اصول HRT در عمل

تاکنون این طور به نظر می آید که ما به روی منبر رفته و در مورد تواضع، احترام و اعتماد فقط حرف زده و موعظه کردیم. باید دید که چطور می توان این صحبت ها را در زندگی روزمره به واقعیت تبدیل کرد. در این بخش، به دنبال راهکارهای عملی ای هستیم تا به وسیله آنها یک سری رفتارهای شخصی را بازبینی کنیم. ممکن است خیلی از مثال هایی که می آوریم به نظر واضح و مبرهن بیایند ولی وقتی که در آنها عمیق شوید متوجه می شوید که بیشتر اوقات شما یا همکارانتان خطاکار هستید.

غرورت را کنار بگذار

این جمله با زبان بی زبانی عدم تواضع را به روی آدمی که خیلی رفتارهای فروتنانه ای ندارد، می آورد. هیچ کس دوست ندارد با کسی همکار باشد که دائماً فکر می کند مهم ترین انسان در اتاق است. حتی اگر واقعاً شما باهوش ترین آدم در یک بحث باشید، لازم نیست مدام هوشتان رو در چشم سایرین فرو کنید. به عنوان مثال، آیا در مکالمه ها دوست دارید همیشه حرف اول یا حرف آخر را شما بزنید؟ آیا فکر می کنید لازم است که درباره هر موضوع و مبحثی نظر بدهید؟ یا شاید احتمالاً آدمی را میشناسید که این طور باشد.

توجه کنید که تواضع و فروتنی به این معنی نیست که اجازه دهید بقیه سوار شما بشوند. اعتماد به نفس هیچ اشکالی ندارد. فقط مثل آدم های همه چیز دان رفتار نکنید. همیشه سعی کنید تمرکزتان روی برانگیختن غرور تیمی باشد، نه فردی. به جای اینکه نشان دهید که شخصاً آدم فوق العاده ای هستید، سعی کنید حس موفقیت و غرور تیمی را بین همکارانتان تقویت کنید. یک مثال خوب در این راستا، بنیاد نرم افزار Apache است که کارنامه بلندبالایی در ساخت تیم های منسجم در تولید نرم افزارهای مختلف دارد. تیم های این بنیاد به صورت فوق العاده ای به همکاری و کار گروهی خود می بالند و اجازه نمی دهند که یک شخص با فردگرایی برای پز دادن از تیم سوء استفاده کند.

غرور خودش را به روش های زیاد و متفاوتی نشان می دهد، و بیشتر اوقات مانع کارایی شما میشود. به قسمتی دیگر از سخنرانی ریچارد همینگ که در این راستا توجه کنید:

> جان تاکی [1] همیشه خیلی معمولی لباس می پوشید. وقتی وارد دفتر کار مهمی میشد، ممکن بود مدت ها طول بکشد تا بقیه متوجه شوند که او چه شخص مهمی است و بهتر است به صحبت های او گوش دهند. برای مدت زمان زیادی، جان مجبور بود با این طرز رفتار بقیه کنار بیاید. غرور و تکبر کار بیهوده ای هست. من هیچ وقت نگفته ام که جلوی دیگران کوتاه بیایید. بحث من این است که اگر با بقیه کنار بیایید و خودبزرگ بینی را کنار بگذارید، حتما سودش را می بینید. ولی برعکس اگر غرور و تکبر را انتخاب کنید و معتقد باشید که همیشه حرف حرف شماست، به مرور در طول عمر کاری خود هزینه اش رو می پردازید ... یادگرفتن سیستم های اطراف به شما کمک میکند که از آنها در راستای اهداف و نیازهای خودتان استفاده کنید. اگر نه، مجبورید دائما در جنگ با سیستم های دور و برتان باشید.

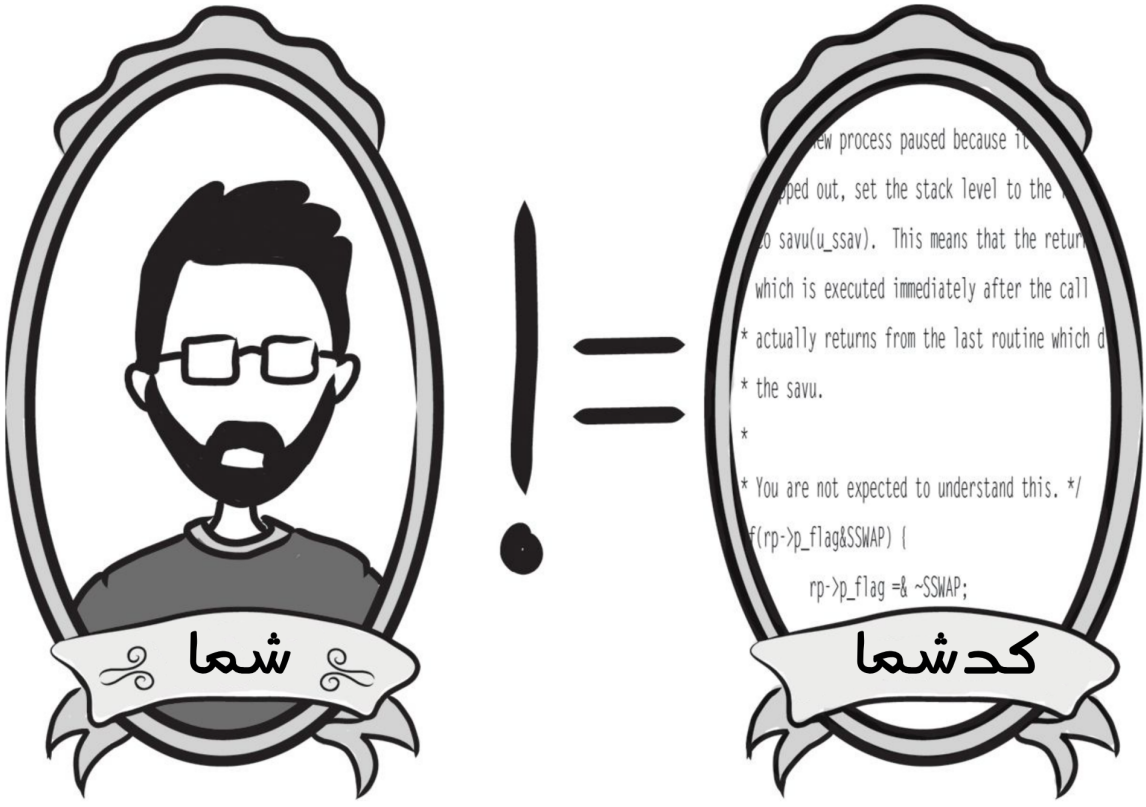
نحوه انتقاد کردن و انتقاد پذیری

داستانی در رابطه با شخصی به نام جو که به تازگی کار جدیدی به عنوان برنامه نویس آغاز کرده بود، مطرح میکنیم. او همان هفته اول شدیداً روی کد نرم افزار و پروژه تمرکز کرد. از آن جا که برای کارش اهمیت بالایی قایل بود و به نحوه برنامه نویسی دقت داشت، به مرور از هم تیمی های خودش در مورد کارشان سوال کرد. محترمانه در مورد نحوه طرز فکر و پیش فرضیاتشان اطلاعات بیشتری درخواست کرد و تکه کدهایی را برایشان ایمیل می کرد تا هم نظرشان رو بپرسد و هم مودبانه پیشنهاد بدهد که چطور می توانند منطق کدشان را بهتر کنند. چند هفته که گذشت، به دفتر رئیس بخش احضار شد. جو با تعجب پرسید: "مشکل چیه؟ من چه کار اشتباهی انجام دادم؟" رئیسش با نگرانی گفت: "ما شکایت های زیادی در مورد تو شنیدیم. جو، به نظر میاد که رفتار تو با همکارانت خیلی تند و خشن است. چپ و راست از آنها ایراد می گیری. از تو ناراحتن. لازمه که کمی ملایم تر باشی!" جو مات و مبهوت بود. در یه محیط مناسب و بر پایه اصول HRT ، همکاران جو باید با خوشحالی از نظرات و پیشنهادهای جو استقبال می کردند. ولی در

این مورد خاص، به نظر می آید که جو باید به خاطر عدم اعتماد به نفس همکارهای خودش ملایم تر رفتار می کرد و نظرات و پیشنهادهای خودش رو به صورت لطیف تری مطرح می کرد.

در یک محیط حرفه‌ای مهندس نرم‌افزار، انتقادات هیچ وقت شخصی نیستند. بلکه بیشتر اوقات در راستای بهتر شدن محصول نهایی مطرح می شوند. نکته ریز اساسی این است که شما و همکارانتان بتوانید یک انتقاد سازنده نسبت به یک کار ابتکارآمیز را از توهین به شخصیت طرف، تفکیک دهید. حمله به شخصیت آدم‌ها هیچ سودی ندارد و تقریباً غیرممکن است که آدم بتواند بابت آن کاری انجام دهد. ولی انتقاد سازنده، باعث کمک به پیشرفت میشود. مهم‌تر از همه یک انتقاد سازنده همیشه همراه با احترام مطرح می شود چرا که براساس علاقه به بهبود کار و پیشرفت طرف مقابل خواهد بود. اگه واقعاً شخصی برای شما محترم باشد، همیشه در انتخاب لغات و عبارات دقت می کنید. نزاکت در انتخاب کلمات و جملات مفید در انتقاد و پیشنهاد، مهارتیست که فقط با تمرین زیاد به دست می آید.

در مقابل، شما نیز لازم است که هنر انتقادپذیری را یاد بگیرید. یعنی نه تنها لازم است که در مهارت و توانایی‌های خود فروتن باشید، بلکه باید اعتماد داشته باشید که همکارانتان قلباً خیروصلاح و پیشرفت شما (و البته کار و پروژه) را در نظر دارند. برنامه‌نویسی هم مهارتیست که مثل هر توانایی دیگر با تمرین و ممارست بهتر میشود. اگر همکارتان در مورد فرضاً توانایی شعبده‌بازی شما پیشنهاد یا نظری میداد، آیا این نظر را یک حمله شخصی به هویت خود تلقی می کردید؟ فکر می کنیم، یعنی امیدواریم که این طور نباشد. به طور مشابه، کدی که شما می نویسید یا محصول خلاقانه ای که خلق می کنید برابر با شخصیت و ارزش هویت شما نیست. تکرار میکنیم: شما، کدی که می نویسید نیستید! شما هم این رو تکرار کنید: شما، چیزی که خلق می کنید نیستید. نه تنها لازم است که خودتان این قضیه را باور کنید، بلکه باید همکارانتان را هم قانع کنید که انسانها، کدی که می نویسند، نیستند.



به عنوان مثال، این نحوه غلط حرف زدن با یک همکار با اعتماد به نفس احتمالا پایین است:

> "بابا تو کل منطقی که روی این متد پیاده سازی کردی اشتباهه، باید از فلان مدل استاندارد که همه هم می دونن استفاده می کردی."

این نحوه‌ی انتقاد، پر است از الگوهای اشتباه: اولاً می گویید که کارش کلاً "اشتباه" است (انگار که دنیا سیاه و سفید است). طلبکارانه از او میخواهید که کاری که کرده را مطابق نظر شما تغییر دهد و او را متهم می کنید که چیزی را ساخته که در تضاد با باور و اعتقاد همه آدم هاست (به او حس حماقت القا میکنید). جوابی که دریافت خواهید کرد، یک عکس‌العمل عموماً احساسی و تدافعی خواهد بود.

روش بهتر برای انتقال همان منظور ولی با کلمات مناسب‌تر می تواند این طور باشد:

> "ببین این مطقی که توی این متد پیاده سازی شده یکم من رو گیج کرده. به نظرم شاید فلان مدل اگه پیاده سازی بشه باعث خوانایی بهتر کد و نگهداری راحت‌ترش بشه."

دقت کنید که چطور با فروتنی، مشکل را به خودتان مرتبط می کنید نه به شخص مقابل. او اشتباهی مرتکب نشده، بلکه این شما هستید که در فهم کد او دچار مشکل شدید. پیشنهاد شما صرفاً برای این است که طرف مقابل به شما کمک کند تا کد را بهتر متوجه شوید و احتمالاً پروژه نیز راحت تر به اهداف بلند مدت خود برسد. همچنین، شما از شنونده، طلب کار نیستید. به همکارتان این فرصت را می دهید که با صلح و آرامش نظر و پیشنهاد شما را قبول نکند. بدین صورت موضوع بحث حول محور کد باقی می ماند و وارد موضوع شخصیت و هنر برنامه نویسی افراد نمی شود.

یادگیری سریع از اشتباهات

یک داستان معروف و شاید کلیشه در دنیای بازار در مورد مدیری وجود دارد که با یک اشتباه بزرگ باعث می شود که شرکتی که در آن کار می کند ۱۰ میلیون دلار ضرر کند. روز بعد با ناراحتی وارد دفتر شده و شروع می کند تا وسایلش را جمع کند. همان‌طوری که انتظار داشت به او می گویند که مدیرعامل در دفترش منتظر اوست. با ناراحتی و آرامی وارد دفتر شده و یک برگ کاغذ به مدیرعامل تحویل میدهد. وقتی مدیرعامل می پرسد که این چیست؟ جواب میدهد: "استعفانامه منه. مگه نمیخواین من رو اخراج کنید؟" مدیرعامل در کمال ناباوری پاسخ میدهد که "اخراج کنم؟! من همین الان ۱۰ میلیون دلار خرج تجربه و آموزش تو کردم. حالا اخراجت کنم؟!"[۲]

مطمئناً اغراق در این داستان زیاد است ولی نکته این است که مدیرعامل این داستان به خوبی می داند که اخراج مدیری که اشتباه کرده، نه تنها ۱۰ میلیون دلار ضرر را بر نمی گرداند بلکه باعث می شود شرکت یک مدیر خوب که به خاطر تجربه ای که به دست آورده مطمئناً اشتباهش را تکرار نخواهد کرد را نیز از دست بدهد.

یکی از شعارهای مورد علاقه ما در گوگل این بود: "گزینه شکست، همیشه روی میز موجوده." باور شرکت این بود که اگر شما هر از گاهی شکست نخورید، یعنی به اندازه کافی در کارتان ابتکار نداشتید و ریسک نکردید. شکست خوردن، یک فرصت طلایی است برای یادگیری و بهتر شدن در تلاش های بعدی. یک جمله که به توماس ادیسون نسبت داده شده، این طور می گوید: "من اگر ده‌هزار راه پیدا کنم که هیچ کدام به نتیجه نرسند، شکست نخورده‌ام. دلسرد نمیشوم، چون هر تلاش اشتباهی که تموم میشود، یک قدم من را به هدف نزدیک تر میکند."

در بخش Google X، که بیشتر پروژه‌های آینده‌نگرانه مثل Google Glass و ماشین‌های خودران در آنجا انجام می شود، شکست خوردن تعمداً جزوی از سیستم تشویقی پیش بینی شده است. اعضای تیم‌ها دائماً ترغیب میشوند تا ایده‌های دیوانه‌وار همدیگه را به چالش بکشانند. هر کس به تعداد دفعاتی که در طول یک زمان مشخص، عدم موفقیت یک ایده جدید را ثابت میکنه، پاداش می گیرد و حتی با سایرین رقابت می کند. فقط وقتی که هیچ کس نمی تواند تحت هیچ شرایطی یک ایده جدید را رد کنه، ایده وارد پیاده‌سازی اولیه می شود.

راز یادگیری از اشتباهات این است که خوب و دقیق تجربیات را ثبت کنید. در دنیای کسب و کار به این نوع بررسی علت و معلول وقوع یک اشتباه، "کالبدشکافی" گفته می شود. کالبدشکافی های اشتباهات خود را مکتوب کنید. این کار را با دقت مضاعفی انجام دهید و اطمینان حاصل کنید که نوشته شما یک لیست به درد نخور از عذر و بهانه های الکی نباشد. یک کالبدشکافی مناسب، همیشه درس هایی که آدم از اشتباه یاد میگیرد را دقیق توضیح میدهد و کارهایی که لازم است در نتیجه این دروس، شروع شوند یا تغییر کنند را لیست میکند. حتماً این نوشته را جایی در دسترس قرار داده و مطمئن شوید که کارهای که نوشتید پیگیری خواهند شد. به یاد داشته باشید که نوشتن

اشتباهات کمک میکند تا سایرین نیز (چه الان و چه در آینده) از اشتباهات شما درس بگیرند و تاریخ را تکرار نکنند. ردپاهای خودتان را پاک نکنید. بر عکس آنها را شفاف و پر رنگ کنید تا راه نمایی برای بقیه باشند.

گزارش کالبدشکافی خوب، شامل موارد زیر است:

- تاریخچه اتفاقات:از زمان پیدا شدن اشتباه یا حادثه تا راه حل
- عامل اولیه اتفاق یا اشتباه
- برآورد دقیق هزینه و آسیب ایجاد شده
- مجموعه موارد تحت اقدام سریع برای غلبه فوری بر مسئله
- مجموعه موارد جهت اقدام برای جلوگیری از اشتباه مشابه در آینده
- درس هایی که از اتفاق گرفته شده

برای یادگیری، زمان بگذارید

سیندی یک ستاره بود. یک مهندس نرم‌افزار فوق العاده که حقیقتاً در کارش استاد شده بود. به راحتی به عنوان مدیرفنی پروژه ارتقا یافت و مسئولیت های بیشتری را روی دوش خودش احساس کرد. خیلی زود، مشاور و مربی اعضای تیمش شد و به آنها راه و روش مهندسی را آموزش می داد. در کنفرانس ها و سمینارهای مختلف سخنرانی می کرد و به آرامی مدیریت چند تیم مختلف دیگر را نیز به عهده گرفت. اگرچه از این که در کارش خبره شده بود لذت می برد، ولی اندک اندک در کارش بی حوصله شد. یک جایی در این مسیر، یادگیری موضوعات جدید را فراموش کرده بود. حس باهوش ترین و ماهرترین بودن در هر جمعی، خیلی آرام تازگی خود را برای او از دست داد. با وجود تمام نشانه‌های موفقیت، جای یک چیزی هنوز خالی بود. یم روز که مشغول کار بود، متوجه شد که تخصصی که دارد دیگر خیلی در دوره جدید به کار نمی آید. همه درگیر موضوعات تازه تر شده بودند و دنبال تخصص های جدید تر بودند. سیندی چه اشتباهی مرتکب شده بود؟

تعارف که نداریم، خیلی خوش میگذرد وقتی آدم داناترین فرد یک جمع باشد. مربی گری برای بقیه می تواند واقعاً ارزشمند باشد. ولی مشکل این جاست که وقتی شما به یک ماکزیموم نسبی در تیمتان دست پیدا میکنید، یادگیری را کنار می گذارید. و وقتی یادگیری را فراموش کنید، حوصله شما در کار سر خواهد رفت. عادت به رهبری تیم خیلی راحت تبدیل به یک اعتیاد می شود. فقط با تواضع و فروتنی می توان مسیر را عوض کرد و راه را برای یادگیری موضوعات جدید باز کرد. مجدداً اینجا هم تواضع و تمایل برای یادگیریست که اهمیت دارند. از حاشیه امن خودتان خارج شوید و خودتان را برای یادگیری و پیشرفت به چالش بکشانید. در طولانی مدت آدم خوشحال تری خواهید بود.

یادبگیرید صبور باشید

سال ها پیش فیتز روی پروژه‌ای برای تبدیل مخزن های کد CVS به Subversion (و بعدها Git) کار می کرد. به قدری CVS بدقلق بود که دائماً فیتز با باگ‌های جدید مواجه میشد. یکی از دوستان قدیمی فیتز، به اسم کارل قبل‌ها روی CVS خیلی کار کرده بود و به خوبی با آن آشنا بود. این شد که فیتز و کارل تصمیم گرفتند که با هم روی این پروژه کار کنند و باگ‌ها را برطرف کنند.

وقتی که دونفره شروع به برنامه‌نویسی کردند با یک مشکل مواجه شدند: فیتز از پایین به بالا به مسائل نگاه میکرد. او در هر مشکلی شیرجه میزد و با سعی و خطا و امتحان کردن روش های مختلف سعی میکرد که قضیه را حل کند. کارل اما از بالا به پایین صورت مسائل رو می شکافت. او سعی میکرد که به همه متدها و کدهای موجود نگاه کلی داشته باشد و سپس مشکل را پیدا کند. نتیجه این تفاوت در نگاه، کشمکش های مختلف و بعضا جر و بحث های شدید بین این دو نفر شد. به قدری این مشکل برای آنها خسته کننده شد که دیگر حاضر نبودند همزمان و دونفره برنامه نویسی کنند. لازم به ذکر است که آن دو دوستهای دیرینه‌ای بودند که برای همدیگر اعتماد و احترام خیلی بالایی قایل بودند. این موضوع در کنار صبر و بردباری بالای اونها باعث شد که بتوانند راه دیگری برای همکاری پیدا کنند. تصمیم گرفتند که باهم و دونفره کد نویسی کنند تا وقتی که به یک باگ برخورد کنند. آن وقت برای حل اون باگ، از هم جدا میشدند تا هر کس تنهایی روی مسئله کار کند. وقتی مشکل حل میشد، مجدداً برنامه‌نویسی دو نفره خودشون را ادامه میدادند. صبوری این دو نفر، همراه با تمایلشان برای همکاری، نه تنها سرنوشت پروژه را نجات داد، بلکه دوستی آنها را نیز پایدار کرد.

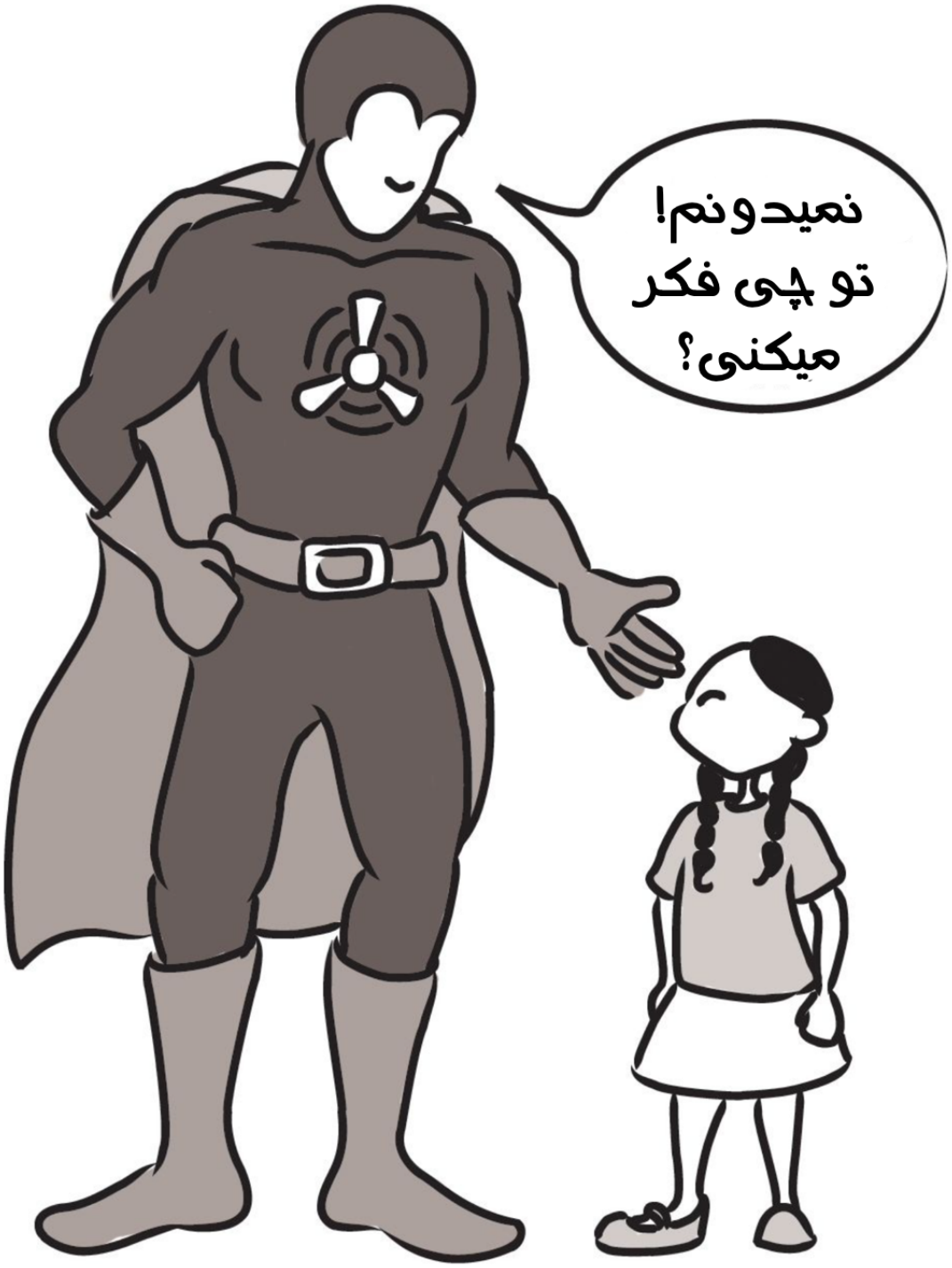
تاثیرپذیر باشید

هرچه آدم تاثیرپذیرتری باشید، شخص تاثیرگذار تری نیز خواهید بود. هرچه آسیب پذیرتر باشید، قوی تر به نظر خواهید آمد. این جمله ها در نگاه اول تناقض های عجیبی به نظر می آیند. ولی همه ما حداقل یک نفر را اطراف خودمان میشناسیم که بی نهایت آدم کله شقی است! هر چقدر هم که آدمها تلاش کنند تا قانعش کنند، همیشه مرغش یک پا دارد. معمولاً سرنوشت این طور آدمها چخ میشود؟ تجربه ما نشان داده که بیشتر اوقات مردم آنها را به عنوان موانعی در نظر میگیرند که باید دور زده شوند. بقیه به نظرات یا مخالفت هایشان خیلی محل نخواهند گذاشت. قاعدتاً نمی خواهید که شما هم به چنین آدمی تبدیل شوید. پس این موضوع را خوب در ذهن خودتان فرو کنید: هیچ اشکالی ندارد که یک نفر نظر شما را تغییر بدهد. موضوعاتی که حاضرین به خاطر آنها مبارزه کنید را با دقت انتخاب کنید. یادتان باشد که برای اینکه حرف شما به خوبی شنیده شود، ابتدا لازم است که شنونده خوبی برای حرف های بقیه باشید. قبل از این که نظر خودتان را بیان کنید یا در مورد مسئله‌ای با قطعیت تصمیم بگیرید، به نظرات بقیه با دقت گوش کنید. اگه شما دائماً تصمیم خودتان را عوض کنید، آدم بی اراده ای به نظر خواهید آمد.

اما در مورد موضوع آسیب‌پذیری، باید گفت که این مهارت در نگاه اول خیلی عجیب به نظر می رسد. اگر یک نفر به ضعف های خودش یا به عدم دانشش در یک موضوع خاص معترف باشد، مگر نه اینکه بقیه او را کمتر جدی میگیرند؟ آسیب‌پذیری یک ضعف است و باعث خراب شدن دیوارهای اعتماد میشود. مگر نه؟

خیر، اصلا درست نیست! پذیرش اشتباهاتتان یا اعتراف به اینکه کاری رو به خوبی انجام ندادید، باعث پیشرفت شما در طولانی مدت میشود. واقعیت این است که تمام اصول HRT در این قضیه قرار دارند. پذیرش اشتباه حاصل تواضع و بیانگر مسئولیت پذیری شماست. همچنین نشانه ای است بیانگر این که شما می توانید به بقیه اعتماد کنید. و در مقابل بقیه هم به شما، صداقت، و قدرتتان احترام بیشتری خواهند گذاشت.

گاهی اوقات بهترین کاری که می توانید انجام دهید فقط این است که بگویید: "نمی دونم."



همه می دانند که سیاست مدارهای حرفه ای هیچ وقت به اشتباه خودشان یا به عدم اطلاع از یک موضوع خاص اعتراف نمی کنند. حتی وقتی که کاملاً برای همه، اشتباهشان واضع و مبرهن باشد. برای همین است که هیچ کس یک کلمه از حرفایشان را باور نمی کند. این رفتار سیاست مدارها شاید به این دلیل است که دائماً از طرف احزاب مخالفشان تحت حمله هستند. ولی در دنیای مهندسی، هیچ دلیلی وجود نداره که شما دائما در حالت تدافعی قرار داشته باشید. هم تیمی شما، همکار شماست، نه رقیب شما.

---

[۱] John Tukey ، ریاضیدان معروف آمریکایی و مبدع الگوریتم Fast Fourier Transform

[۲] روایت های زیادی ازین داستان مطرح شده که به شرکت ها و مدیرعامل های متفاوتی نسبت داده شده اند.