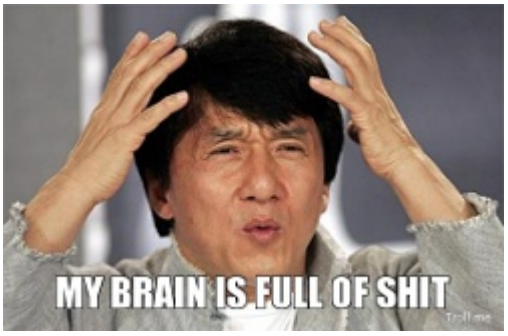
War Report – Estrutura é tudo, camarada...

16/05/2012 12:00

É caro amigo zumbi, consegui sobreviver (pelo menos por enquanto...). Em um mundo onde regras sistemáticas são escritas em sequência, e onde não existe muita preocupação com o reuso, eu venho sobrevivendo. Travando batalhas diárias contra a opressão implantanda involuntariamente em mentes facilmente influenciadas por padrões datados. Obviamente, essas palavras não fariam sentido se não fosse para arrancar uma cara de **WTF** de você enquanto lê essa balela!



E aí, o que andou rolando desde fevereiro, quando **fiz este post**?

Lembrando que a idéia aqui é retratar as minhas experiências com a minha decisão de usar OO na grande maioria dos meus desenvolvimentos ABAP, a partir desse ano, e isso não quer dizer que meus insights ou decisões sejam a verdade absoluta. É um mero "Report", e você pode usar as informações como quiser 😉

O mundo não acabou e não briguei com ninguém (*ainda*). Só tive que explicar para um amigo funcional que curte debuggar o que era aquele monte de "setas" espalhadas pelo código. Interessante que ele se preocupou em entender, diferente de quase metade da galera que eu já vi por aí, que gosta do feijão com arroz de sempre e engole a salada por osmose.

Eu brinco com outras linguagens de programação, e, por conta disso, optei por não usar a SE24 para declara as classes. Se eu tiver que transformar algo em global, primeiro faço local e depois importo para a SE24, como explicado **neste post**. E usando o editor novo + se80, a coisa fica rápida, é questão de costume. A sintaxe das declarações de classes/interfaces/whatever no ABAP é um pouco travada, mas depois de um tempo você se acostuma.

Outro fator interessante é a questão do **tempo** de desenvolvimento. Em alguns comentários aqui do site, muitas pessoas já falaram que desenvolver em OO leva "mais tempo". **Discordo totalmente**. O que leva tempo é pensar em como deixar o seu software bom, e não fazer um mero quebra-galho. Se você se preocupa em deixar seu programa bem estruturado, de fácil manutenção e entendimento, o ABAP Objects vai adiantar seu tempo de programação, e não atrasar.

Aliás, esse ponto do entendimento é interessante... Se você fugir dos padrões de nomenclaturas idiotas e tentar nomear suas classes com coisas que fazem sentido, seu programa vai ficar muito simples de ser lido. Eu passei a comentar menos meus códigos, afinal, se as variáveis e métodos já dizem por si só o que está sendo feito, porque colocar comentários banais? Há muito tempo atrás eu li que "*se o seu código for REALMENTE bom, ele não precisa de comentários*". Não acho que isso seja verdade absoluta, mas isso faz mais sentido para hoje em dia. Mesmo no ABAP. Por exemplo, me diga o que o mapeamento de call transaction abaixo está fazendo na transação que eu inventei:

```
o_bdc->add_main_screen( program = 'SAPPROGDOIDO'
                        dynpro  = 1000 ).

o_bdc->add_value( fieldname = 'P_PLANTA'
```

```
                value     = 'WERKS' ).

o_bdc->add_value( fieldname = 'S_MATNR-LOW'
                value     = '123455' ).

o_bdc->add_enter( ).

o_bdc->add_execute( ).

o_bdc->run( ).
```

Entrou, preencheu dois parâmetros, ENTER, F8. Divertido né? Não importa quais estruturas os métodos preencham, o que importa é que existe uma camada que deixa a leitura do código mais intuitiva. Ah, e como é uma classe, você pode tornar isso global e utilizar em várias programas. Já aquele seu PERFORM de preencher BDC...

```
PERFORM f_add_bdc USING:
'SAPPROGDOIDO' '1000' 'X' '' ''  "Tela Principal
''             ''     ''  'P_PLANTA' 'WERKS'
''             ''     ''  'S_MATNR-LOW' '12345'
''             ''     ''  'BDC_OKCODE' '/00'    "ENTER
''             ''     ''  'BDC_OKCODE' '=ONLI'. "Executar (F8)

CALL TRANSACTION blablabla USING t_bdc.
```

Easy de ler e entender o que acontece, falae? **#NOT** E naquela mapeamentos gigantescos? Dá gosto de ver como fica interessante ler e achar o que está errado, não é não? 😛

Estou usando o CALL TRANSACTION como exemplo, mas esse conceito se expande para **TODO** o seu código. Vai da sua imaginação, caro zumbiwan (padawan zumbi).

Mas e aí, o que você acha dessa coisa toda? Código intuitivo, padrões toscos de nomenclatura, comentários banais, preguiça de pensar e reutilização de classes?

Na próximo report talvez eu conte sobre uma briga... ou talvez eu diga que mais algum amigo entrou na brincadeira!

Abraços!

Comentários

> **Diego Henrique Gomes** — *08/06/2012 19:13*
>
> Mauricião, to nessa também, comecei a estudar a parte OO, para em breve desenvolver tudo de tal maneira. Mas na hora do aperto, como você disse. Só nos resta ir por onde melhor conhecemos.
> Você poderia postar um relatório simples , com algumas explicações... acho que todo mundo curtiria...

> > **Mauricio Cruz** — *10/06/2012 19:39*
> >
> > Eae Diego, blz?
> >
> > Cara, o abap101.com tem vários exemplos de aplicações de orientação a objetos e MVC em programas ABAPs. Vale a pena ler os últimos posts, pois desde o ano passado eles não postam mais nada procedural, somente exemplos com orientação a objetos.
> >
> > Anyway, OO vai aparecer bastante por aqui, fique tranquilo!

> Abraços!

Haroldo — *30/05/2012 13:52*

Querendo ou não, OO está aí pessoal.
Se não for no ABAP, vai ser em outra linguagem.

Se todos sabemos que sempre terá alguem para nos apressar no projeto, vamos nos antecipar então. (fazendo OO em horas vagas)
Legal você pegar um código e saber que o cara que escreveu se preocupou em organizar e deixar tudo fácil, caso algum consultor precise modificar alguma coisa.

Minha opnião é a seguinte:
Pra saber o melhor, você tem que ter pelo menos noção de como era antes. Ou seja, acho indispensável conhecer o procedural antes do OO.

Abraços...

Ricardo — *18/05/2012 08:45*

Opa, muito bom o post. tb acho estruturado bem melhor de se ler e dar manutenção.

abraços

Custodio — *17/05/2012 01:55*

E bem quando eu havia deletado o ABAP Zombie dos favoritos, eis que aparece um novo post!

Concordo com a maioria do seu post, mas achei o exemplo bem ruim. Nada te impede de colcoar nomes marotos nas suas subrotinas, algo como:

perform:
add_main_screen using ‘SAPPROGDOIDO’ dynpro = 1000 ,
add_value using ‘P_PLANTA’ ‘WERKS’ ,
add_value using ‘S_MATNR-LOW’ ‘123455’,
add_enter,
add_execute,
run.

Inclusive, nao gosto da ideia de ter um methodo para cada acao (ente, execute, double click, etc), periro usar como parametro.

Isto posto, tenho programado majoritariamente em OO desde 2006 e quase que exclusivamente em OO desde 2009.

Estou num projeto com 2 SAP Mentors, um deles escreveu a metodologia/padrao de desenvolvimento onde eh bem explicito que todo novo desenvolvimento DEVE ser em OO. Mas, inexplicavelmente, quase ninguem segue e nao sei por que raios eles aprovam...

> **Mauricio Cruz** — *17/05/2012 10:31*

Uma das explicações para os desenvolvimentos passarem é dinheiro, simples assim.

Tomei de base para o exemplo o padrão de criação de FORMs para BDC que eu tenho certeza que mais de 80% do público do blog utiliza no dia-a-dia, vai por mim 🙂

E se você tiver um exemplo que ache mais legal, por favor, coloque nos comentários!

ABAPdepressao — *16/05/2012 15:04*

Zumbi depressivo reportando.

Desde que eu vi o post aqui, adotei a ideia pra mim e todo desenvolvimento novo que eu fiz desde então foi OO, e assim vai ser. Não arrumei briga ainda, espero não arrumar, mas acho que não vai demorar muito...

Eu entendo que a galera acha que demora mais pra desenvolver usando essa metodologia porque pra usar OO de maneira eficiente e correta você tem que parar de pensar tudo "em sequência", que é como a grande maioria aprende, e identificar onde estão os objetos que vão fazer parte do programa, como eles vão trocar mensagens entre si e com o usuário, e quais são as suas propriedades e métodos.

É um tipo de planejamento que não existe quando você escreve programas procedurais, principalmente reports, onde você acaba ficando de certa forma "preso" pela sequência de eventos. Mas assim que isso está feito as classes saem quase que naturalmente.

Um jeito muito bacana de desenvolver a ideia de trabalhar só com OO é começar a mexer com WebDynpro. Você é **obrigado** a pensar em OO e em MVC pra fazer uma aplicação WebDynpro que funcione. É f***, não tem uma documentação bacana (além dos PDFs / DOCs que você consegue "na camaradagem"), dá pra ficar bem perdido com a API, mas é o jeito.

Citando a documentação da SAP:

Subroutines are obsolete. Do not create new subroutines in new programs.

Mauricio Cruz — *16/05/2012 15:17*

Opa, animal saber que vc entrou na empreitada também. A única parte triste é que isso te deixa depressivo.

Diz a lenda que um dos manos aqui do blog vai trazer alguns posts de Webdynpro em breve, mas por hora é só uma lenda.

Abraços, viva o caos dos módulos em 2012

ABAPdepressao — *16/05/2012 15:21*

O que me deixa depressivo mesmo é ter que usar MACROS num programa escrito no ano do nosso senhor de DOIS MIL E DOZE por causa de um processo burro. O resto feito em OO tá bacana 😀

[]s

Mauricio Cruz — *16/05/2012 15:24*

> Você deveria ser o ABAPdaMeiaDepressão então 😛 Esse negócio de dar manutenção em código antigo é engraçado.. antes eu ficava PUTO, depois eu ficava "OK", agora acho divertido tentar entender o que o cara pensou em 1910 para programar o código em 2012. E esse cara deve ter, no máximo, 25 anos, eeeeeeee

> **Fawcs** — *16/05/2012 14:16*
>
> Bom, eu até hoje, o que eu vi de OO foi SÓ na academia(thanks prof. Furlan)! E eu acho bem mais legal tralhar com isso. O problema é que, no meu caso, trabalho no suporte e então só vou dando manutenção em algumas linhas de certos programas... Não sei, pra mim parece mais complicado tentar implementar OO nessas condições =P

> **Mauricio Cruz** — *16/05/2012 14:18*
>
> Mas, imaginando que você vai começar a trabalhar em projetos de implantação... você tentaria partir para o OO? Aliás, você acha que, se boa parte dos programas que você altera aí no suporte tivessem sido feitos utilizando ABAP Objects, seu trabalho seria mais fácil?

> **Fawcs** — *16/05/2012 15:03*
>
> Bom, eu ainda sou Zombiwan(fiz academia mês passado!), então só peguei chamado facinho até agora, mas imagino que com OO ficaria mais fácil de dar manutenção , eu acho que os programas ficam mais organizados e mais intuitivos, é o que você falou de colocar um nome bom na classe e nos métodos, fica fácil de ler o programa.
>
> Mas ai eu sou da opinião que você não precisa usar classes e métodos pra programar em OO, ontem mesmo eu peguei um programa que é basicamente só perform e eu consegui achar o que eu queria só pelos nomes e pela organização.
>
> Sobre o projeto, não sei, como eu estou aprendendo ainda, to muito mais preocupado em aprender outras coisas=)

> **Mauricio Cruz** — *16/05/2012 15:18*
>
> Só para acertar seu comentário: se não existe o conceito de "Objeto", logo, não podemos considerar que o código X foi feito com OO. O que você encontrou foi um código bacana feito de maneira procedural.
>
> Abraços e força aí nas correções de códigos de 1930!

> **Fawcs** — *25/05/2012 09:32*
>
> Ah sim.. mas de vez em quando rola um desenvolvimento do 0... e aí existe limite de hora pra responder os chamados, complica pra um novato fazer em OO =)