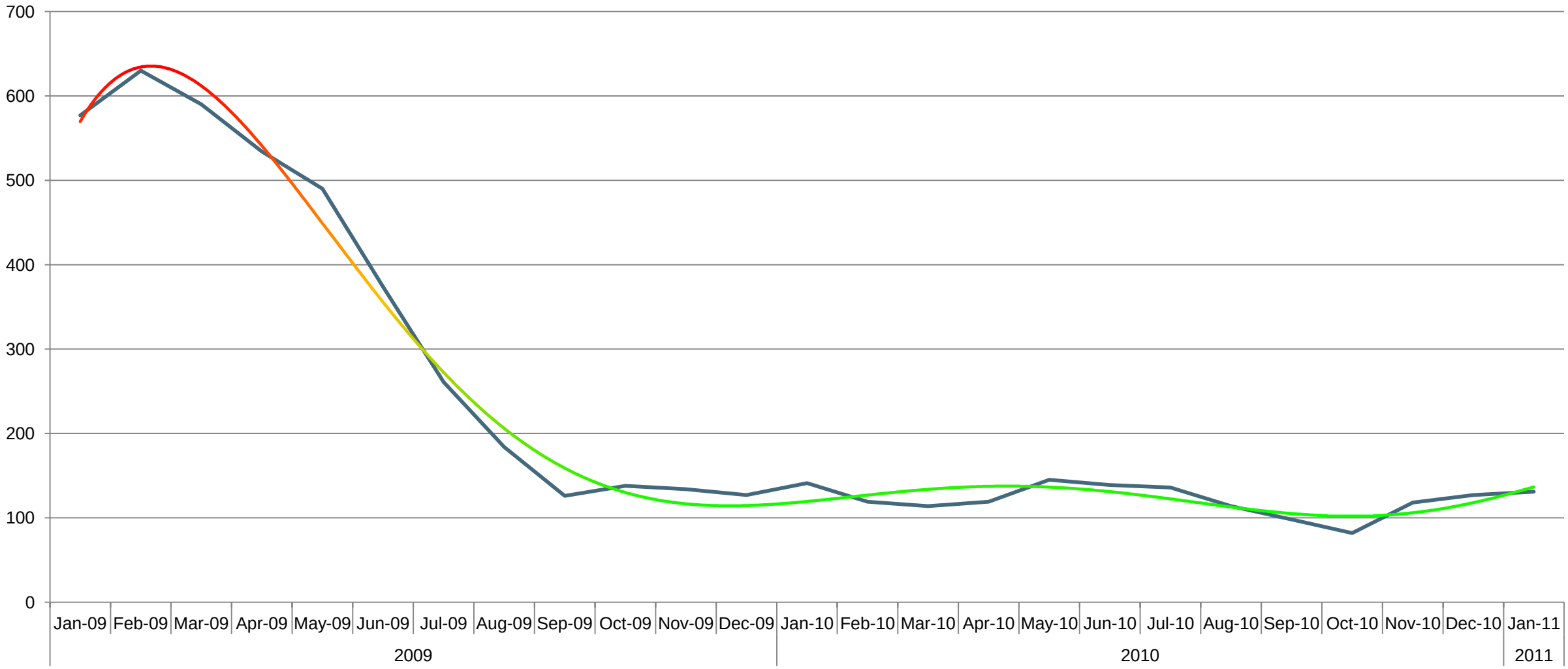
Z pohľadu prevádzky a údržby softvéru
- V údržbe sú vetvy A (spojená A+C) a B – obe do konca roku 2011 spolu s najnovšou verziou (A+B+C)
- Počet opravných balíkov klesol na 20 ročne, znížil sa počet celkových verzií paralelne v údržbe
- Postupný presun z opravného typu údržby na prispôsobovanie, zlepšovanie a prevenciu

Z pohľadu trhu, partnerov a zákazníkov
- >30 000 zákazníkov ku koncu roku 2010
- Prudký nárast indexu spokojnosti zákazníkov
- Ziskovosť riešenia v rámci celého ekosystému (TCO)

Čo to stálo a čo nás zachránilo
- Náklady na *spätné inžinierstvo a reštrukturalizáciu* boli >10 000 človekodní a investície pretrvávajú dodnes
  - Odstránenie návrhových (architektonických) chýb a problémov vyplývajúcich z nepostačujúcich štandardov vyžadovalo obrovské úsilie a náklady
- Doba návratu je dlhá (4 roky), pretože je potrebné plánovať postupný prechod (upgrade) zákazníkov
- Spätné získanie dôvery zákazníkov a partnerov je mnohonásobne ťažšie ako jej strata
  - Mnoho marketingových a „face-to-face“ aktivít podporovaných priamo vývojom softvéru
- Získanie internej reputácie
- Našou „kotvou v prístave“ boli tisíce zákazníkov, ktorí produkt vnímali pozitívne

Pokles počtu nevyriešených incidentov ako dôsledok zmien

V rokoch 2009 a 2010

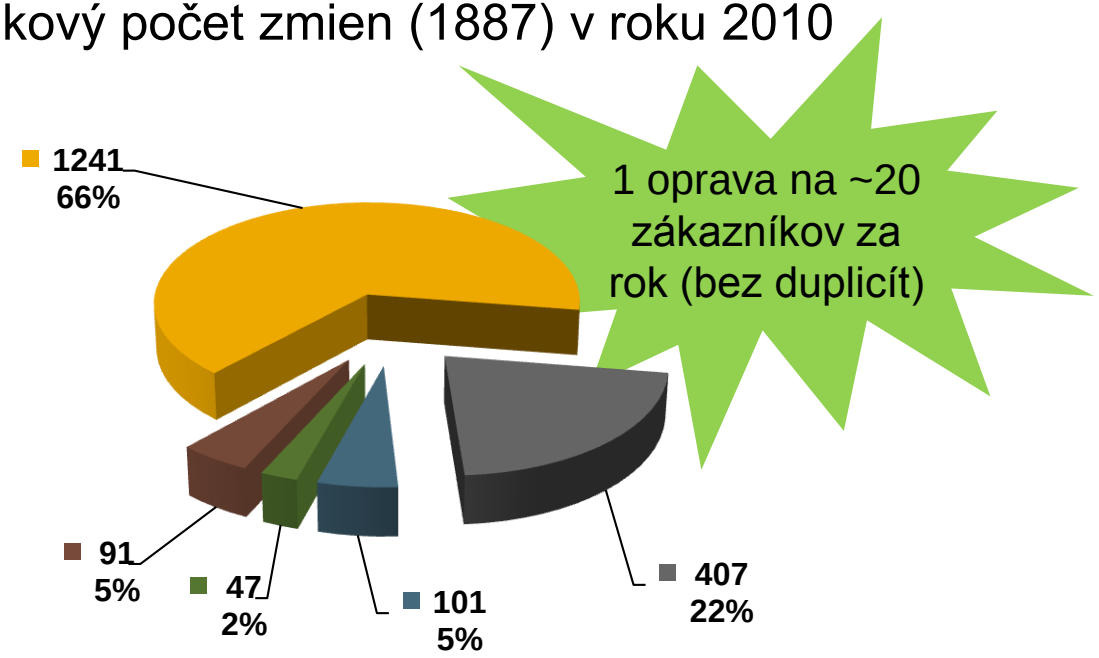


Údržba softvéru z pohľadu typov zmien a investícií

~30 000 zákazníkov celosvetovo (40 krajín), 2010

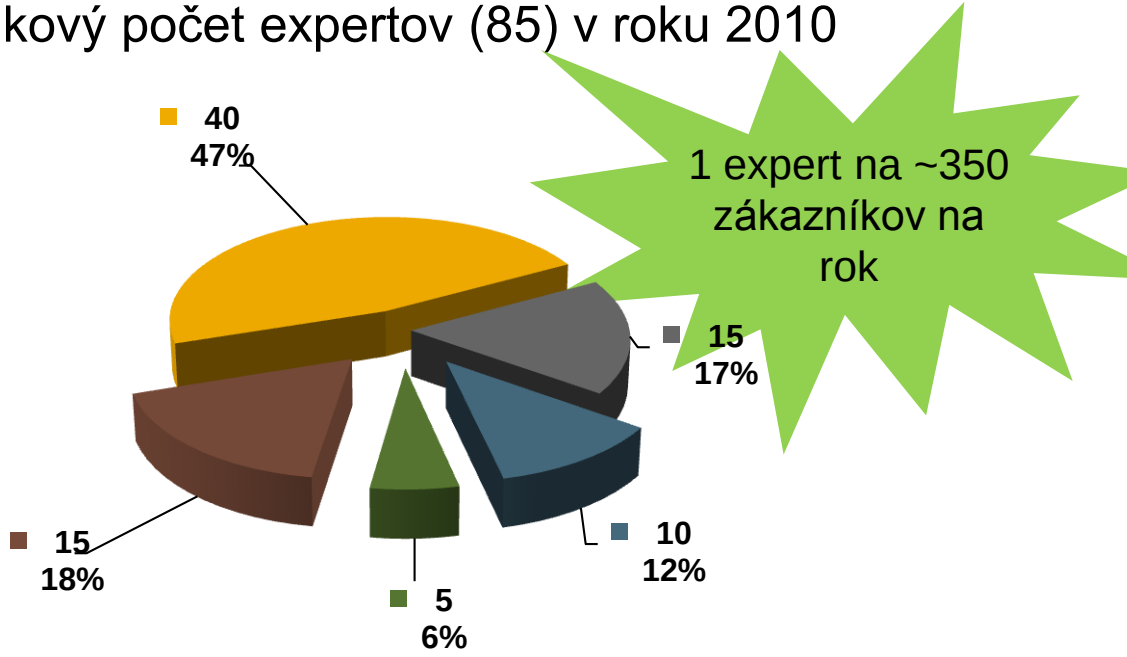
Typy zmien počas údržby

Celkový počet zmien (1887) v roku 2010



% podiel expertov (sw inžinieri) na riešení *

Celkový počet expertov (85) v roku 2010



- Opravy kódu (softvérové defekty)
- Opravy konzistencie BD (následky)
- Opravy v špecifikácii a kóde (dovývoj)
- Náhradné riešenia (bez zmeny kódu)
- Legislatívne zmeny/nové funkcie/zlepšenia

** neobsahuje investície do vývoja nových verzií, ktoré taktiež zahŕňajú opravy, prispôsobenia, zlepšenia a prevenciu pre existujúcich zákazníkov*

Na čo si dať pozor, aby ste sa vyvarovali tomu, čo už iní „prežili“

Schopnosť balansu udržateľnej kvality a kontinuálnych inovácií je kľúčová

Návrh, architektúra a štandardy tvorby softvéru

- Značne vplývajú na prevádzkyschopnosť a udržiavateľnosť softvéru:
  - spôsob aplikácie opravných balíkov,
  - zložitosť prechodu na nové verzie,
  - automatizácia činností spojených s prevádzkou a údržbou softvéru (angl. Application Life Cycle Management Automation)
  - interná štruktúra kódu, testovanie, atď.
- Od začiatku (fáza analýzy) je potrebné uvažovať s čo najnižším TCO v rámci celého ekosystému

Stratégia verzií

- Minimalizovať počet podporovaných verzií a s tým spojené náklady na údržbu
- Schopnosť dodávať v kratších i dlhších iteráciách

Procesné aspekty

- Analyzujte riskantnosť zmien v údržbe, avšak nájdite vhodný balans medzi konzervatívnym a inovatívnym
- Minimalizujte kompromis medzi kvalitou (napr. investovanie do testovania) a novými funkciami
- Aplikujte princípy agilného vývoja (fixné šprinty, inkrementálny a iteratívny prístup, kritéria ukončenia šprintu včítane automatizovaných testov)
- Spracúvajte incidenty od zákazníkov v čo najskorších fázach procesu podpory a údržby

Organizačné aspekty

- Neoddeľujte tímy pre nový vývoj a údržbu
  - pokiaľ je potrebné robiť veľa opravných typov zmien, niečo nie je v poriadku v rámci vývoja sw
- Adresujte zmeny počas údržby do tímov, ktoré majú skúsenosti s daným vývojom, resp. ich samé robili