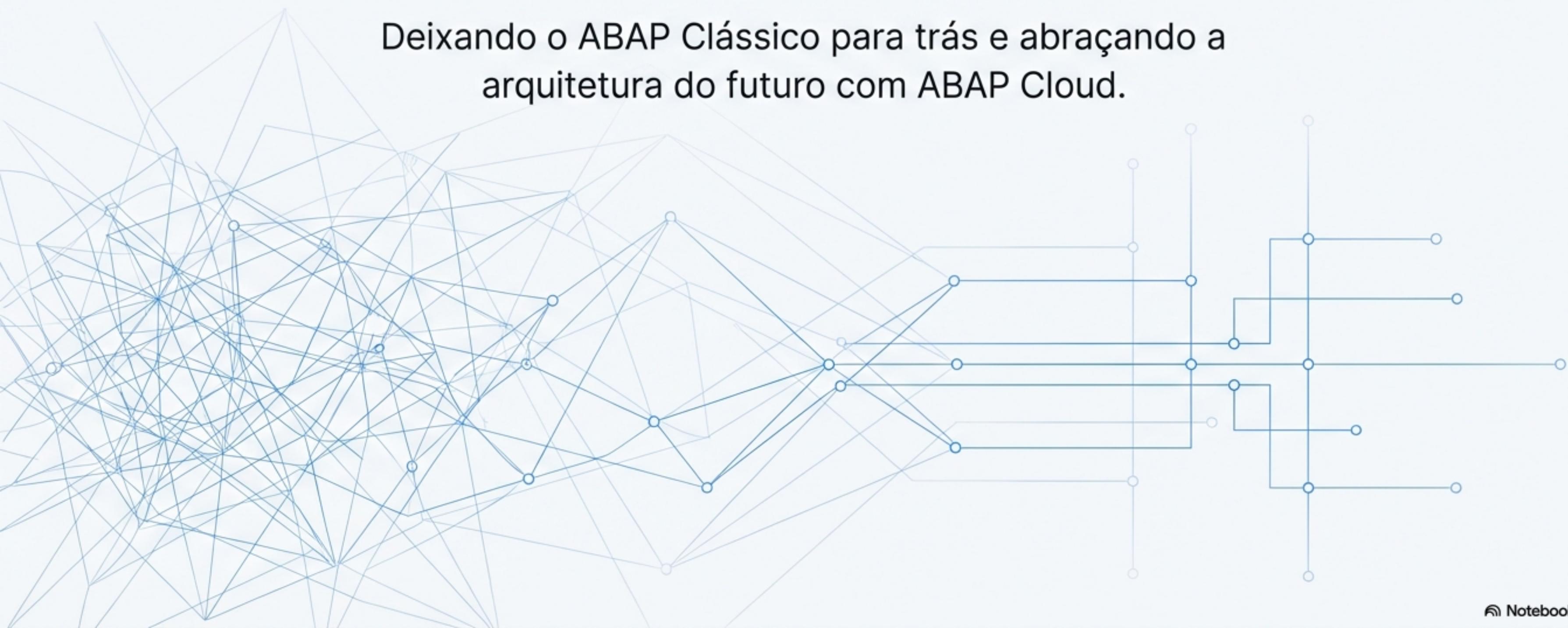
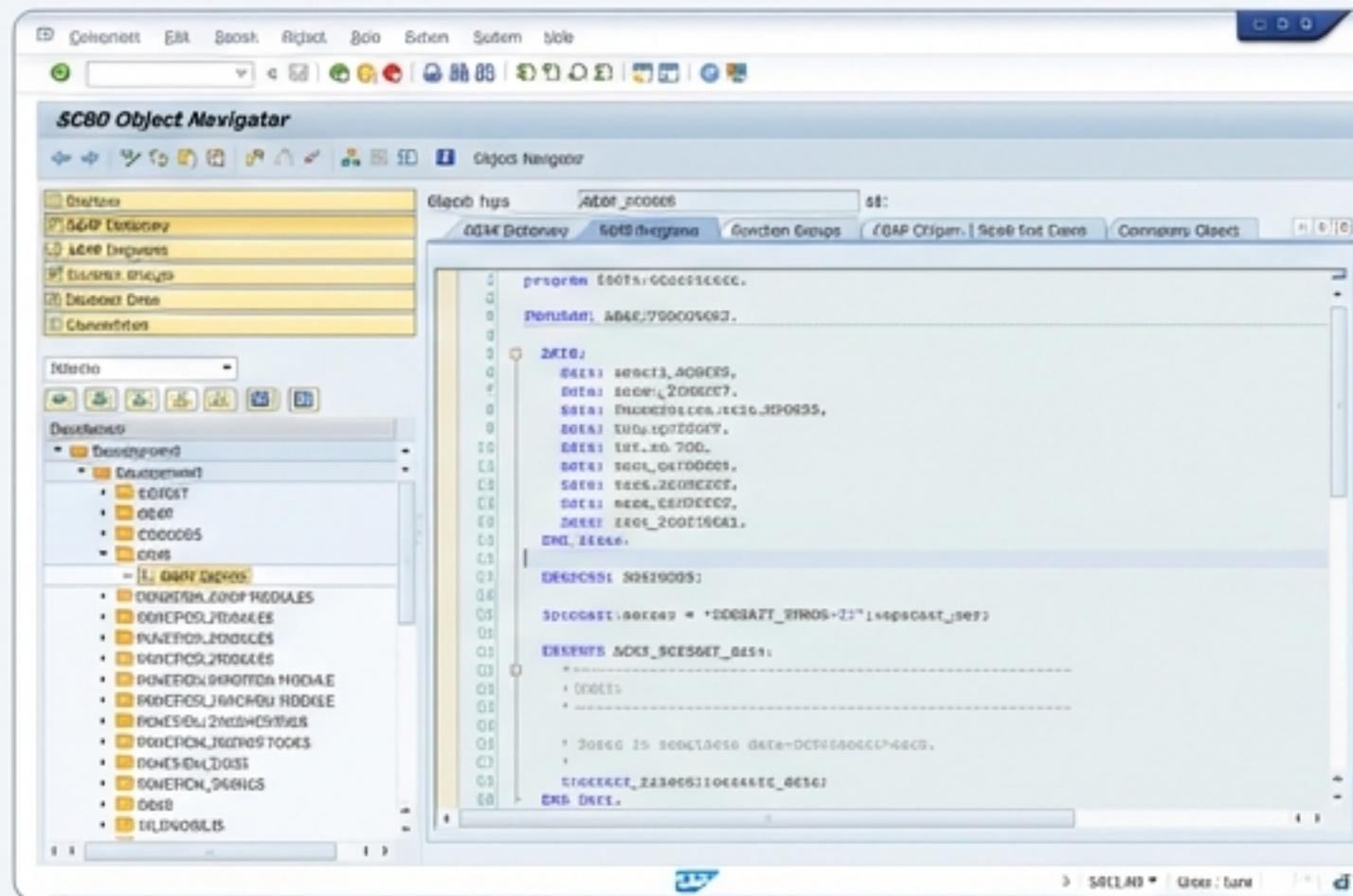


Iniciando a Jornada: O Paradigma do ABAP Moderno

Deixando o ABAP Clássico para trás e abraçando a arquitetura do futuro com ABAP Cloud.



Não é apenas uma mudança de interface. É uma mudança de filosofia.



O PASSADO: ABAP CLÁSSICO

A ferramenta que reinou por décadas (SE80) dá lugar a um ambiente integrado, aberto e essencial para o desenvolvimento em nuvem.

The screenshot shows the SAP ABAP Development Tools interface. It features a 'Project Explorer' on the left with a tree view of packages and classes, and a 'Code Editor' on the right displaying modern ABAP code. The code includes annotations like '@EndUserText' and '@AccessControl.authorizationCheck', and uses a more modular and object-oriented syntax compared to the classic ABAP shown in the first screenshot.

O FUTURO: ABAP CLOUD

As Vantagens Decisivas do Ambiente Moderno (ADT)



Velocidade e Refatoração

Ferramentas poderosas que não existem no SAP GUI: renomeie métodos em todo o sistema, extraia constantes e formate código (“Pretty Printer”) com um clique.



Visão Multi-Sistema

Visualize e compare códigos de diferentes sistemas (Desenvolvimento vs. Qualidade) lado a lado na mesma interface, sem múltiplas janelas de login.



Funcionalidades Exclusivas da Nuvem

O desenvolvimento de Core Data Services (CDS Views), Behavior Definitions (RAP) e Service Bindings é suportado **apenas** no ADT.

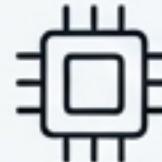


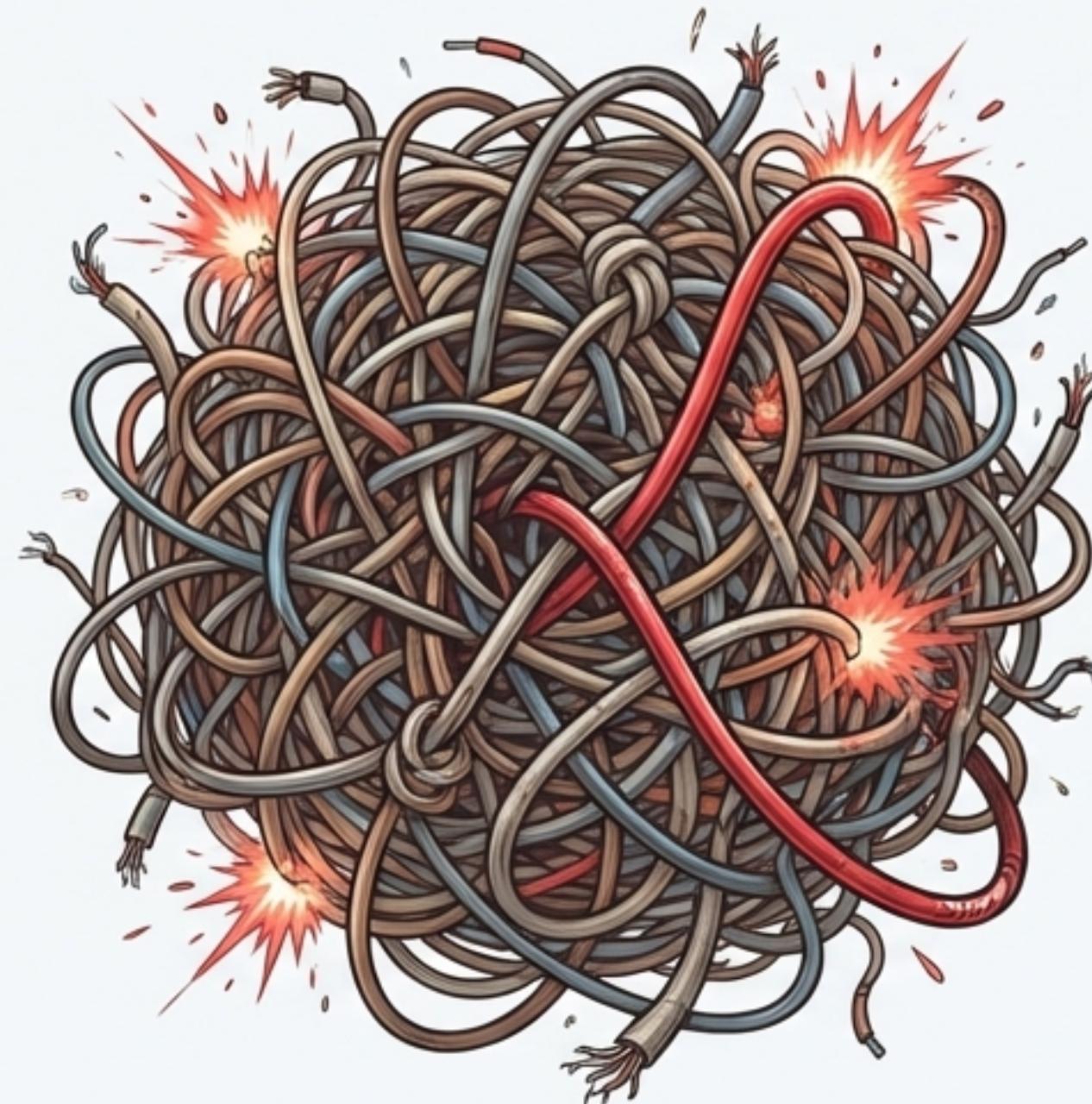
Integração com DevOps

Integração nativa com **abapGit** para versionamento de código e pipelines de CI/CD, alinhando o ABAP com as práticas modernas de engenharia de software.

O Legado do 'Código Espaguete': Por que a Mudança Foi Necessária

O Problema: Superpoderes Perigosos

-  Acesso irrestrito a qualquer tabela, inclusive as internas da SAP.
-  Modificação do comportamento padrão do núcleo do sistema.
-  Acesso direto e arriscado ao sistema operacional.



A Consequência Inevitável: O "Inferno" do Upgrade

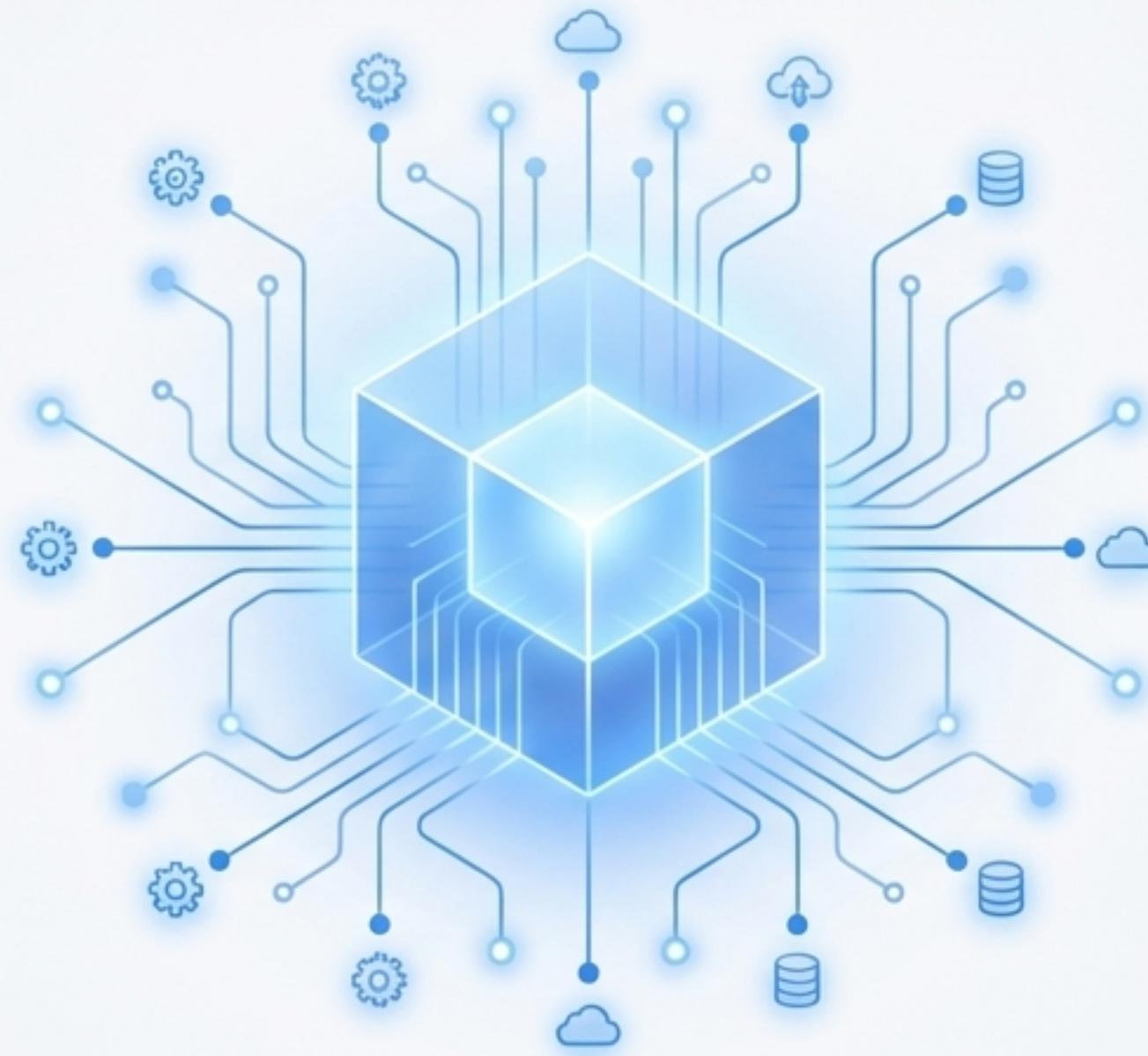
-  Códigos personalizados quebravam a cada atualização, pois dependiam de estruturas internas que mudavam.
-  Projetos de upgrade tornavam-se caros e demorados, dominados pelas transações SPDD e SPAU.

A Solução Estratégica: Clean Core

A estratégia para garantir que o sistema ERP possa ser atualizado automaticamente (como seu smartphone) sem quebrar as customizações.

Como funciona:

O ABAP Cloud impõe restrições técnicas rigorosas para proteger o núcleo do sistema e garantir a estabilidade a longo prazo.



As Regras do Novo Mundo: Como o Clean Core é Aplicado



Language Version 5

O compilador bloqueia comandos obsoletos ou perigosos. Não é mais possível usar `CALL SCREEN` (Dynpros), `WRITE` (Listas clássicas), ou acesso direto a arquivos do servidor.

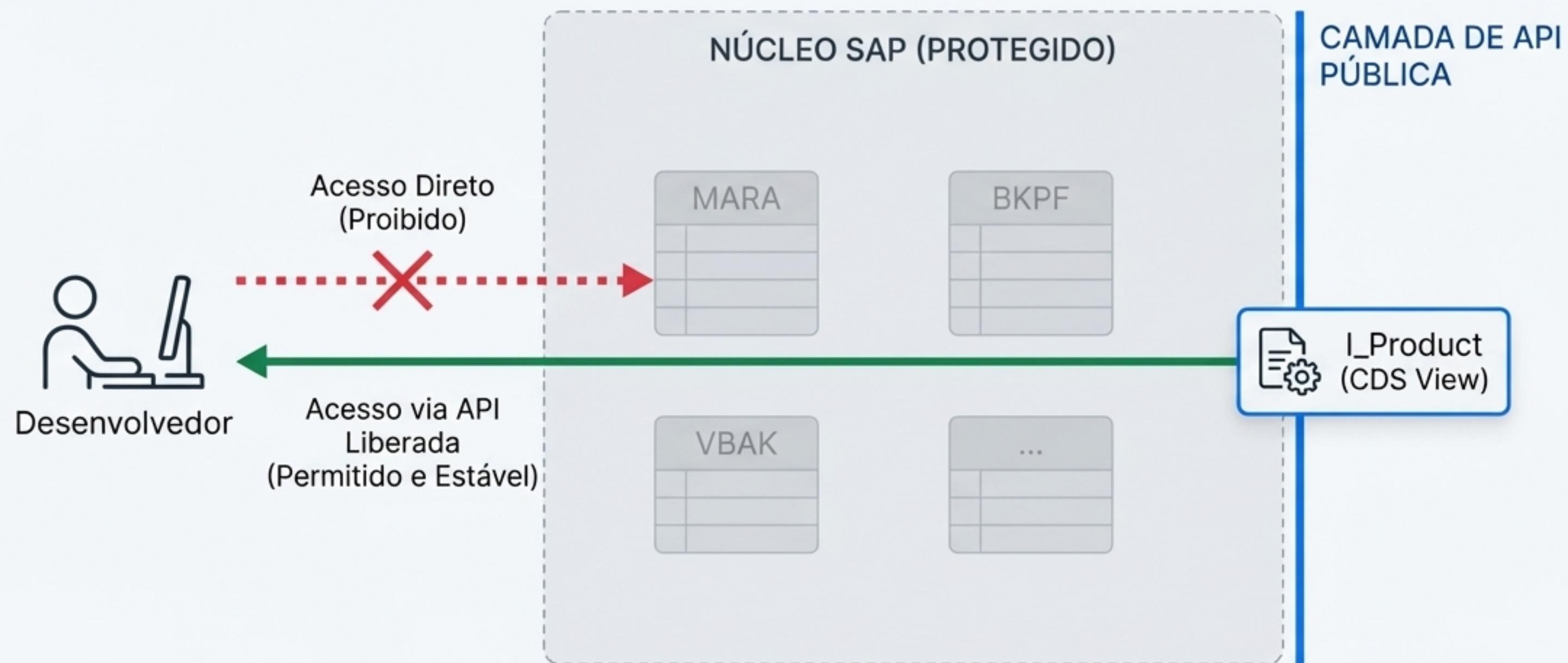


Released Objects (A Regra de Ouro)

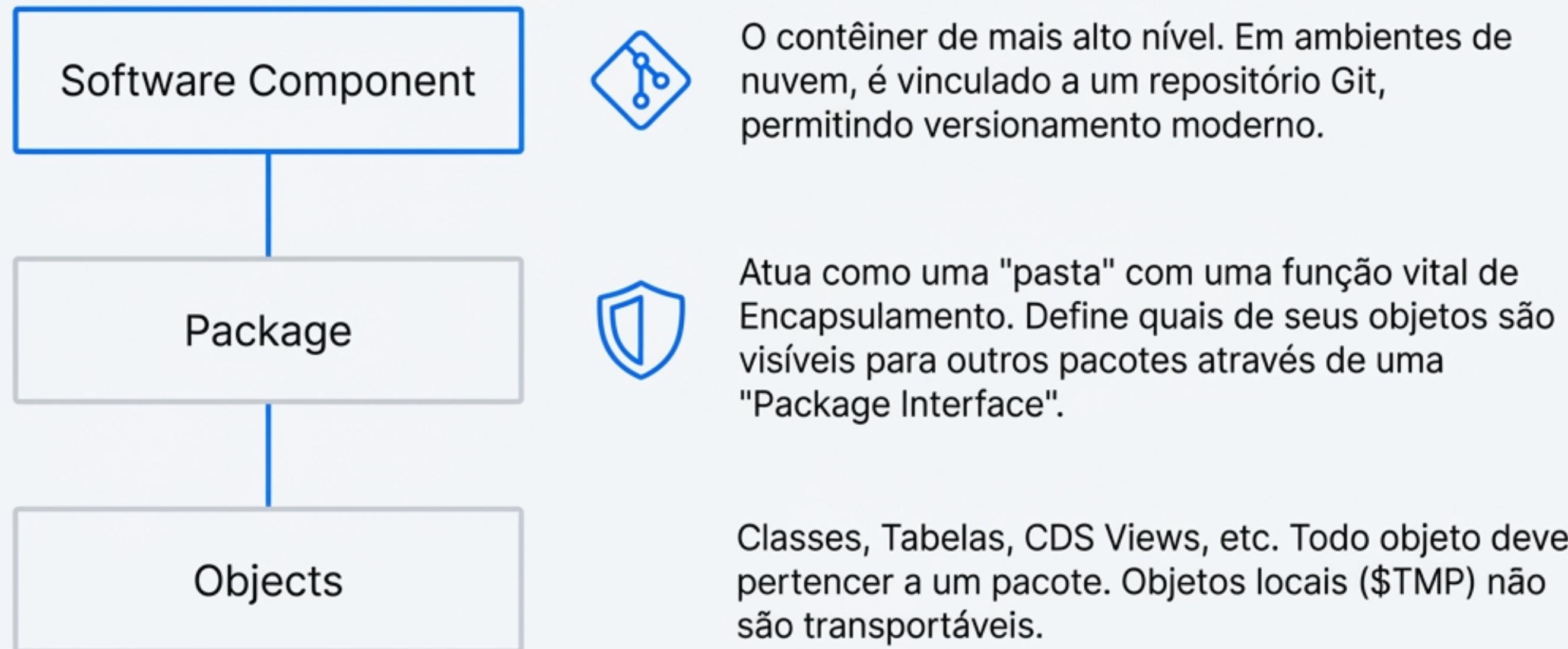
Você só pode referenciar objetos SAP que foram explicitamente marcados como APIs PÚBLICAS (Whitelisted). Qualquer outra chamada resulta em erro de sintaxe.

A Regra de Ouro em Detalhes: O Contrato dos "Released Objects"

Você não acessa mais a "sala de máquinas". Você usa a "recepção" oficial.



A Nova Arquitetura de Código: Estrutura para Reuso e Desacoplamento



Seu Primeiro Programa: Adeus, REPORT.

~~REPORT~~



```
CLASS ...  
IMPLEMENTS  
if_oo_adt_classrun.
```

No ABAP Cloud, a lógica de apresentação (UI) é totalmente separada da lógica de backend.

Para testar a lógica, utilizamos uma **Classe ABAP Global** que implementa a interface `if_oo_adt_classrun`, que funciona como um 'contrato' para tornar a classe executável.

Anatomia do 'Hello World' Moderno

```
CLASS zcl_hello_world DEFINITION PUBLIC FINAL CREATE PUBLIC.  
  PUBLIC SECTION.
```

```
    INTERFACES if_oo_adt_classrun. ←  
  ENDCLASS.
```

```
CLASS zcl_hello_world IMPLEMENTATION.  
  METHOD if_oo_adt_classrun~main.
```

```
    out->write( 'Hello World! Bem-vindo ao ABAP Moderno.' ). ←
```

```
    DATA(lv_date) = cl_abap_context_info->get_system_date( ). ←
```

```
    out->write( |A data de hoje no servidor é: { lv_date DATE = ISO }| ).
```

```
  ENDMETHOD.  
ENDCLASS.
```

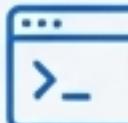
1. O contrato que torna a classe executável via console com F9.

2. O novo 'WRITE'. Envia a saída para a aba 'Console' no Eclipse.

3. Exemplo de uma API pública (Released Object) para obter dados do sistema, substituindo variáveis como `sy-datum`.

4. A sintaxe moderna e poderosa para concatenação: String Templates.

A Transformação em Resumo: Clássico vs. Moderno

Recurso	ABAP Clássico (Legacy)	ABAP Moderno (Cloud)
 IDE Principal	SAP GUI (SE80, SE38)	Eclipse com ADT
 Saída de Texto	Comando `WRITE 'Texto'.`	Método `out->write('Texto').`
 Tipo de Programa	Report (`REPORT z...`)	Classe Global com `if_oo_adt_classrun`
 Leitura de Dados	`SELECT * FROM tabela_sap`	`SELECT * FROM cds_view_liberada`
 Telas (UI)	Dynpro / Web Dynpro	SAP Fiori (UI5 / Fiori Elements)
 Variáveis de Sistema	`sy-datum`, `sy-uname`	Classes API como `cl_abap_context_info`

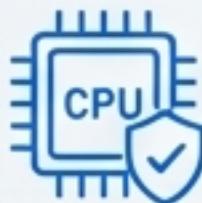
Glossário Essencial: A Nova Linguagem do Desenvolvedor ABAP



ADT (ABAP Development Tools): IDE baseada em Eclipse, mandatória para desenvolvimento ABAP moderno (RAP, CDS, Cloud).



ABAP Cloud: Modelo de desenvolvimento restrito focado em 'Clean Core', que proíbe acesso direto ao sistema e obriga o uso de APIs liberadas.



Clean Core: Estratégia arquitetural para manter o núcleo do ERP livre de modificações, garantindo upgrades seguros.

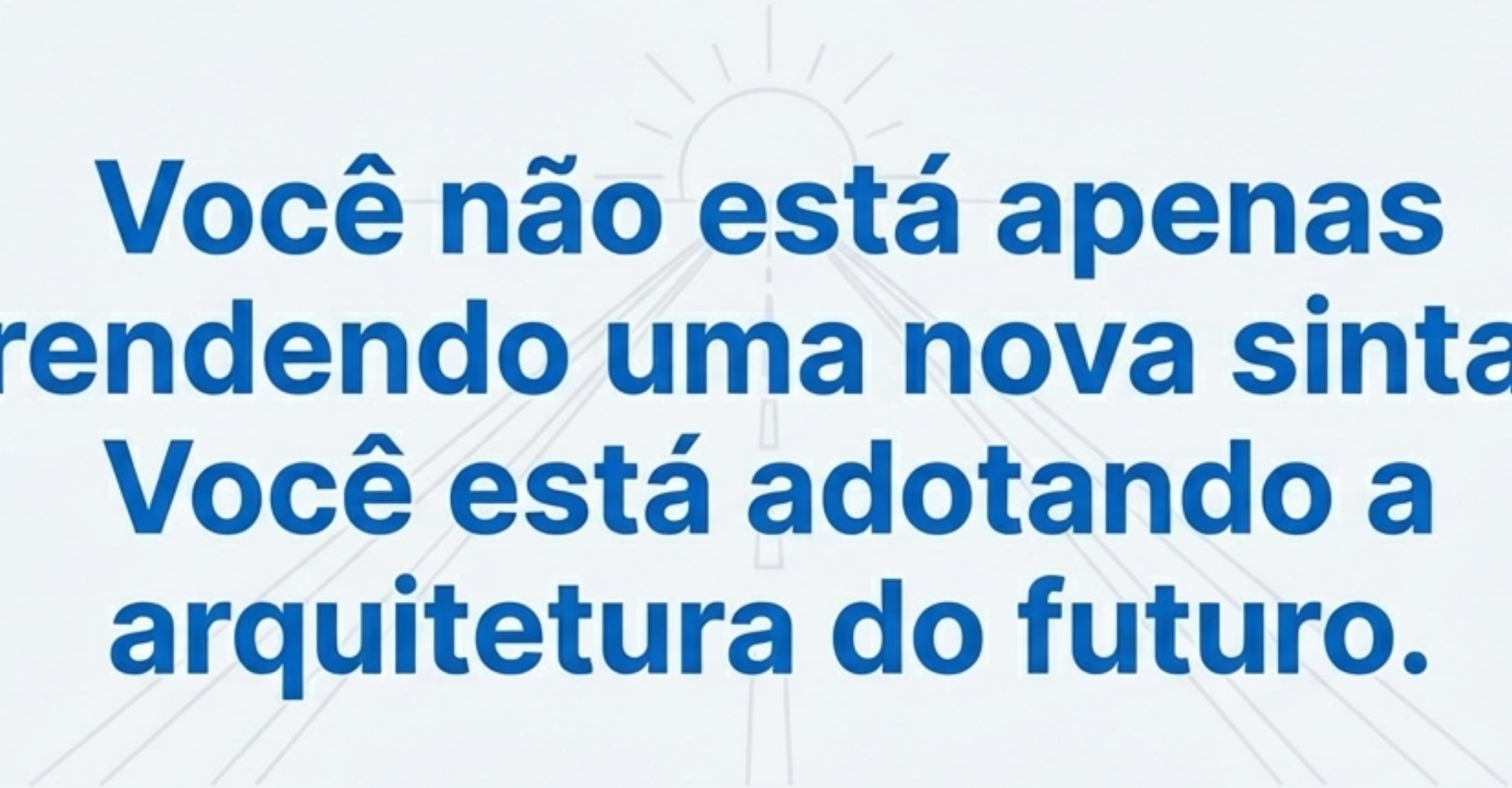


Released Object: Artefato SAP (Classe, CDS) marcado com um contrato de estabilidade. Apenas estes podem ser usados no ABAP Cloud.



abapGit: Cliente Git para ABAP, essencial para versionamento de código e transporte em ambientes de nuvem e BTP.

O Fim desta Jornada... e o Início da Próxima.



**Você não está apenas
aprendendo uma nova sintaxe.
Você está adotando a
arquitetura do futuro.**

Dominar o paradigma do ABAP Cloud é o que separa o desenvolvedor
de hoje do arquiteto de soluções de amanhã.