DYNAMICS MINDS

Microsoft Dynamics 365 and Power Platform Community Conference

MAY 26 – 28, 2025
Portorož, Slovenia, Europe

FINANCE & OPERATIONS

BUSINESS CENTRAL

CUSTOMER ENGAGEMENT

POWER PLATFORM

5th TRACK

PARTNER BizTRACK

DYNAMICS MINDS

Microsoft Dynamics 365
and Power Platform Community Conference

MAY 26 – 28, 2025
Portorož, Slovenia, Europe

An introduction to BC development and delivery

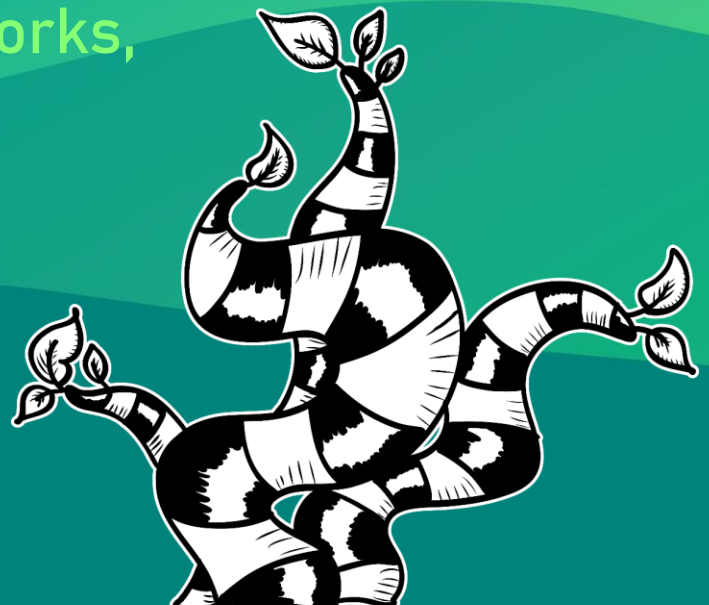
Tobias Fenster

Managing Director @ 4PS, Chief Engineer @ Hilti

tobiasfenster.io

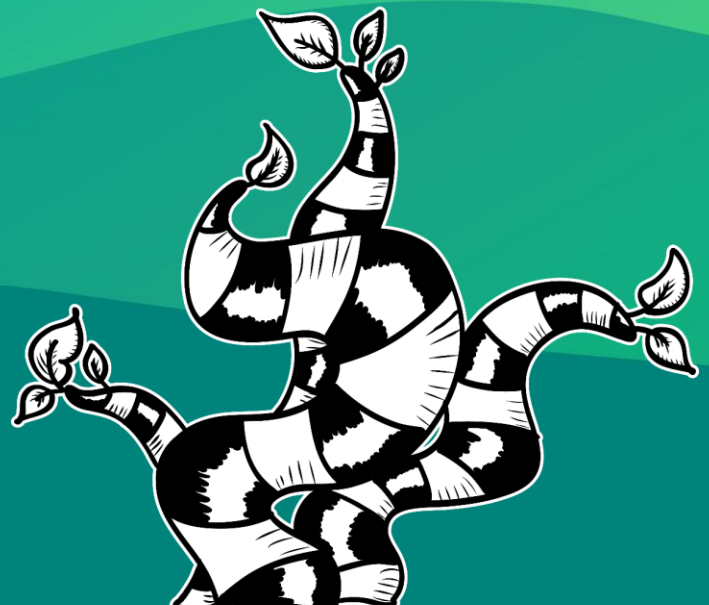
Disclaimer!

This is meant for people new to or interested in Business Central development and delivery. If you already know how this works, you probably won't learn a lot.



DYNAMICS MINDS

Development in BC (and NAV)



DYNAMICS MINDS

General approach (past and present)

Customer code

Partner(s) code

Local Microsoft code (DE, UK, …)

International Microsoft code

DYNAMICS MINDS

Implementation reality (past)

Implementation reality (present)

- You can't change or remove Microsoft code, you can only extend it
  - With a few exceptions
  - And a concept to replace whole "modules" if they are designed for it
- You can create for the whole world (AppSource) or for a customer ("Per Tenant Extension" PTE)
  - But you must say which localizations you support
- Overall a lot cleaner, better to understand and maintain
  - But also more limited

DYNAMICS MINDS

General structure

- Business Central with classic 3-tier architecture: database, middle/application layer, (web) client
- Code is imported into the database, the rest is generated (more on that later)
- Because of generative approach, developer has far-reaching, but not complete influence on all layers

DYNAMICS MINDS

Tooling (past): C/SIDE

Codeunit 50000 CrInvForCust10000And70000 - C/AL Editor

```
Documentation()
OnRun()
LOCAL CreateInvoiceDocument(VAR SalesHeader : Record "Sales Header")
WITH SalesHeader DO BEGIN
  INIT;
  "Document Type" := "Document Type"::Invoice;
  "No." := '';
  INSERT(TRUE);
END;

LOCAL TestIfCustomerIsBlocked()
WITH Customer DO BEGIN
  GET('10000');
  TESTFIELD(Blocked, Blocked::" ");
END;

LOCAL SelectCustomer10000(VAR SalesHeader : Record "Sales Header")
WITH SalesHeader DO
  VALIDATE("Sell-to Customer No.", '10000');

LOCAL SaveHeaderToDatabase(VAR SalesHeader : Record "Sales Header")
WITH SalesHeader DO
  MODIFY(TRUE);

LOCAL CreateSalesLine(VAR SalesHeader : Record "Sales Header";VAR SalesLine : Record "Sales Line")
LOCAL TestIfItemIsBlocked()
LOCAL SelectItem70000(VAR SalesLine : Record "Sales Line")
LOCAL SaveLineToDatabase(VAR SalesLine : Record "Sales Line")
LOCAL OpenSalesInvoicePage(VAR SalesHeader : Record "Sales Header")
```

Tooling (present): VS Code + AL extension

File Edit Selection View Go ··· 'S Base Application DE - PurchaseOrder4PD.PageExt.al - BC - Visual Studio Code - Ir

EXPLORER

- 4PS BASE APPLICATION DE
  - app
    - src
      - Subcontracting
        - contract
          - SubcontractorContracts4PD.Page.al
        - invoice
        - permission
        - reduction
        - setup
        - ActionHandler4PD.Codeunit.al
        - PrintdataBuffer4PD.Table.al
        - PurchaseOrder4PD.PageExt.al
        - PurchaseProcessingRC.PageExt.al
        - PurchaseQuoteExt4PD.PageExt.al
        - VendorCardExt4PD.PageExt.al
        - VendorExt4PD.TableExt.al
        - VendorListExt4PD.PageExt.al
      - Telemetry
      - whitevision
      - .gitkeep
    - Translations
    - .gitignore
    - .objidconfig
    - 4PS B.V._4PS Base Application DE_25.1.0.0.a...
    - app.json ↓M
    - AppSourceCop.json
  - test
- OUTLINE
- TIMELINE

AL PurchaseOrder4PD.PageExt.al

app > src > Subcontracting > AL PurchaseOrder4PD.PageExt.al > {} PageExtension 11316765 "Purchase Order 4PD"

0 references | Simon Häuser, 7 months ago | 2 authors (Simon Häuser and one other)

```
1  pageextension 11316765 "Purchase Order 4PD" extends "Purchase Order"      Simon Häuser, 2 ye
2  {
       Simon Häuser, 15 months ago | 2 authors (Simon Häuser and one other)
3      layout
4      {
5          movebefore("Invoice Received Date"; "Subcontracting Invoice")
6          addafter("Subcontracting Invoice")
7          {
               0 references | Simon Häuser, 15 months ago | 2 authors (Simon Häuser and one other)
8              group(Subcontracting4PD)
9              {
10                 ShowCaption = false;
11                 Visible = Rec."Subcontracting Invoice";
12
                   0 references
13                 field(SubcontractNo4PD; Rec."Subcontract No.")
14                 {
15                     ApplicationArea = All;
16                     ToolTip = 'Specifies the value of the Subcontract No. field.',
                       Comment = 'de-DE=Nachunternehmervertrag Nr.||nl-NL=Hiermee wordt de
                       waarde van het veld Onderaannemerscontractnr. opgegeven.';
17                     Editable = (Rec.Status <> Rec.Status::Released) and (Rec.Status <>
                       Rec.Status::Closed);
18
                       0 references
19                     trigger OnAssistEdit()
20                     var
21                         SCContractMgt: Codeunit "SC Contract Mgt. 4PD";
22                     begin
23                         SCContractMgt.ShowSubcontractAndCreateOptionally(Rec);
24                         CurrPage.Update(false);
25                     end;
```

master* 10↓ 0↑ Launchpad ⊳: 0 ⊗ 0 ⚠ 0 No AL workspace folder is active. An AL workspace is active, if a contained AL project Spaces: 4 UTF-8 CRLF {} AL

Development language (C/)AL

- 4th generation programming language ("4GL") with a much more generative approach than 3GL (C#, Java, Pascal, ...)
  - Developer writes business code, engine generates backend (C#) / frontend (HTML, JavaScript etc) code
- Optimised for the development of application logic
- Very quick to learn, relatively easy to use
  - But like everywhere else, it takes talent and hard work to be above average...
- Pascal-like syntax
- Past C/AL, present AL

Objects in AL

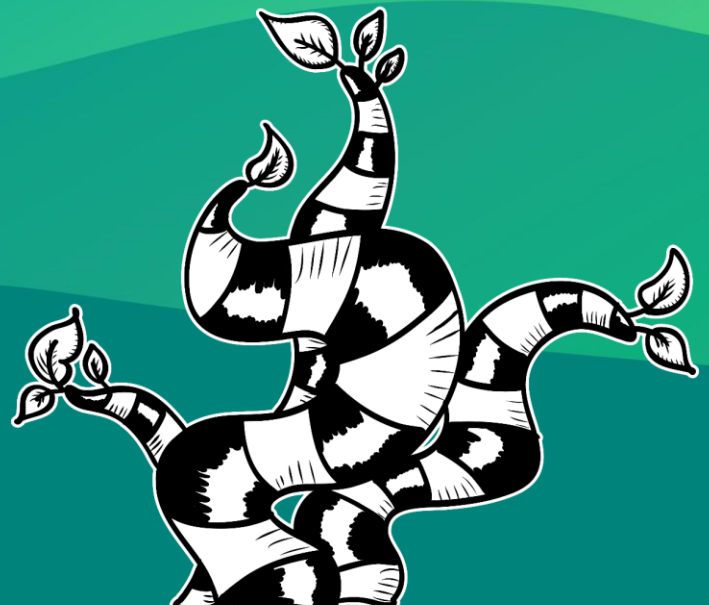
- Objects in Business Central != Object-orientation (OOP) in other languages
- Code and structures are stored in different object types:
  - Tables (correspond to tables on SQL Server)
  - Pages (UI for displaying data, among other things)
  - Codeunits ("containers" for business logic, roughly comparable to DLLs)
  - Others for different purposes (reports, XML ports, profiles, ...)

DYNAMICS MINDS

Demo: Development

Based on AL learning sample in official docs

Delivery in BC (and NAV)



DYNAMICS MINDS

General approach (past)

- Very manual process
  - Develop in your local database (not file based, no real version control!)
  - Extract the changed objects into a file (“.fob”)
  - Connect to the target environment
  - Import the .fob file
- Could be automated in pipelines, but very cumbersome and with ugly workarounds

DYNAMICS MINDS

General approach (present)

- File-based development, resulting in modular .app packages
- Build in build pipelines
  - AL-Go by Microsoft works in GitHub
  - ALOps works on Azure DevOps
  - COSMO Alpaca works on Azure DevOps and GitHub (and provides development environments)
- Release publicly available solutions to AppSource
- Release customer-individual solutions directly to customer environments
  - Also possible directly from VS Code, but you shouldn't…

DYNAMICS MINDS

General approach (present)

- Test automation as part of the code
  - Typically run in pipelines as well
- Relatively new feature: page scripting
  - Non-technical people record steps they take in the client
  - Results in YAML files describing the steps
  - Can be exported and imported into other environments
  - Can run in pipelines as well

DYNAMICS MINDS

Demo: Delivery

THANK YOU!