As pequenas alegrias do código aberto

03/01/2023 23:33

Fala zumbizada! Aparentemente aconteceram algumas coisas desde a última vez que postei por aqui, mas vou escrever como se estivesse postando toda semana. Uma dessas coisas é que agora eu estou escrevendo ABAP desde a origem – sim, entrei pra SAP faz alguns meses! Mais sobre isso num próximo post.

Eu desenvolvo software profissionalmente desde dezembro de 2008. Desde então, tirando um projeto ou outro com SAPUI5, um pouco de Node ou alguns poucos scripts em Groovy, esse meu tempo de trabalho foi passado quase totalmente lidando com uma linguagem de quarta geração que é mantida por uma empresa cujos clientes, segundo ela própria, geram 87% do comércio global.

O caso do ABAP é algo impressionante. Olhando pra esse fact sheet, eu diria que é a linguagem de código fechado mais usada no mundo, apesar da SAP não fazer questão de divulgar números precisos de utilização. Mas entenda bem o sentido de "usada" – aqui, quero dizer que é a linguagem de código fechado mais *executada* ao redor do mundo, mesmo que não seja *escrita* por muitas pessoas. Ainda é um mundo muito fechado.

Isso gera uma situação inusitada: temos uma linguagem de código fechado com uma utilização absurdamente grande – por exemplo, sendo responsável por **US$ 46 trilhões** de comércio mundial (mesmo que não seja uma responsabilidade pelo ciclo completo) – mas que, por ser de manutenção privada, gera divergências com a sua base de usuários desenvolvedores – como, por exemplo, uma demanda da comunidade por uma API de grid editável que completou, em 2022, 14 anos sem ser atendida, ou o caso de todo mês alguma alma perdida compartilhar nos blogs da SAP Community um método diferente de upload/download de planilhas Excel, sendo que nenhum deles é documentado ou completamente suportado de maneira oficial.

É claro que existem motivações e intenções diversas pra SAP continuar com essa abordagem. Mas o código aberto é uma força irrefreável. Como toda empresa grande, a SAP aprendeu a lidar com esse tema aos poucos, às vezes dando algumas capotadas como foi o caso do Code Exchange (que teve vida curta), mas eventualmente chegando nos métodos e ferramentas mais aceitos por desenvolvedores como um todo – git, GitHub, nuvem. Ainda assim, os esforços mais relevantes nesse sentido em torno do ABAP ainda vem da comunidade. O abapGit é pra mim uma das principais ferramentas do ABAP hoje em dia, e obviamente já não é mais novidade – a primeira versão do abapGit é de 2014, e o dotabap já lista 236 projetos. Claro, alguns desses são contribuídos pela própria SAP, como o code-pal-for-abap (uma ferramenta que ajuda a aderir às regras do Clean ABAP), mas basta rolar a página inicial do dotabap pra ver todo tipo de integração, API, prova de conceito e até jogos (sempre tem algum doido querendo fazer joguinhos em ABAP).

Um dos efeitos dessa abertura é ter um canal mais amplo para obter, discutir e aproveitar conteúdos da própria SAP numa plataforma que é amigável aos desenvolvedores. Isso é muito importante, mas não é o principal efeito do código aberto. Na minha visão, o ponto mais importante da abertura – e também o argumento principal desse post – é que fica possível **compartilhar** e **ter retorno** da comunidade sobre o código. Publicando código num repositório git aberto, seja qual for a linguagem, expõe aquilo que você considera importante para que mais pessoas vejam, usem, aprovem ou reprovem, comentem e sugiram – ou até mesmo façam por si próprias – melhorias ou mudanças. E mesmo que você use apenas repositórios git privados, eles servem como pontos de backup e recuperação, com histórico independente do sistema de transporte, e podem ficar restritos apenas a uma empresa ou a um cliente. Repositórios git privados podem ser usados também para revisões (usamos assim dentro da SAP!), através do mecanismo de pull requests.

Em 2020/2021, eu participei de alguns projetos que envolviam SAPUI5 e/ou CAP (a versão Node), usando como plataforma o runtime Cloud Foundry no SAP BTP. Desenvolvendo views SAPUI5 no VSCode, eu acabei criando o Fiori XML Lint, uma extensão feita em Typescript pra me ajudar a seguir os padrões do Fiori em views XML. Além disso,

como eu tinha que lidar com ambientes Cloud Foundry e Neo no BTP, eu criei nesse período mais duas ferramentas, dessa vez em Node e linha de comando, pra me ajudar também: o cf-destination-proxy, um proxy pra acessar destinations no Cloud Foundry do BTP a partir de uma IDE local (obviamente o VSCode também 😉), e o neo-deploy-mta-oauth, pra fazer deploys de MTAs num ambiente SAP BTP Neo usando autenticação OAuth sem precisar do Jenkins.

Eu criei essas três ferramentas primeiramente porque eu tinha problemas **reais** que precisavam ser resolvidos pra que eu tivesse uma vida tranquila no desenvolvimento dos projetos. Mas aí eu percebi que podem ser problemas reais que **outras pessoas** também enfrentam, e então eu publiquei as ferramentas nas plataformas onde elas poderiam ser encontradas e, porque não, **analisadas** por outros desenvolvedores. Se você explorar os links que eu compartilhei no último parágrafo, você eventualmente vai chegar nos repositórios do GitHub de onde elas se originam (mas não precisa sofrer, se quiser pode entrar diretamente no meu perfil lá). E aparentemente eu estava certo sobre outras pessoas terem necessidades parecidas com as minhas: o Fiori XML Lint já tem quase 5.000 downloads, o cf-destination-proxy tem mais de 1.700 e o neo-deploy-mta-oauth tem mais de 900 (apesar desse último número ser meio inflado por se tratar de uma ferramenta que pode ser usada pra CI/CD, mas mesmo assim). Não são números astronômicos, mas a dopamina que eu recebo vendo eles subindo ou quando alguém abre uma PR é muito bem-vinda 😄

É claro que o ABAP é um mundo à parte que ainda tem suas próprias restrições quando se trata de código aberto. Mas nada impede você, caro companheiro(a) zumbi, de compartilhar as ferramentas que você cria pra te ajudar no dia-a-dia com a comunidade (claro, sempre que isso seja legalmente possível), e crescer e aprender muito durante o processo. É por causa desse espírito altruísta e colaborativo que temos coisas em ABAP como o próprio abapGit, o abap2xlsx (que sinceramente é a resposta pra todas as necessidades de Excel que você possa ter em ABAP), o abaplint, e muitos outros.

Então, a mensagem é: compartilhe as coisas que você faz com o mundo, sejam feitas em ABAP ou em qualquer outra linguagem, sejam ferramentas essenciais ou joguinhos doidos, sejam simples e inúteis ou complexas e salvadoras. O mundo agradece e você vai ser um zumbi mais feliz.

Comentários

> **Giovanni Mileo** — *07/11/2023 22:05*
>
> Muitoooo top ! Continuem com o excelentes trabalho

> **Bruno Pidde** — *30/03/2023 18:29*
>
> AAAAAêêêêêê, como é bom ver uma postagem nova por aqui.
> Muito bom o conteúdo Leo. Seria MUITO LEGAL um novo episódio de podcast ein? hehe

> > **Leo Schmidt** — *21/04/2023 16:54*
> >
> > Valeu Bruno!
> >
> > Seria muito legal mesmo, mas pensa num povo difícil de reunir

> **Guilherme Machado** — *04/01/2023 16:17*

> Ah e por favor não desistam do “ABAPZombie Guide to ABAP”

> **Guilherme Machado** — *04/01/2023 16:05*
>
> Lol New post, este blog é simplesmente sensacional a bagagem que é transmitida pelo conteudo que vocês publicam é simplesmente insano, Obrigado por compartilhar os modos “Zombie” essa experiencia principalmente a quem esta iniciando nesse mundo ajuda muito.

> > **Leo Schmidt** — *17/01/2023 09:51*
> >
> > Valeu Guilherme!