Predgovor

Jedan od naših omiljenih slatkiša ovde u Danskoj je Ga-Jol, čiji jak miris sladića je savršen dodatak našem vlažnom i često hladnom vremenu. Deo šarma Ga-Jola za nas Dance su mudre ili duhovite izreke ispisane na poklopcu svake kutije. Jutros sam kupio paketić poslastice i otkrio da nosi ovu staru dansku izreku:

Ærlighed i små ting er ikke nogen lille ting.

„Iskrenost u malim stvarima nisu male stvari." Bio je to dobar predznak u skladu sa onim što sam ovde već hteo da kažem. Male stvari su bitne. Ovo je knjiga o skromnim brigama čija vrednost ipak nije mala.

Bog je u detaljima, rekao je arhitekta Ludwig Mies van der Rohe. Ovaj citat podseća na savremene argumente o značaju arhitekture u razvoju softvera, posebno u Agile svetu. Bob i ja povremeno se upuštamo u žestok dijalog o tome. I da, Ludwig Mies van der Rohe je ukazvao na korisnost i bezvremenost građevina koje predstavljaju najbolju arhitekturu. S druge strane, lično je odabirao brave za vrata svake kuće koju je projektovao. Zašto? Jer male stvari jesu važne.

Tokom naše „rasprave" o razvoju zasnovanom na testiranju, Bob i ja smo otkrili da se slažemo da softverska arhitektura ima važno mesto u razvoju, mada verovatno imamo različite predstave o tome šta to konkretno znači. Takve razlike su, međutim, relativno nevažne, jer možemo prihvatiti zdravo za gotovo da odgovorni profesionalci na početku projekta odvoje *neko* vreme za razmišljanje i planiranje. Pojmovi projektovanja zasnovanog na testiranju i kodu nestali su kasnih 1990-ih. Ipak, pažnja prema detaljima postala je još više važna osnova profesionalizma, više nego što je to velika vizija. Prvo, kroz praksu na detaljima profesionalci stiču stručnost i sigurnost za rad na velikim stvarima. Drugo, najmanja a pomalo neuredna konstrukcija, vrata koja se ne zatvaraju čvrsto ili blago zakrenuta pločica na podu, ili čak neuredan radni sto, u potpunosti kvare šarm celine. O tome se radi kada je u pitanju jasan i čist kod.

Ipak, arhitektura je samo jedna metafora razvoja softvera, a posebno za onaj deo softvera koji isporučuje početni *proizvod* u istom smislu kao što arhitekta isporučuje netaknutu zgradu. U današnje doba Scruma and Agile fokus je na brzom stavljanju

proizvoda za tržište. Želimo da fabrika radi maksimalnom brzinom u proizvodnji softvera. To su ljudske fabrike: razmišljanje, osećanje za kodiranje koje se radi na osnovu prethodnih proizvoda i zahtevi korisnika da bi stvorili proizvod. Metafora proizvodnje snažno odgovara takvim razmišljanjima. Proizvodni aspekti japanske auto industrije, sveta montažnih pokretnih traka, nadahnjuju Scrum.

Ipak, čak i u auto industriji, najveći deo posla nije u proizvodnji, već u održavanju – ili njegovom izbegavanju. U softveru, 80% ili više onoga što radimo je čin popravke a čudno se naziva „održavanje". Umesto da se bavimo tipičnim zapadnjačkim fokusom na *proizvodnji* dobrog softvera, mi više razmišljamo kao kućni majstor u građevinskoj industriji ili automehaničari u automobilskoj industriji. Šta bi japansko rukovodstvo reklo na *to*?

Negde oko 1951. godine na japanskoj sceni se pojavio pristup kvalitetu nazvan Održavanje potpune produktivnosti (Total Productive Maintenance, TPM). Njegov fokus je na održavanju, a ne na proizvodnji. Jedan od glavnih stubova TPM-a je skup takozvanih 5S principa. 5S je skup disciplina – i ovde poučno koristim termin „disciplina". Ovi 5S principi su u stvari osnove Leana – još jedna zvučna reč na zapadnoj sceni i sve istaknutija reč u softverskim krugovima. Ovi principi nisu opcija. Kao što ujka Bob govori u svojoj prvoj stvari, dobra softverska praksa zahteva takvu disciplinu: fokus, prisustvo uma i razmišljanje. Ne radi se uvek samo o tome kako da se postigne optimalna brzina rada fabričkih mašina. Filozofija 5S sadrži ove koncepte:

- *Seiri* ili organizacija („sortirajte"). Znati gde su stvari – korišćenje pristupa kao što je pogodno imenovanje – presudno je. Mislite da imenovanje identifikatora nije važno? Pročitajte u narednim poglavljima.
- *Seiton*, ili urednost („sistematizovati"). Postoji stara američka izreka: *Mesto za sve i sve na svom mestu*. Deo koda treba da bude tamo gde očekujete da ćete ga naći – i, ako nije, treba da preradite kod da bi pomenuti deo bio tamo.
- *Seiso* ili čišćenje (misli se na „sijati"): Na svom radnom mestu ne sme biti okačenih žica, masti, ostataka i otpada. Šta autori kažu o tome da zagađujete svoj kod komentarima i isključenim redovima kodova koje beleže istoriju ili želje za budućnost? Otarasite ih se.
- *Seiketsu*, ili standardizacija: Grupa se slaže o tome kako održati radno mesto čistim. Da li mislite da ova knjiga govori o tome da imate dosledan stil pisanja koda i skup praksi unutar grupe? Odakle dolaze ti standardi? Nastavite sa čitanjem.
- *Shutsuke*, ili disciplina (*sam*odisciplina). To znači imati disciplinu da se sledi praksa i često razmišljati o svom poslu i biti voljan menjati se.

Ako prihvatite izazov – da, izazov – čitanja i primene ove knjige, shvatićete i ceniti poslednju tačku. Ovde konačno stižemo do korena odgovornog profesionalizma u profesiji koja bi trebalo da se bavi životnim ciklusom proizvoda. Dok održavamo automobile i druge mašine pod TPM-om, održavanje kvarova – čekanje na greške na površini – izuzetak je. Umesto toga, idemo na viši nivo: svakodnevno pregledamo mašine i zamenjujemo potrošne delove pre nego što otkažu ili uradimo ekvivalent promeni ulja na 10.000 kilometara da bismo sprečili habanje. Kôd refaktorišemo nemilosrdno. Možete

poboljšati još jedan nivo, kao što je TPM pokret inoviran pre više od 50 godina: pravite mašine koje su na prvom mestu što održivije. Učiniti vaš kod čitljivim je podjednako važno kao i učiniti ga takvim da može da se izvrši. Krajnja praksa, uvedena u TPM krugove oko 1960. godine, jeste fokusiranje na uvođenje čitavih novih mašina ili zamenu starih. Kao što nas Fred Brooks opominje, verovatno bismo morali da ponovo napišemo od nule velike komade softvera svakih sedam godina ili da ga očistimo od lošeg koda. Možda bi trebalo da ažuriramo Brooksovu vremensku konstantu na redosled nedelja, dana ili sati umesto godina. Tu se kriju detalji.

Postoji velika moć u detaljima, ali postoji nešto skromno i duboko u ovom pristupu životu, što bismo stereotipno mogli očekivati od bilo kojeg pristupa koji ima japanske korene. Ali to nije samo istočnjački pogled na život; engleska i američka narodna mudrost pune su takvih opomena. Gornji citat *Seiton* izašao je iz pera jednog ministra iz Ohaja koji je urednost doslovno posmatrao „kao lek za svaki stepen zla“. A šta je sa *Seiso*? *Čistoća je pola zdravlja*. Koliko god da je lepa kuća, neuredan radni sto oduzima joj sjaj. Šta kažete na *Shutsuke* o malim stvarima? *Onaj ko je veran u malom, veran je i u velikom*. Kako stojite sa željom da preradite program na vreme, ojačavajući svoju poziciju za naredne „velike“ odluke, umesto da ih odlažete? *Stići na vreme štedite vreme*. *Ko rano rani dve sreće grabi*. *Sve što možeš uraditi danas ne ostavljaj za sutra*. (Takav je bio izvorni smisao fraze „poslednji odgovorni trenutak“ u Leanu, sve dok nije pala u ruke softverskih konsultanata.) Kako bi bilo kalibrisati mesto malih, individualnih napora u velikoj celini? *Rastu moćni hrastovi iz malih žirova*. Ili kako integrisati jednostavan preventivni rad u svakodnevni život? *Bolje sprečiti nego lečiti*. *Jedna jabuka na dan, tera doktora iz kuće van*. Jasan i čist kod poštuje duboke korene mudrosti naše šire kulture ili naše kulture onakve kakva je nekada bila i *može* biti osetljiva na detalje.

Čak i u velikoj arhitektonskoj literaturi pronalazimo izreke koje ukazuju pažnju detaljima. Pomislite na brave Miesa van der Rohea. To je *seiri*. To je pažnja za svako ime promenljive. Trebalo bi da imenujete promenljivu sa istom pažnjom sa kojom imenujete prvorođeno dete.

Kao što svaki vlasnik kuće zna, takvoj brizi i stalnom održavanju čistoće nema kraja. Arhitekta Christopher Alexander – otac obrazaca i jezika obrazaca – svaki čin projektovanja vidi kao mali, lokalni čin popravke. On smatra da je izrada fine strukture jedini zadatak arhitekte; veći oblici se mogu prepustiti obrascima i i njihovoj primeni od strane stanovnika. Projektovanje nije samo dodavanje nove sobe kući, već i pažnja posvećena krečenju, zameni dotrajalih tepiha ili dogradnji kuhinjskog elementa. Većina umetnosti odjekuje analognim osećanjima. U potrazi za drugima koji smatraju da je Bog u detaljima, našli bismo se u dobrom društvu francuskog autora iz 19. veka Gustava Flauberta. Francuski pesnik Paul Valery savetuje nas da pesma nikada nije gotova i traži neprestanu preradu, a prestanak rada na njoj je napuštanje. Takva preokupacija detaljima je zajednička svim koji teže savršenstvu. Možda je ovde malo novog, ali čitajući ovu knjigu naći ćete se pred izazovom da prihatite dobre discipline koje ste odavno predali apatiji ili želji za spontanošću i prostom „reagovanju na promene“.

Nažalost, obično ne smatramo takvu brigu ključnim temeljem umeća programiranja. Rano napuštamo svoj kod, ne zato što završen, već zato što se naš sistem više vrednosti fokusira na spoljašnji izgled, nego na suštinu onoga što isporučujemo.

Ova nepažnja nas na kraju košta: *Sve jednom dođe na naplatu*. Istraživanje je pokazalo da i u teoriji i u praksi, svi napori se čine da bi se kod održavao jasnim i čistim. U mojim danima radeći u Organizaciji za istraživanje softverske proizvodnje Bell Labs (zaista *proizvodnja*!) imali smo prateće nalaze koji su sugerisali da je dosledan stil uvlačenja redova koda jedan od statistički najznačajnijih pokazatelja niske gustine grešaka. Želimo da arhitektura ili programski jezik ili neki drugi važan pojam budu uzrok kvaliteta; kao ljudi koji svoj profesionalizam duguju savladanom alatu i visokim metodama projektovanja, osećamo se uvređeni zbog značaja koju te fabričke mašine, odnosno programeri, pridaju jednostavnoj doslednoj primeni stila uvlačenja. Citiram svoju vlastitu knjigu od pre 17 godina, takav stil razlikuje izvanrednosti od obične kompetencije. Japanski pogled na svet shvata presudnu vrednost običnog radnika, i još više, sistema razvoja koji duguju jednostavnim, dnevnim poslovima tih radnika. Kvalitet je rezultat milionskih nesebičnih izraza pažnje pažnje – ne zbog neke sjajne metode koja je pala s neba. To što su ovi postupci jednostavni, ne znači da su pojednostavljeni, i teško da znači da su laki. Oni su ipak tkivo veličine i, još više, lepote, u bilo kom ljudskom poduhvatu. Njihovo ignorisanje uopšte nije humano.

Naravno, i dalje sam zagovornik šireg razmišljanja, a posebno vrednosti arhitektonskih pristupa koji su ukorenjeni u dubokom domenu znanja i upotrebljivosti softvera. Knjiga nije o tome – ili, bar, nije očigledno o tome. Ova knjiga ima suptilniju poruku čiju dubinu ne treba potceniti. Uklapa se sa izrekama ljudi koji se bave kodom, poput Petera Sommerlada, Kevlina Henney i Giovannija Aspronija. „Program je projekat“ i „Jednostavan kod“. Dok moramo da vodimo računa da je interfejs program i da njegove strukture imaju mnogo toga da kažu o našoj programskoj strukturi, ključno je da kontinuirano zauzmemo skromni stav da projektovanje živi u kodu. I dok prerada u metafori proizvodnje dovodi do troškova, prepravka dizajna dovodi do vrednosti. Naš kod treba da posmatramo kao prelepu artikulaciju plemenitih napora dizajna – dizajna kao procesa, a ne statičke krajnje tačke. U kodu su arhitektonske metrike spajanja i kohezije. Ako slušate kako Larry Constantine opisuje povezanost i koheziju, on govori u smislu koda – a ne o apstraktnim koncepatima koji bi se mogli naći u UML-u. Richard Gabriel nas u svom eseju, „Abstraction Descant“, savetuje da je apstrakcija zlo. Kôd je protiv zla, a jasan kôd je verovatno božanski.

Vraćajući se svojoj maloj kutiji Ga-Jola, mislim da je važno imati na umu da danska mudrost savetuje ne samo da treba da obratimo pažnju na male stvari, već i da budemo *iskreni* u malim stvarima. To znači da budemo iskreni prema kodu, iskreni prema kolegama o stanju našeg koda i, pre svega, iskreni prema sebi u vezi sa našim kodom. Da li smo dali sve od sebe da „kamp ostavimo čistijim nego što smo ga zatekli“? Da li smo preradili svoj kôd pre prijavljivanja? To nisu periferne brige, već brige koje stoje direktno u centru vrednosti Agile. U Scrumu je preporučena praksa da refaktorisanje bude deo koncepta „Gotovo“. Ni arhitektura ni jasan kod ne insistiraju na savršenstvu, samo na iskrenosti i pružanju najboljeg što možemo. *Grešiti je ljudski; oprostiti, božanski*.

U Scrumu sve činimo vidljivim. Iznosimo prljav veš. Iskreni smo o stanju našeg koda jer kôd nikada nije savršen. Postajemo kompletnije ličnosti, vredniji božanskog i bliži toj osećaju u detaljima.

U našoj profesiji očajnički nam je potrebna sva pomoć koju možemo dobiti. Ako čist pod prodavnice smanjuje nesreće, a dobro organizovani alati povećavaju produktivnost, onda sam za sve to. Što se tiče ove knjige, ona je najbolja pragmatična primena Lean principa na softveru koji sam ikada video u štampanoj knjizi. Nisam ništa manje očekivao od ove male grupe mislećih ljudi koji se godinama zajedno trude ne samo da postanu bolji, već i da daruju svoje znanje softverskoj industriji u delima kao što je ovo koje je sada u vašim rukama. Ostavlja svet malo boljim nego što sam ga zatekao pre nego što mi je ujka Bob poslao rukopis.

Završivši ovu knjigu zadataka sa dubokim uvidima, slobodan sam da očistim svoj sto.

James O. Coplien
Mørdrup, Denmark