Condivido con tutti le mie riflessioni in modo da raccogliere poi le considerazioni di coloro che avranno voglia di leggersi la lunga email che sto per scrivere :)

Guardando la documentazione si vede come il modello che stiamo utilizzando è vicino al Trunk Base Development.

Il Trunk Base Development prevede un tronco principale, che per noi è il master, e dei branch di release, che per noi sono i maintenance.

Il modello Trunk Base Development  prevede però 1) interventi limitati sui rami di Maintenance e 2) la possibilità di rilasciare dal tronco in un paio di giorni.

Nell'ultimo periodo abbiamo adottato il seguente approccio:

1. Ramo master: contiene tutti gli sviluppi
2. Ramo di maintenance: è la versione di produzione
3. Ramo di prerilascio: staccato dal maintenance riceve i fix (ma nella realtà poi sono sviluppi), viene usato per i test di prerilascio e poi fatto il merge sul maintenance

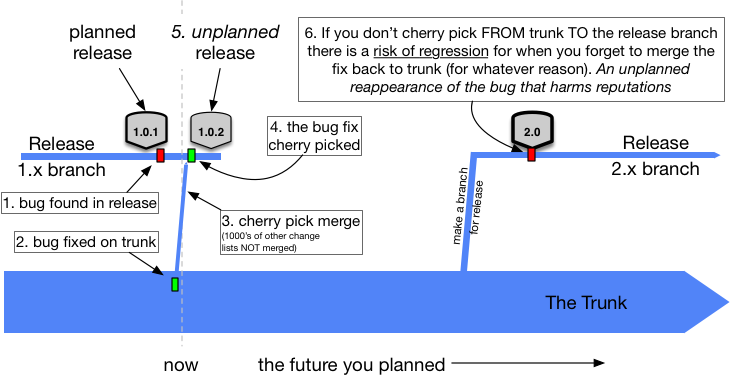
Se fare il ramo di prerilascio è una pratica che si è rivelata buona in quanto ci permette di fare test senza sporcare il ramo di produzione abbiamo un altro problema che ci crea periodicamente dei grattacapi.

Il problema che affrontiamo è il riportare successivamente gli sviluppi fatti sui maintenance sul master, lì dobbiamo gestire conflitti e la possibilità di errore in fase di merge è alta.

Come affronta il problema il modello Trunk Base?

E' molto chiaro dall'immagine sottostante. Il modello prevede che il fix venga fatto sul master (trunk) e poi riportato sul maintenance (release) tramite cherry pick (il cherry pick consente di portare un singolo commit da un ramo ad un altro). In pratica il contrario di come stiamo facendo.

Questo approccio risolve completamente il problema dei conflitti, elimina la necessità di fare dei merge dei maintenance (release) sul master (trunk) e annulla l'eventualità che un fix non venga riportato sul master.



Tutto bello, ma è applicabile a noi?

Il problema che io vedo è che spesso mettiamo le mani su versioni di diverso tempo fa. Il codice che dobbiamo fixare sul maintenance potrebbe essere molto diverso o addirittura non esistere più sul master.

In fondo è lo stesso problema che affrontiamo ora quando facciamo un fix sul maintenance e lo dobbiamo riportare sul master tramite merge.

La conclusione a cui sto arrivando è che il merge del maintenance sul master non andrebbe fatto mai in nessun caso e che i fix andrebbero portati da master a maintenance o viceversa in maniera chirurgica tramite cherry pick o a mano.

Questo significa anche che sviluppi più corposi andrebbero fatti esclusivamente sul master e che quando un cliente chiede uno sviluppo e non un fix deve essere portato forzatamente all'ultima versione.

Cosa ne pensate?