DevOps: SOLUZIONE INSOSPETTATAMENTE SEMPLICE A UN PROBLEMA APPARENTEMENTE IMPOSSIBILE?

A cura di Corrado Pomodoro, Senior Manager di HSPI

Introduzione

Come commenta l'editore IT Revolution Press del libro "*The DevOps Cookbook*", al momento, più che una *best practice*, DevOps è simile ad un movimento filosofico, sostenuto da una comunità di professionisti motivati dall'ambizione di replicare le eccezionali performance di casi come quello di Flickr, presentato alla conferenza O'Reilly Velocity, Web Performance and Operations Conference, 2009: "*release early, release often*", 10 full deployment del sito sui server della società. Ma non solo Flickr: l'approccio DevOps sembra essere alla base del successo di diverse altre aziende molto note come Spotify, Netflix, la stessa Facebook, e praticamente ogni social network.

Gene Kim, pluripremiato CTO e ricercatore (coautore del libro "*The Visible Ops Handbook*", "T*he Phoneix Project*" e, appunto "*The DevOps Cookbook*"), nella pubblicazione "*Top 11 Things You Need To Know About DevOps*" di IT Revolution Press, riassume la filosofia e i concetti alla base del movimento **DevOps**.

In questo articolo ne riportiamo i passaggi più significativi che descrivono la genesi del movimento, i principi su cui si basa, le sue relazioni con le best practice di sviluppo e IT service management, i benefici sottesi, integrandoli con alcuni importanti contributi dei principali attori del movimento stesso, Patrick Debois[1], John Allspaw, Paul Hammond ed altri.

Cosa è DevOps e da dove viene?

Il termine "DevOps" si riferisce al movimento professionale emergente che sostiene un rapporto di collaborazione tra lo sviluppo e le IT Operation, conseguendo un rapido flusso di lavoro pianificato (ad esempio con alti tassi di deployment), aumentando allo stesso tempo l'affidabilità, la stabilità, la resilienza e la sicurezza dell'ambiente di produzione. Perché sviluppo e IT Operation? Perché rappresenta in genere la catena del valore tra il business aziendale (in cui sono definiti i requisiti) e il cliente (dove viene percepito il valore).

Le origini del movimento DevOps si fanno comunemente risalire intorno al 2009, con la convergenza di numerose iniziative adiacenti che si rafforzano a vicenda:

- Il Velocity Conference, con la presentazione "*10 deploys A Day*" tenuta da John Allspaw e Paul Hammond;
- Il movimento "*Infrastructure as a code*" (Mark Burgess e Luke Kanies), il movimento "*Agile infrastructure*"(Andrew Shafer) e il movimento di "*Agile system administration*" di Patrick Debois;
- Il "*Lean Startup*" di Eric Ries;
- Il "*Continuous integration and release*" di Jez Humble;
- La diffusa disponibilità di tecnologie PaaS in cloud (platform as a service, come Amazon Web Services).

Il termine sembra essere stato coniato da Patrick Debois, consulente belga con anni di esperienza nell'ambito dei sistemi informativi. L'idea di base prende spunto dalle metodologie di sviluppo Agile, estendendole al mondo dell'esercizio dei sistemi. La comunicazione e l'interazione tra i gruppi è vitale perché i Sistemi Informativi creino valore per il business.

1. Patrick Debois è programmatore, manager, amministratore e collaudatore. Presentò per la prima volta i concetti DevOps alla conferenza Agile 2008 di Toronto. Nel 2009 ha organizzato il primo DevOpsDays; da allora promuove i concetti di collaborazione e scambio di idee tra i gruppi e spiega come ciascuno può aiutare l'altro per il raggiungimento di migliori risultati per il business.

In che modo DevOps differisce dall'Agile?

Un principio del processo di sviluppo Agile è quello di fornire software mediante incrementi piccoli e più frequenti, a differenza dell'approccio "big bang"

del metodo a cascata. Alti tassi di deployment si accumuleranno spesso di fronte alle IT Operations per il rilascio. Clyde Logue, fondatore di StreamStep, ha affermato: "L'Agile è stato strumentale per lo Sviluppo al fine di recuperare la fiducia del business, ma ha tralasciato l'IT Operation. DevOps è un modo per riguadagnare la fiducia del business rispetto all'IT nella sua interezza".
Nella presentazione di John Allspaw & Paul Hammond al Velocity 2009, si trova:

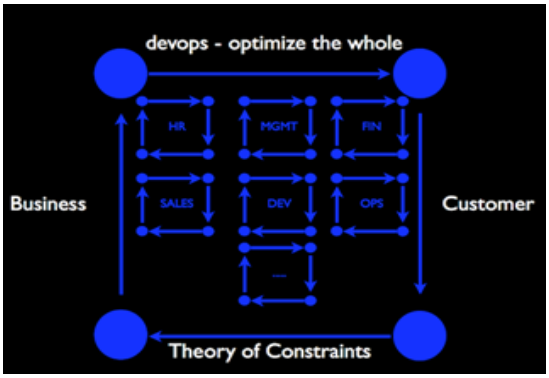
Dev's job is to add new features
Ops' job is to keep the site stable and fast

Ops' job is NOT to keep the site stable and fast
Ops' job is to enable the business

(**this is dev's job too**)

Patrick Debois spiega la giusta prospettiva per definire DevOps in un post[2] del suo blog (www.jedi.be) del maggio 2012. Per farlo, usa prima un commento di Damon Edwards: "*devops is not about a technology , devops is about a business problem*" e poi si rifà alla *Theory of Constraints* (TOC o Teoria dei Vincoli di Eliyahu Goldratt, la quale punta all'ottimizzazione del sistema nel suo complesso piuttosto che ambire all'ottimo locale, una volta chiarito e correttamente verbalizzato l'obiettivo comune.

In questi termini, ottimizzare il sistema nel suo complesso vuol dire indirizzare gli obiettivi di business, il problema del cliente piuttosto che il silo individuale (sviluppo o esercizio che sia), la "catena del valore"- si direbbe in termini "*lean*"- controllando le strozzature e il loro impatto sul *dev* e sull'*ops*, ovunque esse si presentino.



Nel post Patrick Debois introduce le due versioni:

- **devops**: collaborazione, ottimizzazione attraverso l'intera organizzazione; anche al di là dell'IT (HR, Finance...) e dei confini aziendali (fornitori);
- **devops** '**lite**': quando si intende focalizzare l'attenzione alla collaborazione tra 'dev' e 'ops'.

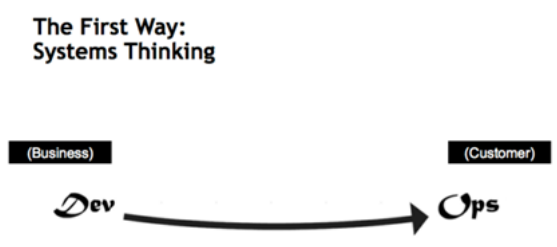
Esistono "best practice" DevOps?

L'aspetto sistemico implica che per risolvere un problema in ambito *dev* o *ops* possa essere necessario intervenire altrove piuttosto che su di essi; diventa pertanto impossibile descrivere in termini prescrittivi i passi necessari per risolvere un eventuale problema in ambito sviluppo/esercizio.
Nella maggior parte dei casi, osserva Patrick Debois, 'dev' riguarda la progettazione, lo sviluppo e 'ops' la produzione; nei due ambiti si sono sviluppate e diffuse diverse metodologie: Scrum e Kanban, ad esempio, nello sviluppo software; ITIL, Visible Ops in ambito produzione.
Queste metodologie non risolvono "strozzature" fuori dal loro ambito; è qui che entra in gioco DevOps, ricercando la collaborazione tra i diversi ambiti al fine di vedere il sistema nel suo complesso e ottimizzarlo.
Gene Kim cerca di chiarire come DevOps si ponga rispetto ad ITIL e alle prassi di IT service Management: "*Sebbene molti vedano DevOps come una negazione di ITIL e ITSM, ho un'opinione diversa: ITIL e ITSM sono ancora le migliori codifiche dei processi che*

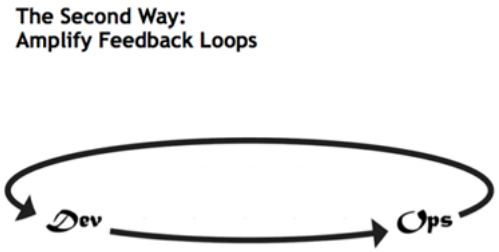
2. http://www.jedi.be/blog/2012/05/12/codifying-devops-area-practices/

sostengono le IT Operation ed effettivamente descrivono molte delle capacità necessarie per supportare un flusso di lavoro secondo il modello DevOps. Agile e continuous integration and release sono output dello Sviluppo ed input delle IT Operation. Per accogliere la cadenza più veloce delle release associata a DevOps, molti dei processi ITIL richiedono automazione (change, configuration release).
L'obiettivo di DevOps non è solo incrementare la frequenza dei rilasci, ma assicurare che avvengano senza causare caos e interruzioni di altri servizi, rilevando e correggendo velocemente eventuali incident. Ciò riconduce alle discipline ITIL di service design, incident e problem management."
Ancora Gene Kim compara DevOps a Visible Ops dichiarando: "i*o ho partecipato alla scrittura del "Visible Ops handbook" nel 2004 con Kevin Behr e George Spafford (il mio socio con cui ho scritto il libro a 4 mani "Il progetto Phoenix"). … uno dei concetti chiave di Visible Ops è la nozione di come controllare e ridurre il lavoro non pianificato. Io vedo DevOps come l'inverso, un modello per creare un flusso veloce e stabile di lavoro pianificato attraverso lo Sviluppo e le IT Operation. E comunque con un approccio più* ***olistico*** *all'eradicazione sistemica di lavoro non pianificato, indirizzando principi di* resilience engineering*, responsible management e riduzione del* technical debt." Tuttavia il dibattito è in corso, tanti sono i casi sottoposti ad esame per potere essere annoverati come prassi DevOps (buone o "best" si vedrà).
Gene Kim e Patrick Debois propongono uno schema di principio per tentare una catalogazione o codifica di queste prassi.
Lo schema prevede dei principi e dei pattern che li attuino.

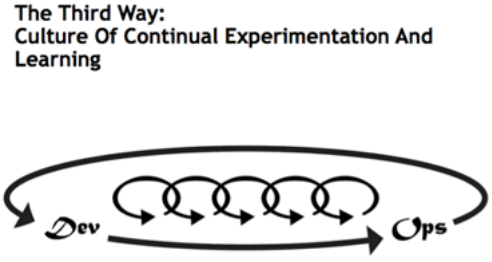
I tre principi (Three Ways) di DevOps



Il primo principio enfatizza la **performance dell'intero sistema**, in contrapposizione alla performance di uno specifico silo di lavoro o dipartimento (che sia una divisione o un singolo amministratore o programmatore). Il focus è su tutti i processi di business abilitati dall'IT, dall'identificazione dei requisiti, allo sviluppo, al passaggio in produzione, al delivery del servizio per il cliente.
Mettere in pratica il primo principio comporta non passare mai un difetto noto alle fasi successive di lavorazione, non permettere mai un ottimo locale a discapito dell'ottimo globale, perseguire una profonda comprensione del sistema (come insegnato da Deming).



Il secondo principio stabilisce il **loop di feedback da Ops verso Dev**. L'obiettivo di ogni iniziativa di miglioramento dei processo è di accorciare e amplificare i loop di feedback in modo che le **azioni correttive** siano pressoché **continue**.
Il principio include la comprensione e la risposta a tutti gli stakeholder interni ed esterni, accorciando e amplificando tutti gli anelli di feedback e integrando le conoscenze dove necessario.



Il terzo principio prevede la creazione di una cultura che favorisca la **sperimentazione continua** – che implica sapere **correre dei rischi e apprendimento da successi e fallimenti** – e la comprensione che la ripetizione e la pratica sono prerequisiti per l'eccellenza.

Sperimentazione e rischio assicurano la spinta continua al miglioramento, anche se ciò significa spingersi più di quanto non si sia mai fatto. Competenze assimilate con la pratica assicurano la capacità di tornare indietro quando ci si è spinti troppo oltre ed il rischio è troppo alto.

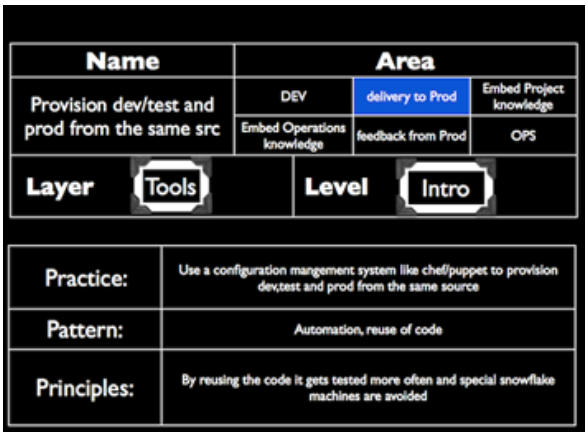
Aree DevOps

Le aree di modellazione delle prassi DevOps proposte da Gene Kim, Patrick Debois e i loro coautori, sono quattro:

- **Area 1:** Extend Development into Production: estendere la funzione di *continuous integration & release* (Jez Humble) alla produzione, integrando QA e information security, assicurando "production readiness" del codice e così via.
- **Area 2:** Create Production feedback into Development: *timeline* completa degli eventi di Sviluppo e IT Operation a supporto della risoluzione degli incidenti, self-service di sviluppo dove possibile, comunicazione che evidenzi come le decisioni locali impattano sul raggiungimento di obiettivi globali.
- **Area 3:** Embed Development into Operations: inserire lo Sviluppo nella catena di escalation della produzione, assegnando risorse di sviluppo alla gestione di *problem* e contribuendo alla riduzione del technical debt; cross-training da Sviluppo ad Operation per ridurre il numero di escalation.
- **Area 4:** Embed IT Operations into Development: incorporare o collegare risorse di Operation allo Sviluppo, definire requisiti non funzionali che possono essere utilizzati trasversalmente per i progetti.

Patrick Debois introduce anche tre livelli di intervento per ciascuna area: Persone (cultura), Processi, Strumenti e un meccanismo di misurazione della maturità della specifica prassi, che può essere secondo Debois, mutuato da CMMI (*Capability Maturity Model Integration*) o CIMM (*Continuous Integration Maturity Model*).

Esempio (usando il CIMM per la maturità che prevede come livello base il livello "Intro"):



I benefici di DevOps

Gene Kim declina i vantaggi di business che le organizzazioni ottengono adottando il DevOps nei seguenti tre:

- Minore time-to-market (maggiore frequenza dei rilascio);
- Aumento della qualità (maggiore disponibilità, maggiore percentuale di successo dei change, minor numero di errori, etc.);
- Maggiore efficacia organizzativa (aumento del tempo dedicato ad attività a valore aggiunto vs tempo perso, aumento di valore al cliente).

Nel documento "Top 11 Things To Know About DevOps" egli descrive inoltre tre esempi di "pattern" che illustrano l'applicazione pratica del modello DevOps in tre situazioni tipiche di conflitto tra sviluppo ed operation.

Strumenti

Sono diversi gli strumenti che supportano un approccio DevOps. Come per molte altre "best practice" si corre il rischio di cercare – invano – lo strumento che prodigiosamente cambi la vita instaurando il nuovo modello.
Oggi i maggiori player, come IBM ed HP, propongono soluzioni DevOps, altri meno noti, come Puppet e Chef,sono considerati il "best of breed" sul tema.
Le tipologie di strumenti di supporto possono essere così classificate in ordine decrescente di importanza:

- Version Control;
- Configuration Management;
- Ticketing;
- Resource monitoring;
- Provisioning.

Ma come lo stesso Patrick sottolinea in un'intervista a Manuel Pais (Infoq[3]) "*gli strumenti, come sempre accade, sono la cosa più visibile; tuttavia, per quanto possano essere straordinari, non possono cambiare la vita: sono importanti, in quanto in qualche modo possono forzare chi li usa ad adottare diversi modalità (bottom-up), ma non fanno la reale differenza se non associati ad un vero cambiamento culturale (top-down). DevOps non è partito come un insieme di strumenti e tecnologie, sebbene sia molto più facile adottare nuovi strumenti cambiando i vecchi piuttosto che cambiare la cultura in azienda*".

Diffusione

Un'indagine del dicembre 2012 di Puppet Labs e IT Revolution Press (2013 State of DevOps Report), che ha coinvolto oltre 4.000 professionisti di esercizio e sviluppo IT appartenenti ad organizzazioni di diversa natura e dimensione in oltre 90 Paesi (Americhe, Asia, Europa, Sudafrica, Australia), riporta un'accelerazione del 26% nell'adozione delle prassi DevOps rispetto all'anno precedente (2011). Secondo quanto riportato nel report, i benefici portati dall'adozione di pratiche DevOps sono stati:

- Migliore qualità dei rilasci software (63%);
- Maggiore frequenza dei rilasci (63%);
- Maggiore visibilità nei processi IT e requisiti (61%);
- Cambiamento culturale verso una maggiore collaborazione (55%);
- Maggiore capacità di reazione ai bisogni del business (55%);
- Incremento di sviluppi "Agile" (51%);
- Migliore qualità del codice (38%).

La ricerca introduce quattro indicatori:

- **Key Agility Performance Indicator:**
  - » Deployment frequency;
  - » Change lead time;
- **Key Reliability Performance indicator:**
  - » Mean time to recover;
  - » Change fail rate.

La survey ha rilevato miglioramenti delle prestazioni tanto maggiori quanto più matura era l'adozione di pratiche DevOps.

Criticità

L'adozione di processi DevOps e strumenti di automazione contribuisce al miglioramento delle prestazioni ma non è certo sufficiente al successo dell'azienda nel suo complesso. Le principali difficoltà ed ostacoli sono di tipo culturale e di tipo comunicativo. L'indagine riscontra i seguenti principali tipi di ostacoli all'adozione del modello:

- Valore di DevOps non compreso al di fuori del gruppo (48%);
- Management non comune alle strutture Dev e Ops (43%);
- Mancanza di strumenti di automazione (33%);
- Mancanza di tempo per implementare DevOps (31%);
- Mancanza di supporto (19%);
- Costi (5%).

Tra coloro che non hanno piani di implementazione di DevOps:

- Mancanza di approvazione e condivisione del management (Management Buy-in) (49%);
- Mancanza di approvazione e condivisione all'interno del team (38%);
- Vincoli di budget (19%);
- Percezione che si tratti più di un movimento temporaneo che di sostanza (14%).

Conclusioni

Il tema merita attenzione ed offre diversi spunti di riflessione: sotto certi aspetti può apparire non sufficientemente maturo per essere considerato una "*best practice*", per altri versi sembra essere l'uovo di Colombo: il conflitto tra sviluppo ed esercizio è ben noto come sono rinomate, anche sotto il profilo del buon senso, le doti taumaturgiche di una buona "comunicazione" che porti a condividere e a

3 http://www.infoq.com/interviews/debois-devops

collaborare.
Meno comune ed immediato, tuttavia, è il richiamo a metodi olistici e scientifici, come la Teoria dei Vincoli, per attuare quel cambiamento che il buon senso da solo non può produrre - altrimenti il conflitto non sarebbe così diffuso e non rappresenterebbe un problema per i tanti anche dotati di buon senso.

Riferimenti

1. *John Willis: The Convergence of DevOps (http://itrevolution.com/the-convergence-of-devops/)*
2. *Patrick debois Blog (http://www.jedi.be/blog/)*
3. *2013 State of DevOps Report (Puppet labs & IT Revolution Press) (http://info.puppetlabs.com/2013-state-of-devops-report.html)*
4. *John Allspow & Paul Hammond: Presentation "10 Deploys per day" (Velocity 2009) (http://velocityconf.com/velocity2009/public/schedule/detail/7641)*
5. *Gene Kim per IT Revolution Press: White Paper "Top 11 Things To Know About DevOps" (http://itrevolution.com/11devops/)*
6. *Intervista di Manuel Pais (Infoq) a Patrick Debois (http://www.infoq.com/interviews/debois-devops)*