SOFTWARE SICURO

COME SVILUPPARE SOFTWARE SICURO E STABILE IN MODO EFFICIENTE, RIPETIBILE ED ECONOMICO

Newsletter N°1 - Ottobre 2018

E' USCITO IL MIO NUOVO LIBRO

SOFTWARE SICURO

L'unico libro che ti fa muovere i primi passi nel mondo del **software critico** ad altissima qualità, senza che tu sia già un esperto di certificazione!

Impara come le migliori tecniche derivate dal **software critico aerospaziale**, nonché dalla programmazione **AGILE**, in realtà combinate insieme possono essere la tua migliore arma nella battaglia del software.

CHIEDI LA TUA COPIA GRATUITA A:

LIBRO@SOFTWARESICURO.IT

INTRODUZIONE

"Siamo in ritardo per colpa del software"

"Dovremmo adottare un nuovo processo di sviluppo ma non abbiamo il tempo"

"Un tool molto interessante ma non c'è budget, mi spiace"

" Il nostro metodo di sviluppo software è molto artigianale"

"Non abbiamo tempo di testare perché dobbiamo uscire col prodotto"

"Il test è noioso e costoso, dobbiamo tagliare i tempi"

"Vogliamo stare concentrati sul nostro core-business"

"Dobbiamo richiamare 10.000 unità dal mercato per un problema"

"Alcune persone si sono ferite utilizzando il nostro dispositivo"

"I parenti dei nostri clienti vittime di incidenti con i nostri veicoli si sono organizzati in una class-action"

Se qualcuna di queste frasi ti è nota perché è girata nella tua azienda o in quella di un tuo concorrente, oppure se il solo sentirla ti fa venire un brivido freddo: allora questa newsletter è per te.

SOFTWARE DA INCUBO

Lavoro da oltre 25 anni nel settore dello sviluppo software e ho avuto modo di ricoprire una lunga serie di responsabilità, ruoli e compiti legati al ciclo di vita del software, in vari ambiti: da sviluppatore a manager, dalla modellazione al testing, da dipendente a libero professionista, dal software gestionale a quello aerospaziale, dall'ambiente safety-critical a quello business-critical.

Bene, sapete cosa mi è rimasta impressa in tutti questi anni di esperienze internazionali?

segue a pagina 2

La sensazione di impotenza nel vedere come il software, benché onnipresente in qualunque aspetto della nostra vita, in realtà sia una disciplina completamente sottovalutata, sottostimata e bistrattata ma di assoluta e sempre crescente importanza e soprattutto pericolo.

Ormai il codice scritto da qualche sviluppatore in giro per il mondo si trova ovunque: dalla lavatrice connessa in Wifi ad Internet e a tutti gli elettrodomestici di casa, al freno a mano elettronico collegato a vari sensori della propria auto che tra poco guiderà da sola, all'aereo che atterra nella nebbia totale o all'apparato medicale guidato da un tecnico esperto che sta effettuando una TAC.

Ma il software mai come ora può determinare il successo o il fallimento di un'azienda, la redditività o l'esaurimento del budget, nonché mandare all'aria intere linee di prodotti o aziende e magari infine provocare morti e feriti, nel caso dell'ambito safety-critical. E la coscienza del ruolo fondamentale della progettazione del software, il cui fallimento può provocare danni incalcolabili nonché morti e feriti, invece di aumentare sta diminuendo.

Certo, ci sono sicuramente tecnici che hanno una visione globale molto completa e strategica, ma spesso non sono nel ruolo giusto per poter intervenire ed ottimizzare il processo. Così come ci sono manager e CEO illuminati, di provenienza tecnica o che comunque si affidano a consulenti esperti, ma la realtà è che quasi tutti gli altri tendono a considerare il software come un male necessario e poco più che inutile, dove sicuramente spendere "il meno possibile" per cui qualunque investimento nella ricerca della qualità viene visto come irritante se non almeno differibile.

Fa niente che studi internazionali su migliaia di progetti dimostrino come investire nella qualità dello sviluppo abbia un ritorno dell'investimento di 15€ per ogni euro investito: si parla del 1500%. No, meglio risparmiare perché "noi siamo bravi" e non abbiamo avuto problemi finora. O con la scusa che comunque la parte fondamentale è l'hardware, il design, la prestazione, il brand ed il software è quasi invisibile.

Cosa che sappiamo non vera: vediamo spuntare pannelli touch-screen, ricevitori GPS e collegati in rete in ogni dove… dagli ascensori ai forni a microonde, dalle biciclette in free-sharing ai dispositivi bio-medicali sono tutti pieni zeppi di software. Così come sentiamo nella cronaca nera incidenti mortali dovuti a malfunzionamenti software su automobili, defibrillatori, treni. O fallimenti di intere aziende dovute a programmi impazziti che effettuano milioni di transazioni folli in Borsa.
E tu pensi che questo sia servito a qualcosa? Credete che ci sia più spontanea consapevolezza da parte delle PMI o delle grandi aziende relativa al ruolo del software e alle misure per ottimizzarne il ciclo di vita ed alzarne la qualità?

Ve lo dico serenamente: il 90% delle aziende (ad essere generosi) che investono in linguaggi moderni di programmazione, in modelli avanzati di sviluppo, in tool che migliorano l'efficienza del test, sono in realtà dovuti come dico sempre ad una semplice pistola puntata alla testa:

LA CERTIFICAZIONE!

Eh già: un aereo per poter volare, una macchina per circolare su strada, un treno per portare merci o passeggeri ed i dispositivi medicali sono tutti accomunati da alcuni fattori. Prima di tutto: sono sistemi safety-critical, dove un malfunzionamento potrebbe essere altamente rischioso in quanto è in gioco la vita umana.

Per cui accade un fenomeno ben conosciuto: un'Autorità statale studierà primo o poi il fenomeno mettendo inseme in un panel i migliori esperti del settore, che scriveranno un Safety Standard, uno standard di sicurezza per le persone che utilizzano quella categoria di prodotti che rende il metodo di sviluppo software un processo il più possibile ingegneristico:

- ***scientifico***
- ***rigoroso***
- ***ripetibile***
- ***formale***
- ***poco soggetto alla soggettività e creatività umane***

Negli anni sono nati tutta una serie di standard ***SAFETY***

CRITICAL, con sigle ben note agli addetti ai lavori, come:

DO-178B/C: safety standard per lo sviluppo di software Avionico

ISO-26262: safety standard per il mondo Automotive

IEC-61508: safety standard in ambiente Industrial Automation

IEC-50128/CENELEC: safety standard ferroviario

IEC-62305: safety standard per apparati biomedicali

Lo Stato o l'Unione Europea imporranno il rispetto di questi standard come condizione indispensabile per tutti questi prodotti per poter circolare o comunque essere utilizzati attraverso un percorso obbligatorio e pieno di verifiche formali, con Autorità Statali che vigilano sul rispetto delle specifiche di sicurezza.

L'obiettivo finale è quello di raggiungere la conformità finale sotto forma di Certificazione, prima di poter essere venduto e utilizzato dal pubblico.

LA FAMOSA PISTOLA PUNTATA ALLA TEMPIA

Che costringerà presto o tardi dirigenti, manager, project-leader e tecnici delle aziende di questi settori a correre ai ripari e a studiare e adottare le migliori tecniche di progettazione, sviluppo e test del software finalmente contribuendo a renderlo sicuro.

Tutto questo anche se spesso tardivo è molto utile, perché comunque darà lo slancio ad investire, misurare il miglioramento e verificare il risparmio (ricordate il rapporto 15:1?). Queste tecniche safety-critical verranno spiegate in questo blog, insieme ad altre metodologie interessanti.

L'obiettivo di questo blog, articoli, ecc. sarà sicuramente quello di aiutare aziende che già debbono sottostare alle forche caudine della Certificazione, anche se loro si sono già mosse o si stanno muovendo ad adottare questo approccio, per cui questo blog sarà comunque un interessante approfondimento di alcune tecniche più recenti ed un importante ripasso.

Ma a chi è dedicata in particolare questa newsletter?

E' dedicato in realtà al restante 10% delle aziende: a quelle che, medio-piccole che siano, comunque non devono fare la Certificazione.

Niente pistola alla tempia.

Niente obblighi, procedure rigorose di fronte alle Autorità ed Enti competenti.

Niente bollini di Qualità per poter vendere.

Tutto il mondo di aziende o di progetti per cui un malfunzionamento, un "bug", provoca al massimo un fastidio, un disservizio, un malcontento. Un'app che si blocca sul telefonino facendoci perdere dei dati, un collegamento Wi-fi che non funziona, il sistema di intrattenimento dell'auto che si riavvia da solo.

Tutte cose irritanti, frustranti che non ammazzano nessuno ma che possono determinare il successo o la catastrofe di un prodotto, di un'azienda, un brand. Che modificano profondamente l'umore dei propri clienti ed utilizzatori, l'immagine di una società e dei suoi prodotti con tutte le conseguenze soprattutto economiche del caso. E per le quali ho studiato una soluzione efficiente e realizzabile da qualunque azienda, anche di una sola persona.

IL SOFTWARE E' CRITICO PER IL TUO BUSINESS?

La maggior parte di queste aziende per storia, per tradizione, per mancanza di una radicata cultura del software e della sua qualità spesso involontariamente ne sottovaluta il ruolo, non solo da un punto di vista tecnico ma anche e soprattutto dal fatto che si ritrova senza accorgersene immediatamente a bruciare budget migliaia se non milioni di euro in problematiche come ritardi nelle consegne, tempistiche falsate di uscita sul mercato, cancellazioni di prodotti, malfunzionamenti bloccanti, insoddisfazione degli utenti ed abbandono del brand.

Oppure gli effetti nefasti li ha ben presenti, ma non ha idea di come contrastarli efficacemente, con rapidità e senza enormi investimenti. Ma che ora ha un alleato e soprattutto un metodo in più.

Negli anni io ho cominciato ad applicare questo approccio rigoroso e scientifico in modo ragionato e adatto anche a realtà più piccole, ereditando tutto quello di indispensabile dal mondo Safety-Critical ed adattandolo ad aziende dove è importante la qualità ma anche la flessibilità, la velocità e i costi.

Ecco allora questo blog, frutto della mia storia personale e ricco di teorie e metodologie ed esempi concreti di applicazione, storie vere di successi e fallimenti, con argomenti tecnici quando necessario e rimando ad ulteriori approfondimenti, è adatto a chiunque si trovi a lavorare in un'azienda che per un motivo o per un altro non ha ancora raggiunto la maturità necessaria a comprendere ed auto-analizzare il proprio grado di maturità nello sviluppo, l'efficienza ed efficacia del ciclo di vita del software e di come intervenire con aggiustamenti a volte incredibilmente semplici e dal ritorno immediato in modo da migliorare tantissimi aspetti non solo tecnici ed informatici ma anche finanziari, di soddisfazione dei consumatori e di fedeltà al brand.

Se volete scoprire esattamente come ho fatto ad applicare metodi scientifici ed ingegneristici tipici di grandi aziende aerospaziali e ad adattarli ad aziende medio-piccole dove il team di programmazione è limitato a gruppi anche da solo 1 persona in su, riducendo costi e rischi e facendole arrivare prima sul mercato con un prodotto più stabile, siete nel posto giusto e questa newsletter fa per te.

Leggetelo, consigliatelo, fatelo circolare: ai tecnici, sviluppatori, designer che vogliono imparare qualcosa di nuovo o vedere un punto di vista globale e sinergico, ai manager che non sono riusciti a stare al passo con le nuove metodologie di design, modellazione e sviluppo, ai dirigenti che sanno di dover agire presto ed efficacemente per contrastare o prevenire problemi peggiori ma non sanno di chi potersi fidare e di come intervenire subito.
Sono convinto che apprezzerete questo mio approccio in quanto vi darà una visione organica e d'insieme a tutta una serie di problematiche che vi stanno erodendo i budget e vi suggerirà soluzioni per implementare delle contro-misure, attraverso varie tecniche ed un processo ben preciso:

IL METODO DELLO SVILUPPO EFFICIENTE DEL SOFTWARE (M.E.D.S.)

Bisogna agire adesso: ogni giorno che perderete a non essere nemmeno consapevoli di quanta distanza vi separa dall'avere un software efficiente e corretto ma soprattutto del percorso più o meno tortuoso per arrivarci, sono soldi persi. Tanti.

Direi allora che vale la pena leggere qualche articolo che vi aiuterà parecchio in questo itinerario.

Per ulteriori approfondimenti e sviluppo degli argomenti trattati, seguimi sul mio blog:

WWW.SOFTWARESICURO.IT

o scrivimi a:

INFO@SOFTWARESICURO.IT

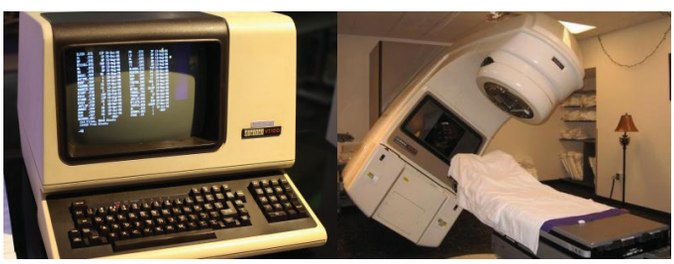
Ti invierò articoli tecnico-pratici su come rendere più efficiente il tuo Ciclo di Vita del Software e ti manderò Casi di Studio reali di Piccole/Medie Aziende o Grandi Imprese dove viene raccontata nei particolari l'applicazione pratica di questo metodo.

5° Comandamento Software: NON UCCIDERE

Il software è uno dei prodotti più complessi mai prodotti dall'uomo, senza paragoni. Non esiste prodotto letterario, programma scientifico, opera architettonica che possa eguagliare il numero di ore/uomo o anni/uomo necessari a produrre un software di grande complessità come quello che equipaggia un moderno aereo, un'automobile di ultima generazione, il computer o smartphone che state utilizzando, un social come Facebook.

Ed il software è talmente complesso che spesso sbaglia… anzi, uccide.Il software uccide le persone, brucia capitali enormi, fa saltare in aria le aziende, crea irreparabili danni di immagine

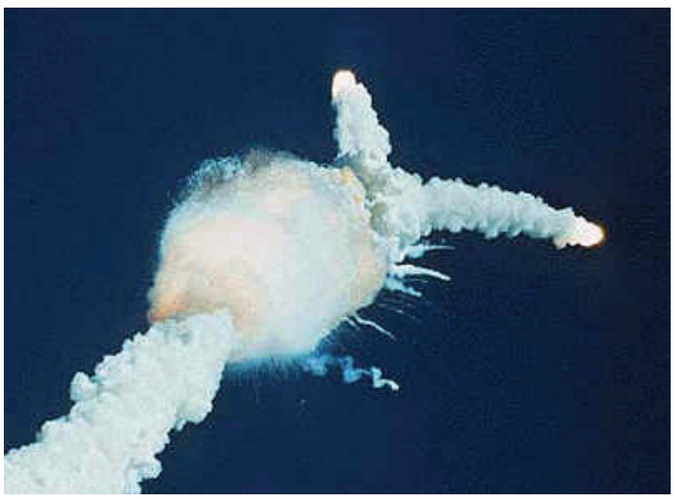
Nel 1982, si sospetta che la CIA abbia volutamente introdotto un bug (errore software) all'interno del codice di controllo della condotta di gas transiberiana in Russia. Per motivi di contro-spionaggio, gli USA hanno deciso di far esplodere tale condotta una volta operativa con il risultato di provocare la più grande esplosione non-nucleare della storia.



Tra il 1985 ed il 1987, un acceleratore di particelle chiamato Terac 25 ha provocato diverse morti in alcune ospedali. Basato su un design precedente, il nuovo apparato era stato dotato di un dispositivo di sicurezza basato su software invece che meccanico ma purtroppo

era stato programmato da un tecnico senza nessuna preparazione formale e senza criteri di safety.

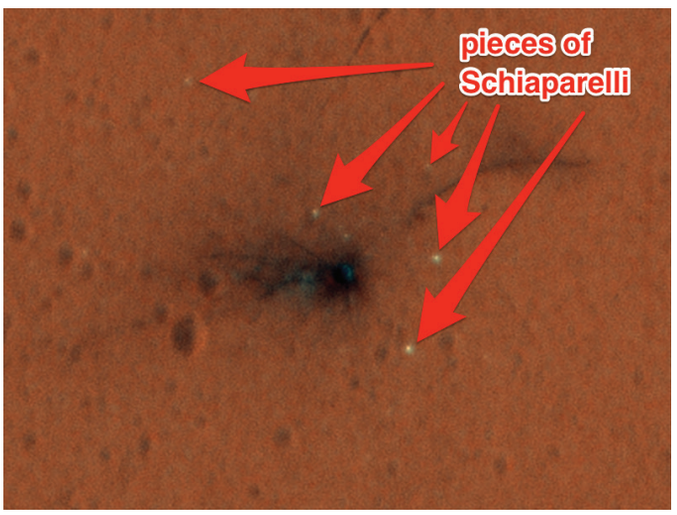
In alcuni casi di utilizzo maldestro, un bug molto sottile chiamato “race condition” ha provocato l’emissione di raggi X ad alta potenza senza uno schermo protettivo che colpivano direttamente il paziente uccidendolo o ferendolo gravemente.



Nel 1996, il razzo Ariane V è stato fatto esplodere in quanto oramai totalmente fuori traiettoria e senza controllo. Il software di navigazione inerziale della precedente versione, Arian IV, era stato ritenuto talmente affidabile da essere utilizzato senza modificarlo e soprattutto senza testarlo con i nuovi requisiti di velocità ed accelerazione del nuovo e più potente vettore. Un banale errore di conversione da 64 a 16 bit ha provocato un overflow che non gestito ha portato all’auto-distruzione con un danno di 370 milioni di dollari.

Nel 1999, il Mars Climate Orbiter della NASA ha oltrepassato il suo obiettivo quando ha cercato di spostarsi in orbita attorno al pianeta rosso. Si è poi polverizzato nell’atmosfera. La ragione: un gruppo di ingegneri che lavoravano alla navicella utilizzava le misure Metriche mentre un’altra squadra usava le unità Imperiali.

Il 1 Agosto 2012, un computer impazzito della Knight Capital ha cominciato ad inviare migliaia di ordini errati di compravendita di azioni al secondo, vendendo a prezzo di mercato invece che sfruttando la forchetta Bid/Ask e rimettendoci sistematicamente ad ogni transazione. In poco più di 30 minuti, l’azienda ha perso 440 milioni di dollari e ha sfiorato il fallimento.



Il 19 Ottobre 2016, la sonda spaziale Schiaparelli si è schiantata sulla superficie marziana nell’ambito della missione ExoMars di esplorazione di Marte in collaborazione tra ESA e Russia. Il paracadute si è aperto per solo 3 secondi invece che per 30 come avrebbe dovuto, con il risultato di arrivare troppo velocemente e di distruggersi all’impatto. Anche in questo caso, il software non ha saputo gestire delle condizioni eccezionali che si sono verificate.

E questi sono solo i danni macroscopici, quelli da centinaia di milioni e che diventano famosi per quello. In realtà, ci sono sicuramente episodi meno eclatanti dove comunque le aziende perdono non soltanto milioni di dollari ed euro, ma soprattutto perdono la fama, l’affidabilità, la posizione di mercato, la fiducia degli utenti e via discorrendo. Tutte cose che possono determinare

problemi non soltanto immediati a breve termine, ma un lungo ed inesorabile declino se non al fallimento.

Non preoccupatevi, c'è ancora di peggio: il processo di produzione del software, nella maggior parte delle aziende incluse le migliori, è quantomeno mal gestito se non fuori controllo e le conseguenze prima o poi diventeranno eclatanti e sotto gli occhi di tutti.

Ma cosa capita, nel frattempo?

Capita che, come un'emorragia interna multipla, il ciclo di vita del software faccia perdere "sangue" dagli organi aziendali in maniera inavvertita ed inesorabile, consumando quantità incredibili (e quasi mai misurate) di soldi, allungando a dismisura i tempi di consegna o di arrivo sul mercato, facendo silenziosamente crescere il malcontento dei clienti finali e via discorrendo.

Per anni ed anni, prima che qualcuno se ne possa accorgere. E di solito, quando ci si accorge che i propri "organi" interni sanguinano, comincia ad essere troppo tardi.

E QUINDI? COSA DOBBIAMO FARE?

Le azioni da fare essenzialmente sono:

Diffondere una Cultura Continua del Software in tutta l'azienda, in modo che ci sia una crescente consapevolezza dei rischi e delle incredibili opportunità di una sua gestione intelligente

Adottare un Metodo per il Ciclo di Vita del Software basato su uno SVILUPPO EFFICIENTE integrandolo con quello tradizionale aziendale dei propri prodotti venduti

Introdurre un ecosistema di Tool, Librerie di terze parti nella produzione del software in maniera accurata laddove siano realmente utili e non sia strategico il loro sviluppo interno

Attivare delle procedure di Verifica, Monitoraggio e Controllo per essere consapevoli in ogni momento dello stato di salute del vostro software e di conseguenza della vostra azienda

Valutare le Metriche adeguate per calcolare il Ritorno dell'Investimento di ogni azione intrapresa in questo processo in modo da rendere sempre più efficiente e sicuro il processo di sviluppo

I prossimi articoli saranno dedicati esattamente a questi capisaldi che potranno determinare la differenza tra il salvataggio e la crescita della tua azienda da una parte, oppure l'inesorabile declino ed aumento esponenziale del rischio dall'altra.

E TU, da che parte VUOI stare?

Per ulteriori approfondimenti e sviluppo degli argomenti trattati, seguimi sul mio blog:

WWW.SOFTWARESICURO.IT

o scrivimi a:

INFO@SOFTWARESICURO.IT

Ti invierò articoli tecnico-pratici su come rendere più efficiente il tuo ***Ciclo di Vita del Software*** e ti manderò ***Casi di Studio*** reali di ***Piccole/Medie Aziende*** o ***Grandi Imprese*** dove viene raccontata nei particolari l'applicazione pratica di questo metodo.

LA CULTURA, QUESTA SCONOSCIUTA! (PRIMA PARTE)

La cultura è importante, ce lo ripetono fin da piccoli. Studiate e diventerete qualcuno. Specializzatevi e farete successo. Quante volte ce lo siamo sentiti dire? E bene o male abbiamo sempre adottato questo approccio alla nostra vita lavorativa: tramite un corso formale di studi, universitario e successivo, oppure tramite corsi professionali e infine tramite auto-apprendimento.

Siamo degli esperti nel nostro settore, ne sappiamo più degli altri, più dei nostri collaboratori e dei nostri concorrenti, o almeno si spera. Non ci fermiamo mai: non è che finito il ciclo di studi abbiamo appoggiato i libri in biblioteca a prender polvere. Mai quanto vorremmo, ma cerchiamo di rimanere aggiornati: tramite corsi online o in aula, riviste di settore, blog specializzati, newsletter, webinar… le sorgenti di sapere sono tante, fin troppe se pensiamo al tempo che abbiamo.

Ma lo sappiamo: la concorrenza è in agguato e se non ci formiamo noi e rimaniamo al passo coi tempi, anzi davanti agli altri possibilmente, è un attimo rimanere indietro prima con il know-how e poi con le vendite. E di conseguenza, con la leadership o comunque la posizione di mercato. Quindi non mettiamo neanche in discussione di dover essere in continua formazione, in costante aggiornamento sulle tecnologie riguardanti il nostro specifico settore, qualunque esso sia.

E per le tecnologie che non sono sotto il nostro diretto controllo, che non fanno parte del nostro core-business, ci affidiamo ad esperti interno o esterni: un commercialista per gli aspetti fiscali, un avvocato per quelli legali, un'azienda di logistica esterna per i trasporti. E' un approccio normale e corretto: la mia tecnologia, arte, servizio vero e proprio che vendo e che mi differenzia sul mercato lo curo personalmente come manager o imprenditore, o con un team specializzato. Tutto quello che non è strategico, lo delego a funzioni aziendali preposte o a società esterne specializzate.

Certo, un po' ne voglio comunque capire. Per non farmi fregare. Per poter scegliere strategicamente. Per capire quello che mi viene offerto. Per contestare quando le cose non vanno. Giusto un po', senza diventare specialista: gli esperti di tutte le competene non strategiche li assumo o li metto sotto contratto, ma intanto un po' mi informo.

E il software? Scusa, che male ti ha fatto esattamente il software?

Pensi che il software si faccia da solo?

Che sia sufficiente "mio cuggino che ne sa di computer"?

Che il team di ingegneri meccanici, elettronici, hardware siano tutti comunque in grado di "sviluppare"?

Sei sicuro che in un'azienda di decine di persone, con team di progetto magari di 10-15 persone che curano tutti gli aspetti elettronici, meccanici, elettrici, prestazionali, estetici siano sufficienti 1-2 softwaristi "perché il software da sviluppare tanto è poco"?

Pensi che uno "smanettone", come vengono chiamati i programmatori, si aggiorni da solo alle nuove tecnologie?

Pensi veramente che il ruolo crescente del software in termini di funzionalità non ne faccia anche crescere la criticità?

Ritieni inutile avere anche solo una conoscenza di base perché tanto ci sono gli esperti?

Se hai risposto un secco NO a tutte queste domande, forse questo blog non fa veramente per te. Ma se hai qualche dubbio… continua a leggere.

Eh no… la Cultura del software è tanto importante quanto il software stesso. Se il codice sviluppato è diventato così strategico e critico nei tuoi prodotti tanto da determinare la maggior parte delle caratteristiche e delle performance della vostra offerta, se un malfunzionamento può portare ad un richiamo di migliaia di prodotti, se persone rischiano la propria salute o la vita in caso di malfunzionamento, se il business dei tuoi clienti potrebbe rallentare o bloccarsi in caso di bug non scoperti prima, non hai altra scelta:

la Cultura del software deve essere diffusa, profonda ed in continuo aggiornamento, per tutta la vita della tua azienda

Non basta assumere persone già esperte: il mondo informatico è in continua evoluzione, devono continuare a poter studiare, sempre, ogni anno. E non è sufficiente che siano delegate solo a loro certe strategie e decisioni delicate perché nessuno ne capisce: tutto il management deve poter capire e giustificare le scelte fatte. Non dico di saper programmare: ma almeno avere un'idea della tecnologia esistente e dei suoi vantaggi e relativi rischi.

Non starò ovviamente a parlare delle tecniche tradizionali di formazione come corsi in aula e online, workshop, webinar, ecc. perché di esempi ne é piena la Rete, ma di Cultura in senso lato:

nuove metodologie, tecniche, iniziative di apprendimento e diffusione della cultura che possano in qualche modo perseguire il fine di rendere tutti quanti più consapevoli di vantaggi e problemi legati allo sviluppo software

Il metodo che spiego ai miei clienti, adatto sia ad un normale ambito di sviluppo orientato alla Qualità ed al Business, che alla Certificazione, chiamato M***ETODO DELLO SVILUPPO EFFICIENTE DEL SOFTWARE (M.E.D.S.)***, fornisce un approccio efficente, un supporto, delle procedure collaudate e una consulenza soprattutto su queste metodologie avanzate. Ognuna di queste può garantire un'efficienza decisamente superiore, risultati molto più veloci e miglioramenti drastici dell'efficienza del ciclo di vita dello sviluppo del software e della sua eventuale certificazione o comunque entrata sul mercato.

Vediamo quali sono alcuni di questi metodi:

Task Force

Si prende un gruppo di specialisti selezionati, scelti in base alle competenze, esperienza, eterogeneità ed altri criteri e li si fà lavorare insieme a stretto contatto (nella stessa stanza se possibile o attraverso strumenti di collaborazione remota) per la risoluzione di un problema specifico, urgente, complesso. Problemi che con le tecniche tradizionali ad esempio di segmentazione del progetto in pacchetti, unità di lavoro, gruppi ecc. non sono risolvibili perché attraversano varie competenze e gruppi, o sono particolarmente gravi e profondi.

La task force di solito ha obiettivi molto ben precisi, da stabilire all'inizio e si scioglierà subito dopo aver fornito una identificazione ed eventuale soluzione al problema o comunque dopo un certo lasso di tempo, per evitare di incancrenire ulteriormente la situazione.

I risultati sono tremendamente efficienti: è stato dimostrato che una task-force anche molto piccola, da 2 a 5 persone, con una guida decisa, un focus preciso e comunicazione molto frequente tra i membri del team, ha un rendimento in termini di produttività e problem-solving anche 10 volte più elevato delle stesse persone che lavorano individualmente o in maniera tradizionale.

Le caratteristiche quindi sono:

- team piccoli (2-5 persone)
- guida da parte di un consulente esterno
- personalità e competenze eterogenee focus preciso (problema specifico, consegna urgente, ottimizzazione performance, …)

- localizzazione stretta (stessa stanza se possibile, collaborazione online stretta)
- durata limitata nel tempo (pochi giorni, massimo qualche settimana)
- risultato: soluzione del problema o sua identificazione e strategie per risolverlo

Tramite il ***METODO DELLO SVILUPPO EFFICIENTE DEL CODICE*** aiutiamo a formare delle Task Force aziendali molto efficienti sia contribuendo direttamente con degli esperti di settore, sia guidando le aziende a formare e condurre le attività secondo degli schemi già collaudati ed efficienti.

Technical Architecture Group

Il Technical Architecture Group (TAG) o altro nome simile (Focus Group), è una struttura permanente all'interno di un'azienda che si occupa di gestire le problematiche tecniche appunto di un progetto di medie o grosse dimensioni. E' composto da un numero limitato di esperti, scelti tra i migliori elementi per competenza ed esperienza, provenienti dai vari gruppi di progetto e spesso anche da altre sedi.

I vantaggi sono enormi: vengono seguite le problematiche di progetto più generali come quelle architetturali, strutturali, di performance, ecc. a livello centrale e condiviso, lasciando poi ai singoli gruppi di lavoro la loro implementazione.

L'attività si svolge prevalentemente online tramite mail, forum, chat con incontri periodici con cadenza es. mensile. Sono invitati i membri permanenti del TAG ed eventualmente i Team Leader dei vari gruppi di volta in volta coinvolti in problemi o argomenti specifici.

Caratteristiche:

- *team medi (5-10 persone)*
- *guida da parte di un consulente esterno o di una persona interna a rotazione*
- *persone competenti ed esperte assegnate per regione, gruppo, competenza*
- *focus su temi generali del progetto (architettura, design, problemi complessi, performance, …)*
- *attività online ed incontri a cadenza circa mensile (da adattare alle esigenze es. alla partenza del progetto sarà più spesso)*
- *durata: per tutta la vita di progetto*
- *risultato: minute degli incontri, documenti tecnici, specifiche architetturali, linee guida, standard di sviluppo, …*

Gap Analysis

La Gap Analysis è un'attività che risulta fondamentale in tutta una serie di situazioni in cui è necessaria una transizione più o meno profonda e rapida di un progetto o di un'intera azienda (se piccola) per modificare, adattare e rendere più effficiente e sicuro un medoto di lavoro.

Tipici esempi che richiedono un rapido e profondo adattamento del modo di lavorare, del ciclo di vita del software, dei tool, del processo, del metodo ecc. sono le famose Certificazioni Safety-Critical (vedi articolo apposito su DO-178B/C, ISO-26262, IEC-61508, IEC-62304, CENELEC), passaggio a modelli di maturità software come il CMMI (Capability Maturity Model Integration), altri tipi di conformità, assessment e via discorrendo.

Si tratta in ogni caso di profondi cambiamenti che l'azienda deve adottare in tempi piuttosto rapidi per potersi adattare velocemente alle richieste ad esempio di un committente, di una gara d'appalto o di una certificazione necessaria per poter legalmente vendere un prodotto.

Qua il discorso è molto semplice: va fatto di tutto per non perdere tempo e risorse, realizzare questo cambiamento velocemente e con il minor impatto, identificando le procedure, i documenti, i prodotti, i tool che si possono salvare e modificare e quelli che vanno implementati da zero.

La Gap Analysis è semplicemente il modo più veloce per andare da A (situazione attuale) a B (situazione desiderata) e consente un risparmio economico e di risorse notevole.

Mancando per definizione stessa le competenze interne per svolgere il nuovo compito desiderato, si realizza quindi di solito rivolgendosi a consulenti esterni o a persone interne all'azienda ma di altri progetti/sedi. Si svolge sotto forma di indagine serrata, **un assessment, su tutte le tematiche, le metodologie, i processi, i documenti, i requisiti, il codice sorgente, i tool ed in generale su tutti gli aspetti che saranno oggetto del cambiamento.**

Caratteristiche:

- personale esterno altamente competente (o da altre sedi/progetti esperti)
- coinvolgimento di tutti i ruoli aziendali e di tutti i metodi, processi, documenti, tool, codice, in poche parole: TUTTO
- durata limitata (3-5 giorni)
- risultato: report iniziale sulla percentuale di compliance generale e specifica per argomento, identificazione di mosse urgenti da implementare

- report completo di tutte le aree analizzate e di tutto quello che è possibile conservare, quelllo che va modificato e quello che va implementato da zero (in termini di processi, documenti, tool ecc.)
- personale esterno altamente competente (o da altre sedi/progetti esperti)
- coinvolgimento di tutti i ruoli aziendali e di tutti i metodi, processi, documenti, tool, codice, in poche parole: TUTTO
- durata limitata (3-5 giorni)
- Risultato: report iniziale sulla percentuale di compliance generale e specifica per argomento, identificazione di mosse da implementare immediatamente (Quick Win), report completo di tutte le aree analizzate e di tutto quello che è possibile conservare, quelllo che va modificato e quello che va implementato da zero (in termini di processi, documenti, tool ecc.)

All'interno del ***METODO DELLO SVILUPPO EFFICIENTE DEL CODICE (M.E.D.S.)*** abbiamo le competenze e gli esperti per condurre Gap Analysis ad ogni livello, per progetti Business, Security e Safety-Critical, in accordo a tutti i principali standard di Certificazione o di Qualità.

Mezz'ora creativa

Questa è una tecnica molto semplice ma anche molto redditizia in termini di risultati a lungo termine che ho imitato anni fa dai primi approcci simili di Google. Le regole sono molto semplici:

mezz'ora alla settimana in cui chiunque (sviluppatori, designer, tester, manager) partecipa ad una riunione ad argomento tecnico totalmente libero, divisa in un quarto d'ora di esposizione, un quarto d'ora di discussione si può parlare di linguaggi, algoritmi, tecniche di programmazione, sistemi operativi, metodi e processi, cicli di vita, case study, grandi fallimenti del software e bug, qualunque cosa relativa al software preso dalle news o dall'esperienza personale una persona alla volta prenota questo spazio in cui espone in un quarto d'ora il suo argomento, anche senza dover evidenziare applicazioni concrete o applicabilità a brevissimo termine su progetti esistenti. Tempo massimo per preparare l'argomento: mezz'ora.

dopo l'esposizione, altri 15 minuti di discussione aperta in cui si parla della validità dell'argomento, sua eventuale applicabilità e prossimi passi.

E' incredibile come dare totale libertà di argomento alle persone, ma con dei vincoli temporali stretti per non ostacolare le normali attività lavorative, scateni la fantasia verso soluzioni tecniche avanzate, innovative ed impensabili adottando i normali processi aziendali. Questa libertà di iniziativa personale poi dà a chiunque la possibilità di coltivare le proprie idee e di sentirsi ascoltato. Se anche solo un'idea su 10 trova uno sbocco pratico concreto, un miglioramento, una maggior consapevolezza, il risultato sarà straordinario rispetto al pochissimo tempo impiegato.

Caratteristiche:

- aperto a chiunque voglia partecipare ed ascoltare o contribuire
- durata ridotta (mezz'ora per preparare l'intervento, mezz'ora per presentare e discutere) focus su temi liberi senza riscontro immediato.
- Risultato: aumentare la partecipazione ed il coinvolgimento di tutti anche in ruoli minori, la creatività e la circolazione di idee con un minimo investimento, senza trascurare la possibile ricaduta in termini di implementazione su progetti reali.

Hackathon

Un altro modo di scatenare la fantasia, creatività e produttività delle persone in un'attività pratica con risvolti immediati è l'Hackathon. Nata nel 1999 dentro aziende come Sun ed OpenBSD, si tratta di una specie di "competizione", in cui uno o più gruppi si ritrovano per un periodo alcune ore o alcuni giorni, per risolvere dei problemi pratici di programmazione, algoritmi, o comunque idee costruttive in generale. Gli eventi possono essere privati ed aziendali, oppure pubblici all'interno di fiere, eventi o altro. Ovviamente, con le moderne tecnologie, è possible rendere pubblico quello che avviene all'interno del team e far partecipare persone da tutto il mondo con tecnologie di web conference.

Le persone si dividono in team che vanno da 2-3 a più persone e possono concorrere in team diversi allo stesso tema, in una vera e propria competizione alla soluzione migliore, oppure suddividendo un argomento complesso in sotto-argomenti. I ruoli tra i partecipanti allo stesso team si dividono, a seconda della specialità, esperienza ecc. e ci sarà anche qualcuno che si occuperà della presentazione finale, magari anche di grafica e logo. A seconda del tempo o della complessità, la soluzione dovrà essere solo studiata e implementata a livello di prototipo, oppure dovrà essere fornita una versione già funzionante.

Alla fine in una cerimonia aziendale privata o pubblica, una giuria assisterà alle varie presentazioni e premierà il team che ha presentato la soluzione migliore secondo i criteri prestabiliti. I premi possono essere solo virtuali, una classifica, ecc. o anche in denaro vero e proprio. Anche in questo caso, la produttività e la creatività hanno uno slancio vertiginoso permettendo di concentrare i migliori cervelli che riescono

a mettere a fuoco senza interruzioni lavorative le proprie capacità su problemi concreti, spesso riuscendo a risolvere cose che avrebbero richiesto mesi/uomo.

Caratteristiche:

- team piccoli (2-5 persone)
- personalità e competenze eterogenee
- diversi ruoli specifici assegnati (leader, designer, sviluppatore, presentatore)
- focus preciso (problema specifico, ottimizzazione performance, …)
- localizzazione stretta (stessa stanza se possibile, collaborazione online stretta) durata limitata nel tempo (pochi giorni) risultato: presentazione di una soluzione del problema e strategie per risolverlo

Con il ***METODO DELLO SVILUPPO EFFICIENTE DEL CODICE*** abbiamo organizzato o contribuito ad organizzare diversi Hackathon all'interno della nostra azienda o di quelle dei nostri clienti, sviluppando alcuni trucchi e consigli per renderlo ancora più rapido ed efficiente.

Per ulteriori approfondimenti e sviluppo degli argomenti trattati, seguimi sul mio blog:

WWW.SOFTWARESICURO.IT

o scrivimi a:

INFO@SOFTWARESICURO.IT

IL SEGRETO SU COME BATTERE LA CONCORRENZA DEI PAESI LOW-COST

Oggi vincere la sfida contro i paesi emergenti low-cost nel campo della produzione di software e firmware non è cosa facile. Quasi tutte le PMI, dopo la crisi del 2008, hanno subito profondi cambiamenti e hanno dovuto reinventare un metodo diverso per sopravvivere a queste potenti economie in espansione.

Dopo la crisi molte piccole e medie aziende di produzione di sistemi dove la componente software ha un peso crescente, hanno sperimentato la guerra dei prezzi provenienti da queste nuove economie, che spesso sono generate non solo dal bassissimo costo della manodopera seppur qualificata di ingegneri e tecnici, ma anche da politiche di aiuto statale dei paesi stessi.

Oltre a subire questo attacco a livello economico, spesso aziende spregiudicate si sono copiate anche i nostri marchi, incuranti di ogni brevetto o registrazione. Ho un cliente che ha addirittura subito una causa legale negli USA perché il suo distributore ufficiale lo ha portato in tribunale per aver venduto direttamente sul mercato a clienti finali senza riconoscere una commissione al distributore. Alla fine il risultato è stato che erano stati dei cinesi a copiare talmente bene il prodotto, con funzionalità apparentemente identiche, marchio e imballo incluso, che persino il distributore non si era accorto della truffa!!

Chissà quante altre storie probabilmente anche tu stesso hai vissuto o sentito raccontare, aziende ridotte a dimezzare o chiudere definitivamente per via di questa concorrenza sleale, basata sullo sfruttamento delle persone abbinato a politiche di aiuto statale dei paesi stessi.

Quando sei investito da questo tipo di concorrenza, all'inizio provi a resistere facendo la cosa apparentemente più logica: abbassare il tuo prezzo! Si inizia sempre in questo modo, ovviamente abbassare il prezzo è l'arma più immediata che hai a disposizione, ma anche la più pericolosa.

Abbassare il prezzo: un'arma spuntata e pericolosa

Ho visto numerose aziende, leader di settore, utilizzare quest'arma del prezzo per tener duro, rinunciare a parte degli utili per poter mantenere il mercato, fino ad arrivare a non generare più utili sempre per restare al passo con la concorrenza.

Una volta che passi questa soglia, dopo qualche anno, da un lato hai ovviamente dei rincari dovuti al costo crescente degli sviluppatori software e firmware, ai tool e programmi di supporto che diventano obsoleti, al team di test che cresce fino ad eguagliare quello di sviluppo, alle tasse che aumentano senza sosta e ti tartassano e così via, dall'altra sei costretto ad andare sottocosto pur di far girare la baracca e pagarti almeno una parte dei costi fissi.

Come dicevo, ho visto aziende famose che in poco tempo, 5/7 anni si sono ridotte a partecipare a infinite gare al ribasso per vincere un bando, un appalto, oppure peggio ancora sono fallite. In molti casi erano anche aziende solide con un certo patrimonio, e hanno investito tutti i soldi solo nell'arma dell'abbassare il prezzo per tener duro!

IL CURATORE FALLIMENTARE

Ho un caro amico che come lavoro fa il curatore fallimentare, una figura senz'altro molto professionale. Quello che però speri è di non avere mai bisogno di lui, nel corso della tua vita. Ogni tanto ci troviamo davanti ad un bicchiere di bollicine, in virtù della nostra amicizia e, fortunatamente, non per i suoi servizi.

Come spesso accade, dopo i soliti 10 minuti in cui si chiede notizie delle rispettive famiglie, poi si cade sempre sul discorso lavoro. Rimango perennemente affascinato dalla sua attività: deve, in parole semplici,

ricostruire a ritroso la vita economica dell'azienda per cercare di capire dove sono stati commessi gli errori e cosa ha portato al fallimento. Una volta ricostruito questo, deve capire se esiste la possibilità di rimettere in pista l'azienda o parte di essa, vendendola ad un altro imprenditore per non arrivare a fare il triste spezzatino anche se spesso accade.

Essendo curioso, faccio parecchie domande, ed una di quelle che mi interessa è quanti anni deve retrocedere per vedere l'azienda in questione fare utili prima del fallimento.

I 4 errori delle aziende di software che ti porteranno al fallimento certo

Mediamente se si torna indietro di 5/8 anni, l'azienda in questione stava ancora facendo utili, e, molte volte, aveva anche un buon patrimonio, e purtroppo con l'avvento della concorrenza asiatica ha iniziato a fare le seguenti 4 mosse:

- ***ABBASSARE I PREZZI DI VENDITA***
- ***NON INVESTIRE PIÙ IN RICERCA E SVILUPPO***
- ***CERCARE NUOVI MERCATI ALTROVE***
- ***INSISTERE A PRODURRE LO STESSO SOFTWARE ALLO STESSO MODO***

Come dicevamo, ***ABBASSARE I PREZZI DI VENDITA*** dei propri prodotti, del software, dei servizi è una scorciatoia che non porta da nessuna parte a meno che non si trovi il modo di abbassare altrettanto i costi di produzione che invece aumentano.

Il secondo punto è ***NON INVESTIRE PIÙ IN RICERCA E SVILUPPO***: tipicamente si punta tutto sul fatto di concentrarsi sulle competenze, sui tool, sui prodotti e sulle tecnologie esistenti , poiché meglio di così non si può avere come costi di R&D e di personale, dimenticandosi che il mondo tecnologico è in rapidissima evoluzione e le tecnologie non durano per sempre e, ancora più importante, si continua a scrivere software e soprattutto a testarlo e rilasciarlo sul mercato e ai clienti senza adeguato controllo qualitativo e con logiche vecchie.

Questo punto di ***CERCARE NUOVI MERCATI***, non è in sé sbagliato, se ben strutturato è un ottimo punto, però non bisogna aspettare di essere alla frutta per sperare in qualche nuovo settore dove c'è meno concorrenza se non addirittura in un "paese delle meraviglie" dove i clienti pagano il 30% in più i propri prodotti. È un ottima strategia se avviata in tempi non sospetti, perché i risultati non arrivano subito!

Infine il punto ***INSISTERE A PRODURRE LO STESSO SOFTWARE ALLO STESSO MODO***: questo punto è probabilmente il più importante, è quello che vedo spesso fare la differenza tra aziende di successo o aziende in difficoltà.

COSA FANNO NEGLI ALTRI SETTORI?

Non necessariamente bisogna cambiare il prodotto finale in sé, magari il prodotto e le sue funzionalità possono rimanere uguali, ma il modo in cui lo si propone al mercato cambia, oppure il modo in cui lo si produce.

Prendiamo un mondo a noi lontano come quello del mobile, in quanto uno di quelli messi più a dura prova dal mercato: ancora oggi esistono aziende che nel mondo del mobile fanno veri e propri affari, vedi l'esempio di IKEA che, pur essendo nel mondo dei mobili, riesce ad essere un'azienda di successo.

Un altro esempio nel mondo dei mobili è un'azienda americana che si propone al mercato come "WE SELL ROOMS" ovvero "VENDIAMO STANZE". Cosa vuol dire? In questo caso propongono stanze già arredate di ogni dettaglio, per esempio il salotto, completo non solo del divano, ma di tutti gli elementi come lampade, soprammobili, tappeti, etc.

ESPANDERE O FOCALIZZARE?

Dunque, con il suggerimento di non INSISTERE A FARE LO STESSO PRODOTTO si intende che bisogna guardare la cosa da angolazioni diverse, cercare di "PACCHETTIZZARE" in settori dove si tende a vendere il prodotto singolo, come quello del mobile. Come si potrebbe fare nel mondo del software? Si potrebbe provare a fornire non solo software ma anche hardware, progettazione elettro-meccanica, servizi, certificazione, assistenza e supporto e così via. Di certo va bene se si hanno le competenze ma anche qua è imperativo non improvvisare.

Vale anche e soprattutto il contrario di "SPACCHETTIZZARE" e puntare sul focus: se nel settore normalmente si tenta di avere un grande range di prodotti, servizi e competenze come potrebbe essere un negozio di elettrodomestici che ha un ampio range di prodotti ed è uguale a tutti gli altri competitor, rispetto al negozio "LA CASA DEL RASOIO" che punta il focus su un prodotto.

Tornando nel settore della produzione di software, una delle chiavi di lettura che ho notato dopo qualche anno di TERREMOTO, è che le aziende che hanno iniziato nuovamente a produrre reddito, si sono in qualche modo buttate nel mondo del rendere più efficiente, snello e sicuro il ciclo di vita di produzione e supporto del software. Il segreto è in tre fasi:

1. aumentare la qualità
2. ridurre i rischi
3. tenere sotto controllo tempi e costi

Al giorno d'oggi queste sono le uniche tre armi che automaticamente tengono lontano la concorrenza dei paesi a basso costo, perché non hanno ancora le armi per vincere su questo terreno. per vincere su questo terreno.

Infatti soddisfare il cliente che è in cerca di prodotti affidabili dove il software è una risorsa che aggiunge flessibilità e potenza, non una minaccia nascosta pronta ad esplodere, è l'unica arma vincente!

QUESTO SISTEMA ESCLUDE AUTOMATICAMENTE LA CONCORRENZA DEI PAESI A BASSO COSTO!

Per poter fare questo però c'è un PROBLEMA: bisogna cambiare passo, ma se gli strumenti che hai a disposizione sono arretrati e inadeguati, la situazione può solo che peggiorare. Cosa voglio dire? Che per poter evitare di competere con i PAESI EMERGENTI, serve cambiare metodo di produzione del software.

A questo proposito, dopo anni e anni di sperimentazioni, innamoramenti e delusioni di tecnologie come **CMMI, 6-SIGMA, AGILE, MODEL-BASED, LEAN SOFTWARE DEVELOPMENT (LSD),** ecc. che sembravano risolvere tutti i problemi ma alla fine ne creavano altri, abbiamo cominciato a sviluppare quella che è un po' la somma dei migliori vantaggi di tutti, lasciando perdere tutto quello che è inefficiente, costoso o inutile per le aziende che non devono sottostare a rigide regolamentazioni come quelle aerospaziali.

E abbiamo creato lentamente quello che poi è diventato il ***METODO PER LO SVILUPPO EFFICIENTE DEL SOFTWARE (M.E.D.S.),*** una strategia completa per tutti il ciclo di vita di un prodotto Safety- o Business-critical, studiato appositamente per soddisfare lo sviluppo, la validazione e verifica di software di alta qualità, con tempi di produzione veloci e prevedibili.

LA RICETTA É: UN METODO DI SVILUPPO SOFTWARE CHE SIA NELLO STESSO TEMPO ***FLESSIBILE E PRATICO COME AGILE/LEAN*** *MA CHE SIA* ***SICURO ED AFFIDABILE COME UNO STANDARD SAFETY-CRITICAL***

Il metodo giusto si chiama M.E.D.S. ed è basata sull'adozione dei migliori principi ispiratori di metodologie come AGILE, DO-178C, LEAN SOFTWARE DEVELOPMENT ecc. andando a superare i limiti specifici come le eccessive iterazioni, la mancanza di documentazione e la scarsa affidabilità soprattutto iniziale in ambienti molto critici, in una nuova e moderna soluzione che risolve i grandi dilemmi di chi si trova a produrre software nel nuovo millennio.

UN METODO CHE TI PERMETTE DI ELIMINARE LA CONCORRENZA DIRETTA CON I PAESI A BASSO COSTO!

Ecco subito riassunti i punti di forza di questa nuova metodologia:

- *è un approccio* ***SCIENTIFICO E RIGOROSO*** *alla produzione di software moderno, sicuro e affidabile*
- *allo stesso tempo, è un metodo* ***SNELLO ED EFFICIENTE*** *che si concentra solo sulle attività indispensabili e con immediato e misurabile ritorno dell'investimento*
- ***AUTOMATIZZA*** *in maniera tremendamente efficiente tutto quello che è ripetitivo e meccanico*
- ***ABBATTE I TEMPI*** *di vari ordini di grandezza per uscire prima col prodotto nel mercato*
- *consente di misurare e migliorare la* ***QUALITÀ*** *del codice in maniera* ***MISURABILE E RIPETIBILE***
- *tiene sotto controllo tutti I* ***COSTI*** *tagliando tutte le attività inutili e ridondanti*
- *azzera quasi completamente* ***I RISCHI DI ERRORI****, mortalità infantile, rigetto da parte dei clienti grazie all'eliminazione immediata dei bug durante lo sviluppo*
- *aumenta* ***AFFIDABILITÀ E REPUTAZIONE*** *non solo del software ma di conseguenza dei prodotti e di tutta l'azienda stessa*

Questo metodo è stato studiato per rispondere ad una precisa domanda del mercato: *produrre software più sicuro, più stabile e di alta qualità andando a ridurre nello stesso momento tempi e costi*

Oltre a ciò, avrai un vantaggio competitivo nel rispondere in tempo RECORD alle richieste del cliente, quelle richieste che i paesi emergenti non riescono a soddisfare.

IL M.E.D.S. HA IL Total Cost of Ownership (T.C.O.) PIÙ BASSO PER LINEA DI CODICE DELLA CATEGORIA!

Ora tocca a te!

Prova a rispondere alle seguenti 3 domande, prenditi 10 minuti di tempo e ti suggerisco di rispondere scrivendole su un foglio: è molto importante scrivere le risposte rispetto che pensarle solo, scriverle è più difficile perché serve concentrazione, quella concentrazione che molte volte per mancanza di tempo non abbiamo, e che però ci fa prendere decisioni SBAGLIATE o peggio ancora non decidiamo affatto.

1. *Cosa cambierebbe nella tua azienda se adottassi un metodo di sviluppo innovativo per rispondere alle esigenze di qualità dei clienti in tempo record?*
2. *E cosa accadrebbe se continuassi ad adottare l'approccio che hai sempre usato?*
3. *Cosa rischi nella tua azienda se non fai nulla e posticipi le tue decisioni?*

Scrivimi le tue 3 risposte a ***info@softwaresicuro.it*** e potremo discutere insieme di una soluzione!

E ti ricordo che...

NON SOPRAVVIVE IL PIÙ FORTE, MA CHI SI ADATTA RAPIDAMENTE AI NUOVI SCENARI!