Introduzione

Negli anni Novanta abbiamo lavorato in aziende i cui progetti avevano problemi. Ci siamo trovati a dire a tutti le stesse cose: forse dovreste fare dei test prima di mettere in circolazione quel che avete fatto; perché il codice viene costruito solo sulla macchina di Mary? Perché non avete chiesto agli utenti?

Per risparmiare tempo con i nuovi clienti, abbiamo cominciato a prendere appunti e quegli appunti sono diventati *Pragmatic Programmer*. Con nostra sorpresa, il libro sembra aver colto nel segno e ha continuato a essere molto diffuso per tutti questi vent'anni.

Vent'anni però sono molte generazioni, in termini di software. Prendete uno sviluppatore del 1999 e buttatelo in un team di oggi: farà fatica ad ambientarsi in questo strano mondo. Ma anche il mondo degli anni Novanta è altrettanto estraneo allo sviluppatore di oggi. I riferimenti del libro a cose come CORBA, gli strumenti CASE e i cicli indicizzati erano diventati nel migliore dei casi pittoreschi, ma molto più probabilmente fonte di confusione.

Al contempo, vent'anni non hanno avuto quasi conseguenze per il buon senso. La tecnologia sarà anche cambiata, ma le persone no. Pratiche e impostazioni che erano una buona idea allora restano una buona idea oggi. Questi aspetti del libro sono invecchiati bene.

Così, quando è venuto il momento di creare questa "edizione del ventesimo anniversario" abbiamo dovuto prendere una decisione. Avremmo potuto ripassare tutto, aggiornare le tecnologie a cui facevamo riferimento e finirla lì. Oppure avremmo potuto riesaminare i presupposti alla base delle pratiche che avevamo consigliato, alla luce di altri due decenni di esperienza.

Alla fine, abbiamo fatto entrambe le cose.

Di conseguenza, questo libro è una sorta di "nave di Teseo" (se nel corso degli anni tutte le parti di una nave vengono sostituite, alla fine sarà ancora la stessa nave?). Circa un terzo degli argomenti del libro sono totalmente nuovi; la maggior parte del resto è stata riscritta, in parte o totalmente. Il nostro intento era rendere le cose più chiare, più pertinenti e speriamo anche un po' inossidabili al passare del tempo.

Abbiamo dovuto prendere alcune decisioni difficili. Abbiamo lasciato perdere l'appendice sulle risorse, sia perché sarebbe stato impossibile aggiornarla, sia perché oggi è più facile cercare quello che si vuole. Abbiamo riorganizzato e riscritto i temi legati alla concorrenza, data l'abbondanza attuale di hardware parallelo e il gran numero di buoni modi di trattarlo. Abbiamo aggiunto contenuti che rispecchiano il mutare di atteggiamenti e ambienti, dal movimento agile che abbiamo contribuito a lanciare, fino alla crescente

accettazione degli idiomi della programmazione funzionale e alla sempre maggiore esigenza di tenere conto di privacy e sicurezza.
C'è tuttavia una cosa interessante: abbiamo dovuto discutere fra noi molto di meno, per i contenuti di questa edizione, di quanto abbiamo fatto nello scrivere la prima. Entrambi avevamo l'impressione che fosse più facile identificare le cose davvero importanti.
Questo, comunque, è il risultato. Godetevelo. Magari adottate qualche nuova pratica. Magari deciderete che qualcuno dei nostri suggerimenti è sbagliato. Lasciatevi coinvolgere nella vostra attività. Mandateci feedback.
La cosa più importante, però, è che ricordiate di divertirvi.

Come è organizzato il libro

Questo libro è scritto sotto forma di una serie di brevi sezioni, ciascuna delle quali è indipendente e affronta un argomento particolare. Troverete molti rimandi incrociati, che vi aiuteranno a collocare ogni argomento nel suo contesto. Potete leggere le sezioni in qualsiasi ordine: questo non è un libro da leggere di fila, dalla prima pagina all'ultima.
Ogni tanto incontrerete qualche riga evidenziata dal titolo *Suggerimento* (come "Abbiate cura del vostro mestiere"). Oltre a sottolineare quanto viene detto nel testo, ci sembra che questi suggerimenti abbiano una vita propria: viviamo seguendoli ogni giorno.
Dove opportuno abbiamo inserito esercizi e progetti "sfida". Gli esercizi in genere hanno risposte relativamente immediate, mentre le sfide sono più aperte. Per darvi un'idea del nostro modo di pensare, al termine del libro trovate le nostre *Risposte agli esercizi*, ma pochissimi hanno un'unica soluzione giusta. Le sfide possono costituire la base di discussioni di gruppo o di saggi in corsi di programmazione avanzata.
La *Bibliografia* contiene invece un breve elenco di libri e articoli che citiamo esplicitamente nel testo.

Che cosa c'è in un nome?

> "Quando io uso una parola", Humpty Dumpty disse in tono piuttosto sdegnato,
> "essa significa esattamente quello che voglio – né più né meno".
> – Lewis Carroll, *Alice attraverso lo specchio*

Nel corso del libro troverete qua e là un po' di gergo: parole o espressioni italiane o inglesi che sono state piegate a indicare qualche concetto tecnico, o orrende parole inventate a cui informatici con qualche conto in sospeso con la lingua hanno assegnato un significato. La prima volta che usiamo questi termini, cerchiamo di definirli, o almeno di dare un'indicazione del loro significato. Siamo sicuri che qualcuno ci sarà sfuggito, mentre altri, come oggetto e database relazionale, sono di uso così comune che l'aggiunta di una definizione sarebbe stata noiosa. Se vi capitasse di incontrare un termine che non avete mai visto prima, non saltatelo: prendetevi il tempo per cercarlo, magari nel Web o in un manuale di informatica; e, nel caso, mandateci una email e lamentatevi, così potremo aggiungere una definizione nella prossima edizione.
Detto questo, abbiamo deciso di vendicarci degli informatici. A volte, esistono termini di gergo tecnico, che abbiamo deciso di ignorare. Perché? Perché il gergo esistente di solito

è limitato a un particolare ambito di problemi, o a una particolare fase dello sviluppo, mentre una delle filosofie fondamentali di questo libro è che la maggior parte delle tecniche che consigliamo sono universali: la modularità vale per il codice, i progetti, la documentazione e l'organizzazione del team, per esempio. Usare il termine convenzionale in un contesto più ampio sarebbe fonte di confusione: non è facile trascurare il bagaglio di significati che il termine originale porta con sé. Quando ci siamo trovati in situazioni simili, abbiamo dato il nostro contributo al declino della lingua inventando i nostri nuovi termini.

Codice sorgente e altre risorse

La maggior parte del codice presentato in questo libro è estratto da file sorgente compilabili, che potete scaricare dal nostro sito web:

`https://pragprog.com/titles/tpp20`

Lì troverete anche collegamenti a risorse che ci sono sembrate utili, con aggiornamenti al libro e notizie di altri sviluppi da "pragmatic programmer".

Feedback

Ci farebbe piacere sentire la vostra opinione. Scriveteci all'indirizzo di posta elettronica `ppbook@pragprog.com`.

Ringraziamenti per la seconda edizione

Abbiamo avuto il piacere di sostenere letteralmente migliaia di conversazioni interessanti sulla programmazione nel corso degli ultimi vent'anni, incontrando persone ai convegni, ai corsi, qualche volta addirittura in aeroplano. Ognuna di queste conversazioni ha migliorato la nostra comprensione del processo di sviluppo e ha contribuito agli aggiornamenti presenti in questa edizione. Grazie a tutti (e continuate a dirci quando sbagliamo).
Grazie a quanti hanno partecipato al processo di beta del libro. Le vostre domande e i vostri commenti ci hanno aiutati a spiegare meglio le cose.
Prima di andare in beta, abbiamo condiviso il libro con alcune persone per avere i loro commenti. Grazie a VM (Vicky) Brasseur, Jeff Langr e Kim Shrier per i loro commenti particolareggiati e a José Valim e Nick Cuthbert per le loro revisioni tecniche.
Grazie a Ron Jeffries per averci consentito di usare l'esempio del Sudoku.
Grazie molte a tutti alla Pearson per averci consentito di creare questo libro a modo nostro. Un ringraziamento speciale all'indispensabile Janet Furlow, che padroneggia tutto quello a cui mette mano e ci tiene in riga.
Infine, un saluto a tutti i programmatori pragmatici in giro per il mondo che hanno reso migliore la programmazione per tutti, nel corso degli ultimi vent'anni. Ai prossimi venti.