نفر–ماه افسانه‌ای

جُستارهایی در مهندسی نرم‌افزار

نویسنده: فردریک بروکس
مترجمان: علی پارسا، علیرضا خلیلیان

سرشناسه : بروکس، فردریک فیلیپس، ۱۹۳۱ – م.
Brooks, Frederick P. (Frederick Phillips)
عنوان و نام پدیدآور : نفر– ماه افسانه‌ای: جستارهایی در مهندسی نرم‌افزار/ نویسنده فردریک بروکس؛ مترجمان علی پارسا، علیرضا خلیلیان.
مشخصات نشر : تهران: اقتصاد فردا، ۱۴۰۴ (به سفارش انجمن انفورماتیک ایران).
مشخصات ظاهری : ۲۲۴ ص.
شابک : ۹۷۸-622-94973-5-7
وضعیت فهرست نویسی : فیپا
یادداشت : عنوان اصلی: The mythical man-month : essays on software engineering, 1995.
عنوان دیگر : جستارهایی در مهندسی نرم‌افزار.
موضوع : نرم‌افزار –– مهندسی
Software engineering
شناسه افزوده : پارسا، علی، ۱۳۲۹ –، مترجم
شناسه افزوده : خلیلیان، علیرضا، ۱۳۶۰ –، مترجم
رده بندی کنگره : QA۷۶/۷۵۸
رده بندی دیویی : ۰۰۵/۱
شماره کتابشناسی ملی : ۱۰۲۴۷۹۷۱
اطلاعات رکورد کتابشناسی : فیپا

این کتاب ترجمه‌ای است از:

The mythical man-month : essays on software engineering
نفر– ماه افسانه‌ای: جستارهایی در مهندسی نرم‌افزار
بروکس، فردریک فیلیپس
مترجمان علی پارسا، علیرضا خلیلیان
چاپ اول: آبان ۱۴۰۴
شمارگان: ۲۰۰
قیمت: ۳۰۰,۰۰۰ تومان

برای خرید کتاب با دفتر انجمن انفورماتیک ایران (۶۶۴۱۲۹۷۶-۰۲۱) تماس حاصل فرمایید.

فهرست مطالب

پیشگفتار ترجمهٔ فارسی

ای صبا نَکهَتی از خاک ره یار بیار ببر اندوه دل و مژدهٔ دلدار بیار
نکته‌ای روح‌فزا از دهن دوست بگو نامه‌ای خوش خبر از عالم اسرار بیار
تا معطر کنم از لطف نسیم تو مشام شمه‌ای از نفحات نفس یار بیار

«حافظ»

کتاب حاضر روایتی واقعی، کوتاه و گیرا است که بر اساس تجربه‌های مدیریت پروژه‌های رایانشی و نرم‌افزاری، به‌زبانی فنی ولی درک‌پذیر برای همگان بیان شده است. جوهرهٔ کتاب حاضر این است که محاسبهٔ هزینه و زمان پروژه‌های تولید نرم‌افزار بر اساس معیار نفر-ماه، افسانهٔ ذهنی یعنی توهم یا پندارِ مدیریتی و درنظر گرفتن این دو به‌جای هم خطای محاسباتی بیش نیست.

نویسندهٔ این کتاب فردریک بروکس دانشمند پُرآوازهٔ رایانه و مدیر پروژه در شرکت آی‌بی‌اِم است. در این پیشگفتار به شرح زندگی، تحصیل و فعالیت‌های کاری و پژوهشی بروکس نمی‌پردازیم و از خواننده محترم دعوت می‌کنیم برای کسب اطلاعات بیشتر به سه منبع زیر مراجعه نماید.

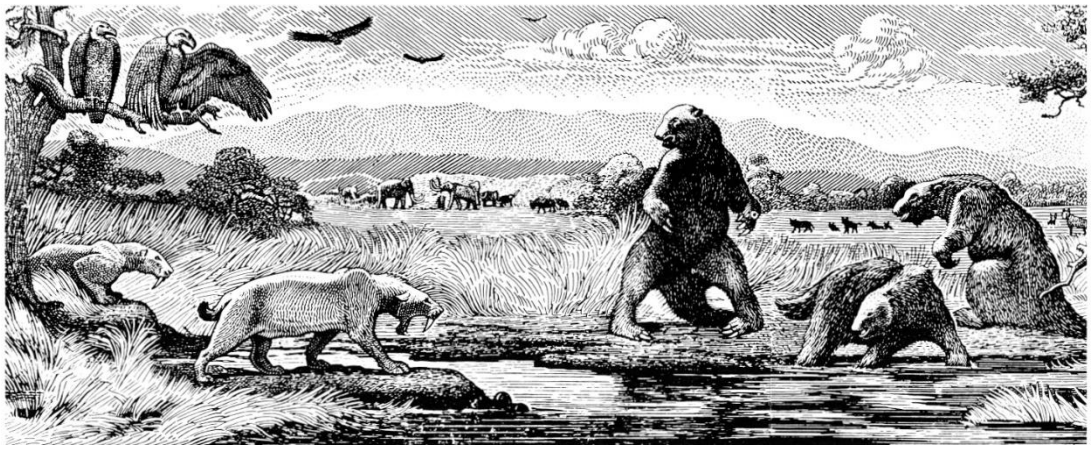
- مقالهٔ «فردریک بروکز» در صفحه ۱۷ شماره ۲۶۴ مجله گزارش کامپیوتر و مقالهٔ «طراحیِ طراحی، جُستارهایی از یک دانشمند رایانه» در صفحه ۵۱ شماره ۲۶۸ مجله گزارش کامپیوتر در وبگاه انجمن انفورماتیک ایران[1]؛
- فصل «فردریک بروکز، خالق نفر-ماه افسانه‌ای» در کتاب «بزرگان دانش رایانه»، نوشته شاشا و لازِر با ترجمهٔ استاد فرهیخته آقای ابراهیم نقیب‌زادهٔ مشایخ، انتشارات گل‌واژه، سال ۱۳۸۴.

بسیاری از مطالب و آموزه‌هایی که بروکس به‌صورت تجربه‌های کلی خود در این کتاب مطرح می‌کند در سال‌های بعد با مطالعه‌های تجربی علمی و تحت موضوع‌های خاص و مشخص بررسی و در نشریه‌های معتبر نرم‌افزاری منتشر شده‌اند و این‌کار در حال‌حاضر به‌طور گسترده‌تر ادامه دارد. در این رابطه خواننده علاقه‌مند می‌تواند به نشریه‌های زیر مراجعه نماید:

Information and Software Technology[2], Journal of Systems and Software[3], Software: Practice and Experience[4], Journal of Software: Evolution and Process[5], Software Quality Journal[6], Empirical Software Engineering[7], ACM Transactions on Software Engineering and Methodology[8], IEEE Transactions on Software Engineering[9]

---

[1] https://isi.org.ir
[2] https://www.sciencedirect.com/journal/information-and-software-technology
[3] https://www.sciencedirect.com/journal/journal-of-systems-and-software
[4] https://onlinelibrary.wiley.com/journal/1097024x
[5] https://onlinelibrary.wiley.com/journal/20477481
[6] https://link.springer.com/journal/11219
[7] https://link.springer.com/journal/10664
[8] https://dl.acm.org/journal/tosem
[9] https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=32



فصل اول باتلاق

کشتی به‌گل نشسته در ساحل همچون فانوس دریایی عمل می‌کند.
«زبان‌زد (ضرب‌المثل) هلندی»

هیچ صحنه‌ای از ماقبل تاریخ زنده‌تر از تقلای مرگبار حیوانات عظیمی که در باتلاق‌ها گیر افتاده‌اند نیست. انسان با چشم دل می‌تواند دایناسورها و ماموت‌ها و ببرهای دندان‌خنجری را درنظر بیاورد که برای رهایی از باتلاق دست‌وپا می‌زنند. هرچه تقلای آنها بیشتر می‌شود، باتلاق بیشتر آنها را در بر می‌گیرد و هیچ حیوانی آن‌قدر قوی یا ماهر نیست که عاقبت در باتلاق غرق نشود.

در دهۀ اخیر برنامه‌سازیِ سامانه‌های بزرگ نیز چنین باتلاقی بوده و موجودات بسیار بزرگ و نیرومندی در آن افتاده‌اند. اغلب آنان با ارائه سامانه‌هایی که کار می‌کرده است توانسته‌اند از باتلاق دربیایند ولی عدۀ کمی از آنها به اهداف و برنامه زمانی و بودجه تخصیص داده شده وفادار مانده‌اند. گروه‌های بزرگ یا کوچک، قدرتمند و ضعیف، پی‌درپی در دام این باتلاق گرفتار آمده‌اند. به‌نظر نمی‌رسد هیچ چیز خاصی مشکلی ایجاد کند چون هر دست یا پایی را می‌توان از باتلاق بیرون کشید. ولی مجموعۀ عواملی که همزمان عمل می‌کنند و برهم اثر می‌گذارند حرکت را مشکل و مشکل‌تر می‌کند. ظاهراً همه از میزان چسبندگی مسئله به شگفت آمده‌اند و کشف و تشخیص ماهیت آن نیز مشکل است. ولی اگر قرار است که این مسئله را حل کنیم باید تلاش در فهم آن داشته باشیم.

پس بیایید از شناسایی فن برنامه‌سازی سامانه‌ها و آشنایی با شادی‌ها و غم‌های آن آغاز کنیم.[1]

محصول سامانه‌های برنامه‌سازی[1]

اغلب در روزنامه می‌خوانیم که دو برنامه‌نویس در گاراژ منزلشان برنامۀ مهمی ساخته‌اند که از کار تیم‌های بزرگ برنامه‌سازی نیز بهتر است. هر برنامه‌نویسی هم آمادۀ قبول این نوع قصه‌ها است چرا که مطمئن است خودش هم می‌تواند برنامه‌ای را بسیار سریع‌تر از گروه‌های حرفه‌ای برنامه‌سازی که براساس گزارش‌ها سالی ۱۰۰۰ خط تولید می‌کنند بسازد.

پس چرا گروه‌های برنامه‌سازی صنعتی را با تیم‌های دو نفری گاراژها جایگزین نمی‌کنند؟ برای پاسخ به این پرسش باید ببینیم که **چه چیزی** تولید می‌شود.

در گوشه بالا و سمت چپ شکل ۱.۱ یک **برنامه** داریم. این برنامه کامل است و آمادۀ اجرا توسط نویسنده و

فصل دوم نفر–ماه افسانه‌ای

آشپزی خوب زمان می‌برد. اگر مجبور شده‌اید منتظر بمانید به‌خاطر این است که بهتر از شما پذیرایی کنند و رضایت شما را جلب نمایند.

«صورت‌غذای رستوران آنتونی، نیواورلئان»

کمبود وقت، بیش از مجموعهٔ سایر عوامل، باعث لطمه خوردن یا توقف پروژه‌های نرم‌افزاری شده است. ببینیم چرا این عامل مصیبت‌بار تا این حد رایج و فراوان است؟

اول بدین دلیل که روش‌های تخمین زدن زمان به حد کافی مناسب نیست. از آن بدتر آنکه در اغلب این روش‌ها فرض اعلام نشده و نادرستی وجود دارد که همان فرضِ پیش رفتن تمام کارها بدون گیر و اشکال است.

دوم اینکه روش‌های تخمینِ زمان ما به‌غلط اقدام و عمل را با پیشرفت اشتباه می‌گیرند و بر این پیش‌انگاشت نادرست که تعداد افراد و ماه‌های پروژه قابل تبدیل و تعویض با یکدیگرند، سرپوش می‌گذارند.

سومین دلیل این است که چون معمولاً از تخمین‌های خود مطمئن نیستیم، مدیران نرم‌افزاری ما فاقد یک‌دندگی مؤدبانه آن سرآشپز غذاکده هستند که بر گزینگان غذاکده نوشته بود:

«تهیهٔ غذای خوب طول می‌کشد. اگر شما را منتظر می‌گذاریم برای ارائه خدمت بهتر و جلب رضایت شما است.»

چهارم آنکه پیشرفت برنامه زمانی به‌دقت کافی مورد بررسی قرار نمی‌گیرد. روش‌هایی که در سایر رشته‌های مهندسی فایدهٔ خود را نشان داده و به‌طور روزمره به‌کار می‌رود، در مهندسی نرم‌افزار روش‌هایی جدید و انقلابی شمرده می‌شود.

و پنجم اینکه وقتی از برنامهٔ زمانی عقب می‌افتیم راه‌حل طبیعی و سنتی آن است که به نیروی انسانی پروژه بیافزاییم. این کار مثل تلاش برای خاموش کردن آتش با بنزین است و اوضاع را بسیار بدتر می‌کند. هرچه آتش بیشتر شود، بنزین بیشتری باید ریخت و این‌کار ما را در چنبره‌ای گرفتار می‌کند که به فاجعه منجر می‌شود.

ما به امر کنترل پیشرفت زمانی در مقالهٔ دیگری می‌پردازیم ولی فعلاً بیاییم و سایر جنبه‌های مسئله را دقیق‌تر بررسی کنیم.

خوش‌بینی

برنامه‌سازان آدم‌های خوش‌بینی هستند. شاید این جادوی عصر جدید کشش ویژه‌ای برای کسانی دارد که به پایان خوش قصه‌ها و دختر شاه پریان باور دارند. شاید آنان که از روی عادت توجه‌شان به هدف نهایی است از برخورد با صدها مورد کار خسته‌کننده و ناامیدکننده نمی‌هراسند. شاید هم علت آن است که رایانش رشتهٔ جوانی است و برنامه‌سازان از آن هم جوان‌ترند و جوان‌ها همیشه خوش‌بین هستند. ولی فارغ از اینکه چطور این جریان انتخاب طبیعی سیر می‌کند، نتیجه یکی است و مرتباً می‌شنویم که: «این دفعه دیگر کار خواهد کرد،» یا «بالاخره آخرین اشکال را پیدا کردم.»

بنابراین اولین فرض غلط که در پس برنامهٔ زمانی برنامه‌سازی سامانه‌ها وجود دارد این است که **همه‌چیز به‌خوبی پیش خواهد رفت**، یا به‌عبارت دیگر **هر کاری فقط آن اندازه که «باید» طول بکشد طول خواهد**

فصل سوم گروه جراحی

بررسی‌ها حاکی از این بود که افرادِ کاری و پیشرو عموماً یک مرتبه بزرگی با افراد ناکارآمد دگرسانی و تفاوت داشتند.[۱]

«سَکمِن، اِریکسون و گرَنت[1]، ۱۹۶۸»

در جلسه‌های انجمن‌های رایانشی آدم مرتباً از زبان مدیران جوان برنامه‌سازی می‌شنود که آنان یک گروه کوچک، پرانرژی و زِبروزرنگ از افراد درجهٔ اول را به پروژه‌های بزرگی که شامل صد برنامه‌ساز متوسط می‌شود، ترجیح می‌دهند. البته همه ما هم چنین می‌خواهیم.

ولی این طرز بیان ساده‌انگارانهٔ گزینه‌ها از برخورد به مسئله مشکل و اصلی طفره می‌رود. مسئله اصلی این است: چگونه می‌شود یک سامانه **بزرگ** را براساس یک برنامه‌ریزی زمانی مناسب بنا کرد؟ بیایید به جوانب این سؤال با دقت بیشتری بنگریم.

صورت مسئله

دیرزمانی است که مدیران برنامه‌سازی متوجه تفاوت‌های زیاد بین برنامه‌سازان خوب و برنامه‌سازان ضعیف شده‌اند ولی با وجود این امر میزان این تفاوت همه را متعجب می‌سازد. سَکمِن، اریسکون و گرنت در یکی از مطالعه‌هایی که انجام دادند میزان کارایی یک گروه از برنامه‌سازان را سنجیدند. حتی در همین گروه کوچک نیز تفاوت متوسطِ کارایی بین بهترین و بدترین حالت ۱۰ به ۱[(۱)] بود و در مورد تفاوت سرعت و میزان حافظه موردنیاز برنامه‌ها، این نسبت به میزان بسیار جالب ۵ به ۱ می‌رسید! خلاصه اینکه یک برنامه‌ساز که بیست هزار دلار در سال می‌گیرد ممکن است ۱۰ برابر بهتر از برنامه‌ساز دیگری باشد که ده هزار دلار در سال می‌گیرد. وارون این مطلب نیز ممکن است درست باشد. این برررسی وجود رابطه‌ای را بین سابقه کار و کارآیی نشان نمی‌داد (و من شک دارم که چنین رابطه‌ای به‌طور کلی وجود داشته باشد).

پیشتر گفتم که تعداد افرادی‌که باید باهم هماهنگ شوند بر هزینه عملیات تأثیر می‌گذارد چرا که بخش مهمی از این هزینه شامل ارتباط‌گیری[(۲)] و رفع ایرادهای مربوطه به ارتباط‌گیری نادرست (یا اشکال‌زدایی سامانه) می‌شود. و این حقیقت ما را ترغیب می‌کند که سامانه را حداقل تعداد افراد شرکت‌کننده بسازیم. البته اغلب تجربه‌هایی که در زمینه سامانه‌های برنامه‌سازی بزرگ وجود دارد نشان می‌دهد که روش استفاده از تمام افرادی‌که به‌نظر لازم می‌رسند، پرهزینه و کُند و نامفید است و منجر به سامانه‌هایی می‌شود که از لحاظ مفهومی دارای یکپارچگی نیستند. مثلاً می‌توانیم از اواِس ۳۶۰، اِگزِک ۸، اِسکوپ ۶۰۰۰، مالتیکس، تی‌اِس‌اِس، سِیج[2] و غیره نام ببریم و این فهرست را می‌توان باز هم ادامه داد.

نتیجه‌ای که از این وضع می‌توانیم بگیریم خیلی ساده است: اگر یک پروژه ۲۰۰ نفری دارای ۵ نفر مدیر باشد

---

[1] Sackman, Erikson, and Grant

[2] OS/360 ،Exec 8 ،Scope 6000 ،Multix ،TSS ،SAGE