بسمه تعالی

Why developing software is different and more expensive than hardware

در این مقاله آقای دیجسترا سعی دارد به این سوال که "چرا نرم افزار گران است؟" پاسخ بدهد. اما در بطن مقاله تلاش کرده اند به سوال "چرا طراحی نرم افزار سخت است؟" پاسخ بدهند که خود به خود گران بودن نرم افزار را نیز توجیه می کند. و برای بهتر جواب دادن به این سوال به تفاوت های و چالش هایی که طراحان نرم افزار و سخت افزار با آن مواجه اند، روی آورده است. ایشان در ابتدا تاکید کرده اند که مشکلات جاری نرم افزار فقط تا حدی می تواند توسط عدم شایستگی برنامه نویس های در گیر، توجیه شود. در بخش دیگری از مقاله به اتهام هایی که از سوی جامعه طراحان سخت افزار متوجه نرم افزاری ها می شود اشاره دارد و عدم توانایی در توضیح طبیعت کار از سوی طراحان نرم افزار را عاملی برای این می داند.

سپس آقای دیجسترا با مقایسه دنیای سخت افزار و نرم افزار سعی دارند تا سخت افزاری ها را کمی با مشکلات دنیای نرم افزار آشنا کنند. به عنوان مثال ایشان به مشکلات مربوط به تکنولوژی و تولید سخت افزار و همچنین منطق آنالوگ اشاره می کنند و با توجه به آماری بودن تضمین کیفیت توسط سخت افزاری ها و از آن طرف محدودیت ها و عدم قطعیت هایی که سخت افزاری ها با آن در کنترل کیفیت محصول تولیدی شان مواجه هستند، در قالب یک نگرانی جدی اشاره می کند.

اما برای اینکه نشان دهند که دنیای برنامه نویسی هنوز هم با مشکلاتی مواجه هست، یک سری تفاوت ها را متذکر می شود به این شکل که اگر مواد اولیه یک طرف قضیه و محصول نهایی طرف دیگر قضیه باشد، برای انجام یک طراحی خوب باید این دو مورد پایدار باشند اما در بحث سخت افزاری می بینیم که مواد اولیه همیشه در حال تغییر هستند و حتی به بوجود آمدن تکنولوژی های جدید این امکان را به آن ها نمی دهد که بتوانند به طور کامل با یک ماده اولیه آشنا شوند و از طرف دیگر در مقایسه با تغییر که در مواد اولیه آنها اتفاق می افتد تفاوت آنچنانی در محصول نهایی شان مشاهده نمی شود و از این رو آن ها فقط یک محصول را همیشه از نو طراحی می کنند، اما نکته خیلی مهمی که وجود دارد و یکی از دلایل سختی طراحی نرم افزار نسبت به سخت افزار این است که نقطه نهایی سخت افزاری ها نقطه شروع کار برنامه نویس ها است، و آن چالش تغییر در نرم افزار کاملا برعکس اتفاق می افتد و برنامه نویس با اینکه از مواد اولیه پایداری برخوردار هستند اما همیشه محصول نهایی شان در حال تغییر است و این به نسبت سخت تر و پر هزینه است!

آقای دیجسترا با توضیح ساده بودن طراحی نرم افزارهای عملی و فنی به دلیل داشتن قواعد و قوانین ثابت شده و مشخص، به سراغ چالش دیگری می روند و از این رو دلیل دیگری می آورند که باز سختی کار نرم افزاری ها را نشان می دهد. با توجه به اینکه سخت افزاری ها برای شناختن مواد اولیه کارشان، دائما رجوع می کنند به قوانین ثابت شده و در دسترس معینی در فیزیک و الکترونیک که این خیلی ساده است اما برای اینکه درک شود طراحی نرم افزار چقدر سخت تر است به نبود این قواعد و قوانین مشخص در حوزه نرم افزار اشاره می کنند و مثال های جالبی از سیستم عامل های اولیه که هیچگونه درکی از مشکلات نوینی که ممکن است برایشان پیش اید می آورد.

در بخش دیگر از مقاله، ایشان با اشاره به پیش زمینه تاریخی برنامه نویسی ساخت یافته و تجربه تدریس شان در دانشگاه MIT اشاره می کنند و تناظر بین لم در ریاضیات و زیر روال در برنامه نویسی را بیان می کنند.

همچنین به عنوان دلیل دیگری از سختی کار و حل یک سری مشکلات در ارائه منطق برنامه، ایشان ضرورت ابداع یک زبان برای تعامل را پیشنهاد می کنند و تجربه شخصی شان در نوشتن یک برنامه در محیط چند برنامگی را می آورند و به مفهومی که آنجا استفاده کرده بودند اشاره می کنند.

ایشان با آوردن خاطره از سخنرانی شان در یک شرکت کامپیوتری، سعی بر این دارند تا نشان دهند که یک مدیر با عدم تسلط بر کارش چقدر می تواند در حوزه طراحی دردسر ساز شود و در ادامه با اشاره به وضعیت مدیران که در ردیف اول مخاطب حرف های ایشان بوند و همچنین تشویق هایی سوی برنامه نویسان در ردیف های آخری به نوعی حرف خود را در مورد مدیران تصدیق می کند.

همانطور که در ابتدای این متن نیز اشاره شد گرچه بخشی از مشکلات می تواند توسط عدم شایستگی و ناکافی بودن اطلاعات برنامه نویسان توجیه شود اما بخش بزرگی از آن نیز که باز یکی از دلایل سختی طراحی نرم افزار است، بر می گردد به اینکه طراحان ناموفق بدون اینکه ابتدا تحلیل های لازم را انجام دهند و به یک طراحی درست برسند سراسیمه وارد محیط کدنویسی می شوند که یکی از تبعات آن نیز تلف شدن بیش از ۸۰٪ انرژی برنامه نویسان شان برای انجام کارهای پشتیبانی بعد از تحویل نرم افزار می باشد.

تست کردن و بررسی محصول دلیل دیگری است که سختی کار طراحان نرم افزار نسبت به طراحان سخت افزار را نشان می دهد. سخت افزاری ها با برخورداری از یک محیط پیوسته این امکان را دارند که بتوانند کلیه حالات محصول شان را تست کنند و از این رو کارایی آن را تضمین کنند، اما این اتفاق برای طراحان نرم افزار به راحتی اتفاق نمی افتد و در نهایت نرم افزاری ها مجبورند تا کیفیت و کارایی محصول شان را با استدلال های ریاضی ثابت کرده و تضمین کنند، که مسلما کار چندان راحتی نخواهد بود.