Consultingwerk
software architecture and development

Refactoring business logic from ABL TTY or GUI to PASOE

Mike Fechner
mike.fechner@consultingwerk.de

Mike Fechner

- Director, Lead Modernization Architect and Product Manager, Architect of the SmartComponent Library and WinKit
- Specialized on object-oriented design, software architecture, desktop user interfaces and web technologies
- 34 years of Progress experience (V5 … OE12)
- Active member of the OpenEdge community
- Frequent speaker at OpenEdge related conferences around the world

Consultingwerk Software Services Ltd.

- Independent IT consulting organization
- Focusing on **OpenEdge** and **related technology**
- Located in Cologne, Germany, subsidiaries in UK, USA and Romania
- Customers in Europe, North America, Australia and South Africa
- Vendor of developer tools and consulting services
- Specialized in GUI for .NET, Angular, OO, Software Architecture, Application Integration
- Experts in OpenEdge Application Modernization

Agenda

- **Modernization Process**
- Application Architecture
- Dealing with (GLOBAL) SHARED Variables
- Dealing with messages or prompts
- Proparse
- Record Locking

Modernization drivers

- The obvious: a new user interface
  - Web interface
  - Modern desktop UI
  - Mobile or satellite applications

- Functional requirements
  - Integration with 3rd party applications (in and out)
  - Localization
  - Hard to keep up with new features
  - Redundancy and spaghetti code killing agility and maintainability

Modernization drivers

- Improved code quality / maintainability
  - Improvements to application longevity
  - Component independency
  - Module independence
  - Method length
  - Test driven development to improve quality and agility
- Get ready for a new generation of software developers
  - Foreseeable retirement of key developers
  - Need to make application attractive to young developers
  - Enable application for distributed development

Other modernization drivers

- OpenEdge Version upgrades
  - WebSpeed retiring with OpenEdge 11.7 on April 1st 2025
  - Progress Dynamics not available in OpenEdge 12
  - Printing solutions, still anyone using Report Builder?

- AppServer enabling to enable performance
  - This is still a thing!
  - Customers still running large processing routines via Client/Server in Wifi

Modernization drivers

- Modernization drivers need agreement between all stake-holders
  - development team
  - business
- When time-pressure comes, goals not directly visible to end users may otherwise be sacrificed
  - code-optimization
  - adherence to architectural standards
  - test-driven-design
  - technical documentation

Know what will remain constant

- I expect that OpenEdge and PASOE will still be around 10 years from now
- I expect that OpenEdge will keep fundamentally backwards compatible with todays source-code
- Majority of application functionality should be moved to PASOE

- I will not even try to foresee the trends in user-interface technology in the next few years

Modernization Strategies

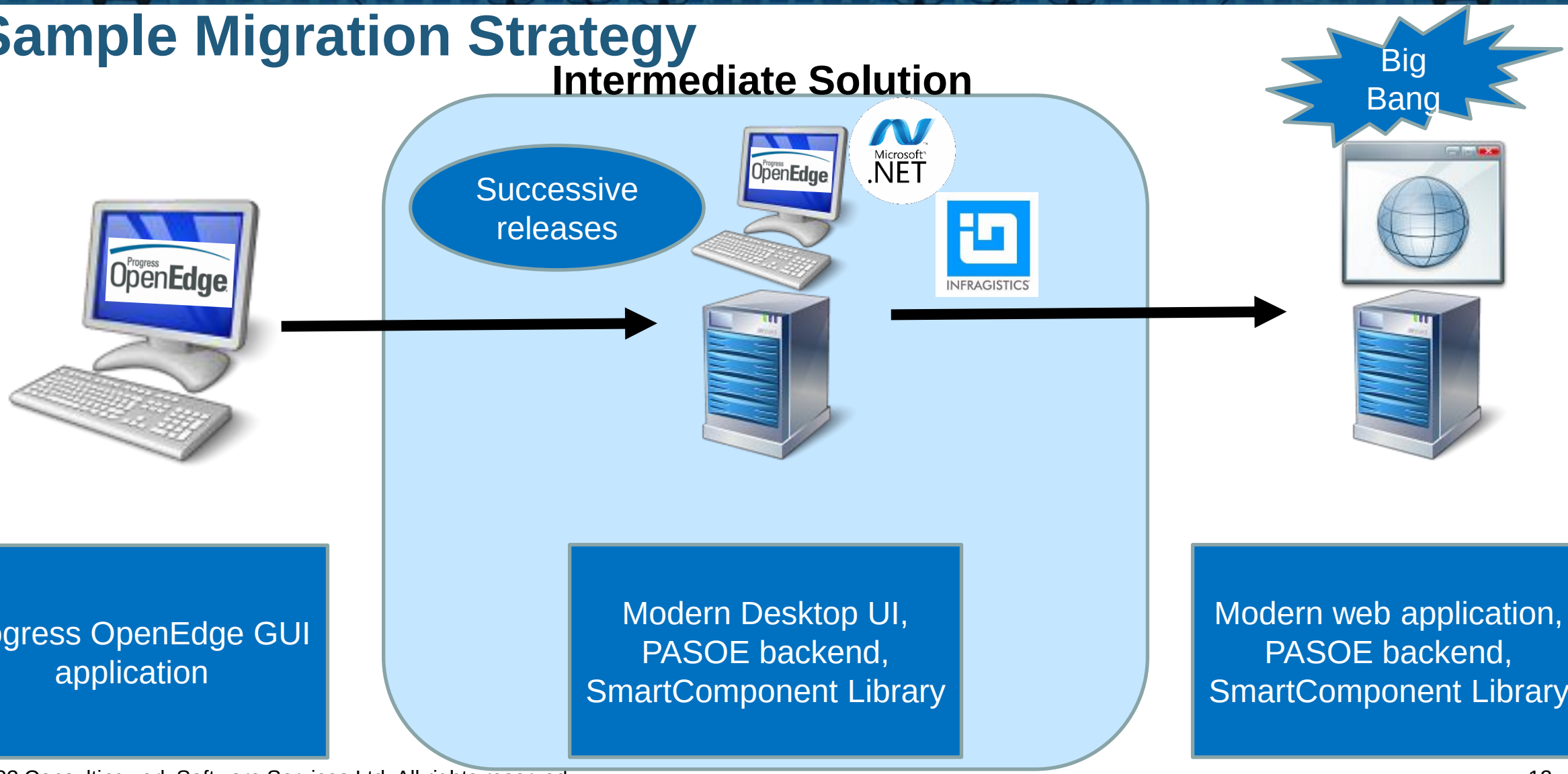
- Modernization of the whole application?
  - Going from ABL GUI to GUI for .NET or Web or Mobile
  - What is the *"final"* UI technology
  - GUI for .NET as an intermediate / integration with legacy GUI while the backend is rearchitected

- Or do we (first) add a few new features?
  - Mobile client for parts of the application
  - REST/REST(ful) interfaces for parts of the application
  - Reduce risk, gather first experience

Modernization Strategies

- Modernizing OpenEdge GUI (or TTY) to N-Tier first
  - Preparing the Application Backend for the Web
- Modernizing the whole application to the Web
  - Driven by demands from? Users, IT organization, marketing?
  - Definition of MVP – minimum viable product
  - How much functionality must be delivered on the web for user acceptance
- Developing one or multiple “satellite” web applications
  - Deliver quick and with reduced risk

Sample Migration Strategy



Quality of the application

- Are parts of the application reusable?
  - With no or little changes
  - Are major functional changes required?
  - Are major changes to the database structure required?

- Can parts of the application serve to describe the requirements
  - Legacy code review as part of the requirements definition
  - Is the existing source code the only (complete) description of the application functionality?

Development Team Skills

- New development process (agile)
- New tools (Progress Developer Studio, SCM, Unit Tests, DevOps, Docker, Frontend tools) and Frameworks
- New architecture: Distributed
- New development languages
  - OOABL
  - HTML, JavaScript, TypeScript, rapidly changing
  - Desktop technologies
  - Web and Mobile frameworks

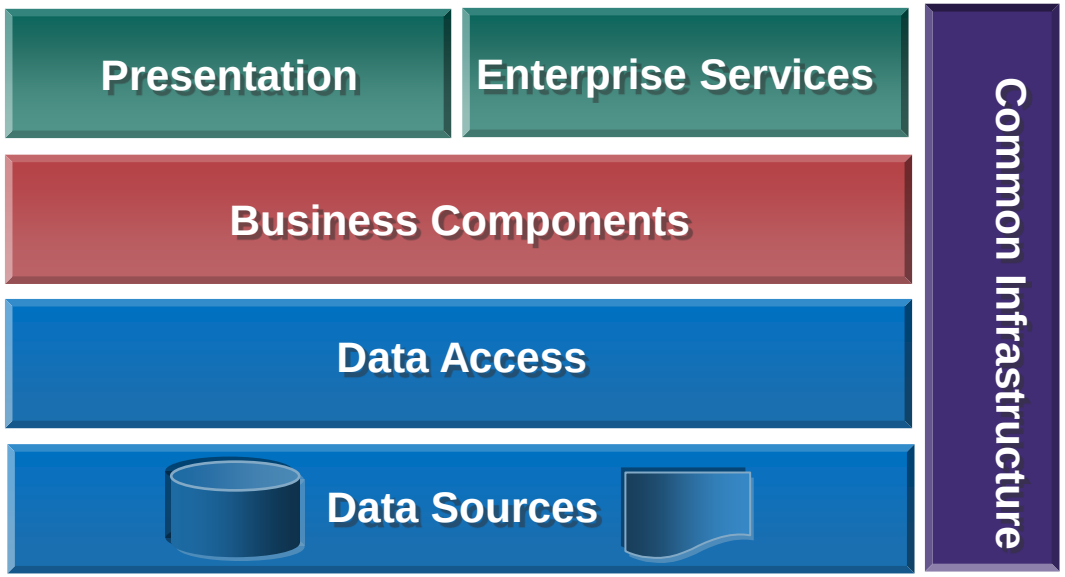
Agenda

- Modernization Process
- **Application Architecture**
- Dealing with (GLOBAL) SHARED Variables
- Dealing with messages or prompts
- Proparse
- Record Locking

The OpenEdge Reference Architecture (OERA)

The OpenEdge Reference Architecture (OERA) defines the general functional categories of components that comprise an application. It can be used as a high-level blueprint for developing OpenEdge service-oriented business applications.

Each layer of the OERA consists of distinct components, each with specific characteristics, roles and responsibilities. In addition, the OERA provides guidelines as to how each of the architectural components interacts. The following diagram illustrates the component architecture and the relationships between each of the components.



https://community.progress.com/s/question/0D54Q0000819wkqSAA/introduction-to-the-openedge-reference-architecture

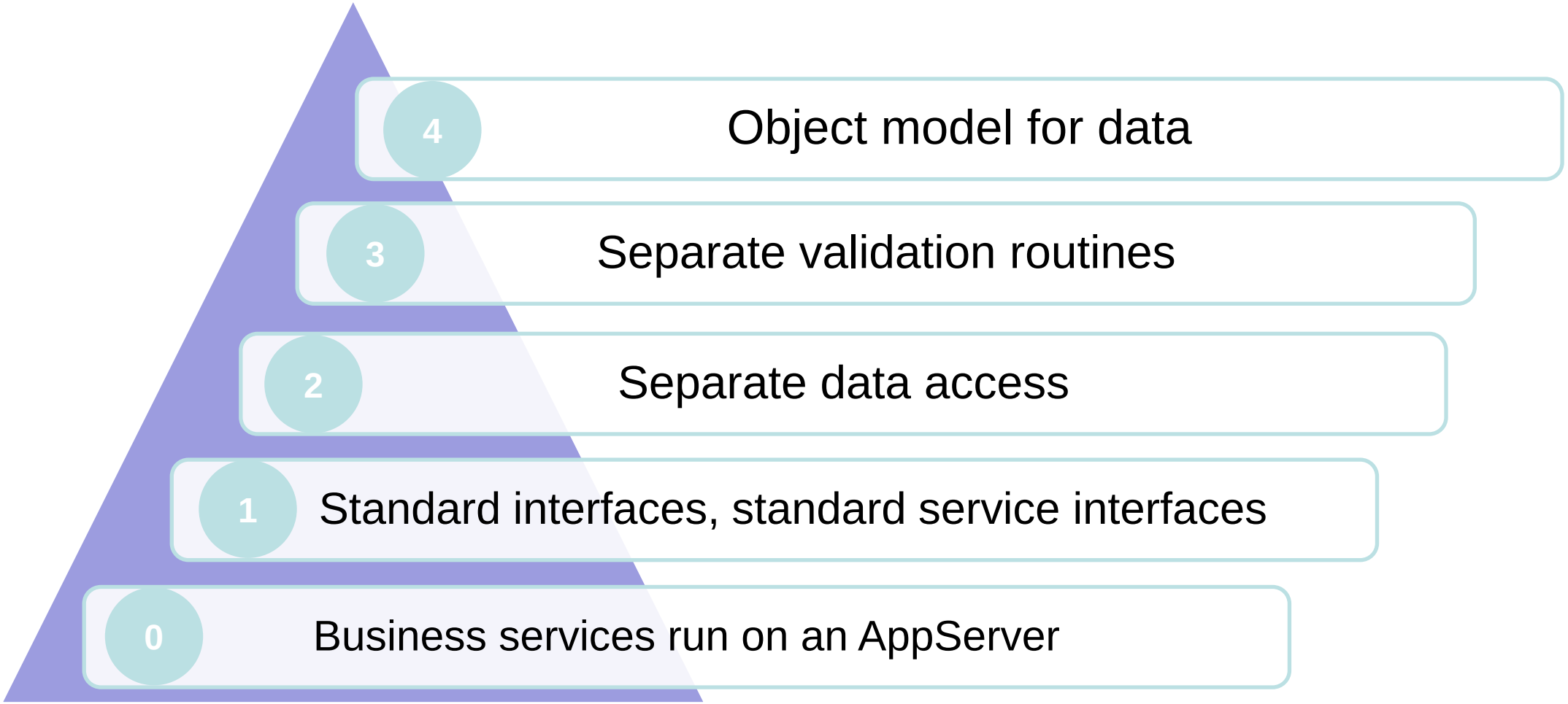
The OpenEdge Reference Architecture (OERA)

- Focus is on high-level architecture "blueprint"
- OERA is not prescriptive …
  - Choose to use procedural or OOABL code
  - Choose to implement some or all layers
  - Choose to keep existing code
- Service Interface Layer almost entirely ignored
- No guidance given on implementation, other than sample code

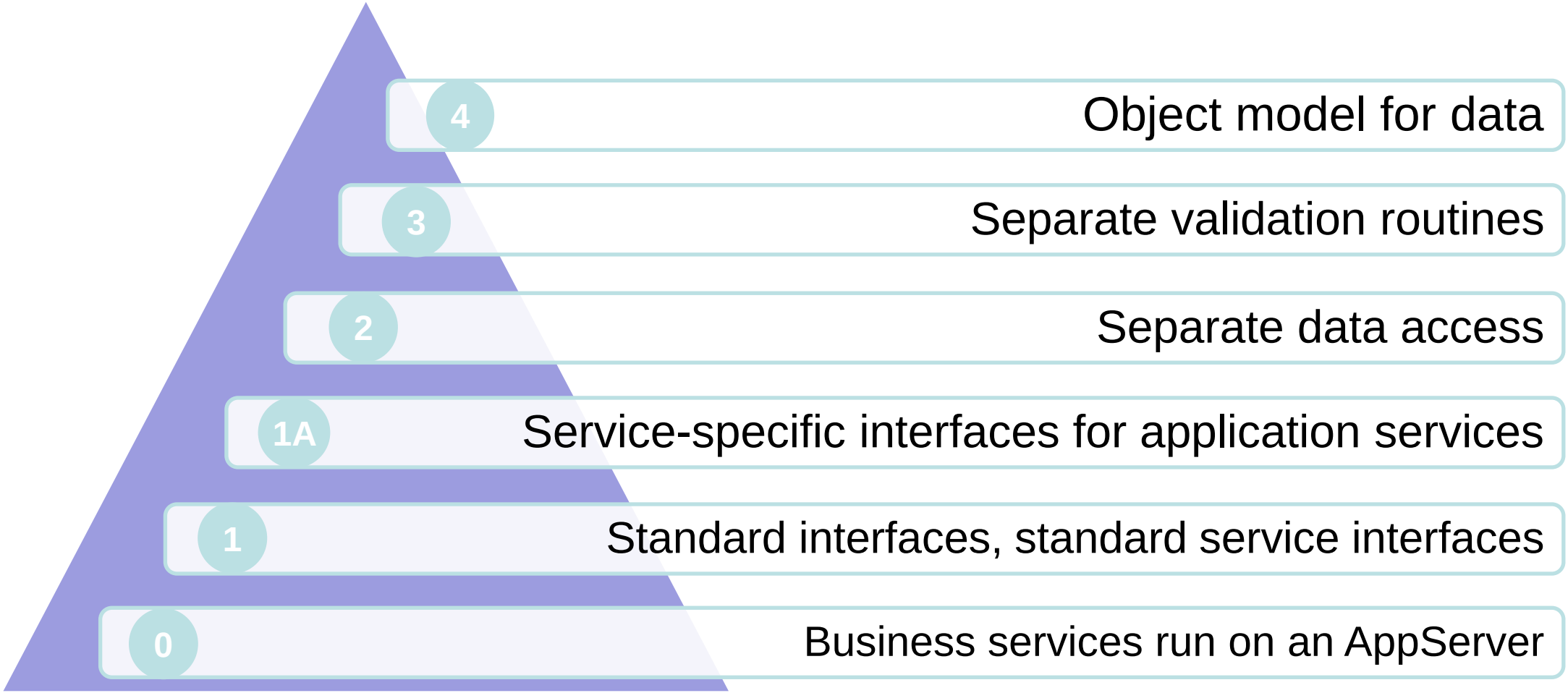
The Consultingwerk OERA Maturity Model

- An *opinionated* attempt to give application architects and developers some orientation in how to implement OERA compliant ABL applications
  - E.g. We don't believe a Data Source layer provides value
- Assumes that different developer teams have different requirements and expectations for the architecture and coding style of modernized ABL applications
- Builds upon the OERA and is focused on *implementation*, primarily relating to the Business Components and Data Access layers
- Soon on https://www.consultingwerk.com/news/blog

The Consultingwerk OERA Maturity Model



The Consultingwerk OERA Maturity Model



Consequences

- ✓ More, smaller programs
  - Programs better follow Single Responsibility Principle / Separation of Concerns
  - Unit testing made easier
- ✓ OOABL ***strongly*** recommended
- ✓ Pattern-based development
  - E.g. fetching data always looks the same, regardless of which business service's data is fetched
- Reuse of existing code typically reduced as a result

- Increasing developer productivity via
  - ✓ Compiler help with OOABL
  - ✓ Standardized development approaches
  - ✓ Reduced merge pain
  - ✓ Smaller programs, smaller impact from changes
  - ✓ Fewer "God programs"

Reduced reuse of legacy code?

- Implementing full separation of concerns can mean it's much harder to reuse existing blocks of code
  - Reusing large parts of existing code promises faster migration process
  - Existing unit- and system tests can continue to be used when reusing legacy code

- Find the balance between migration speed and risk reduction, and future-proofing and increased maintainability

Service Interface(s)

Bouncer

Babelfish

Service Interface(s)

- The Service Interface receives calls from clients or external consumers
- A very important and often under-appreciated component
  - The Service Interface is responsible for Validating the request (including Authentication and Authorization)
  - Ensuring the User-Session is in the correct state
  - Allocating the service (the Application Service, Business Task or Entity)
  - Converting the request data from an external format to internal
  - Converting the response data from internal format to external

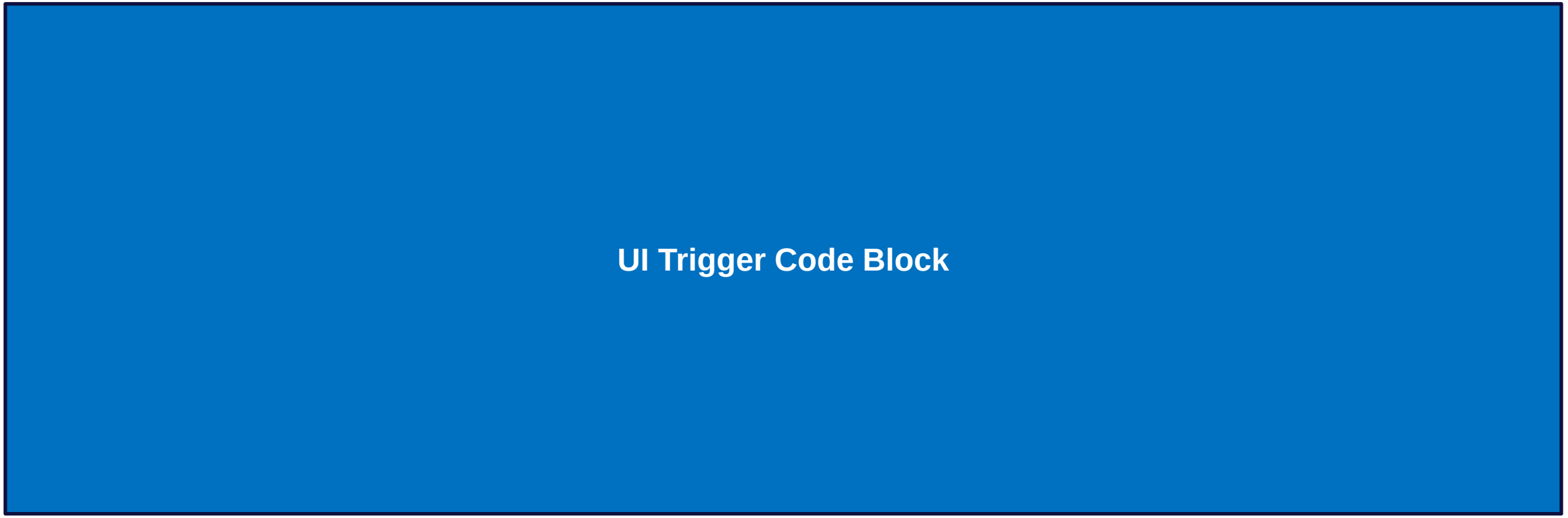
Service Interface(s)

- Business Logic is the most valuable piece of the application
- User interfaces come and go (TTY, ABL GUI, GUI for .NET, Web, Mobile, Chat, …)
- We do not want to rewrite – or even change the Business Logic for every new UI trend
- Multiple parallel used UI technologies should be using the same Business Logic
  - When there are very specific requirements for a single UI (e.g. Wizard style vs. plain data entry, consider using Application Service for this as an aggregate of multiple Business Tasks or Entities)

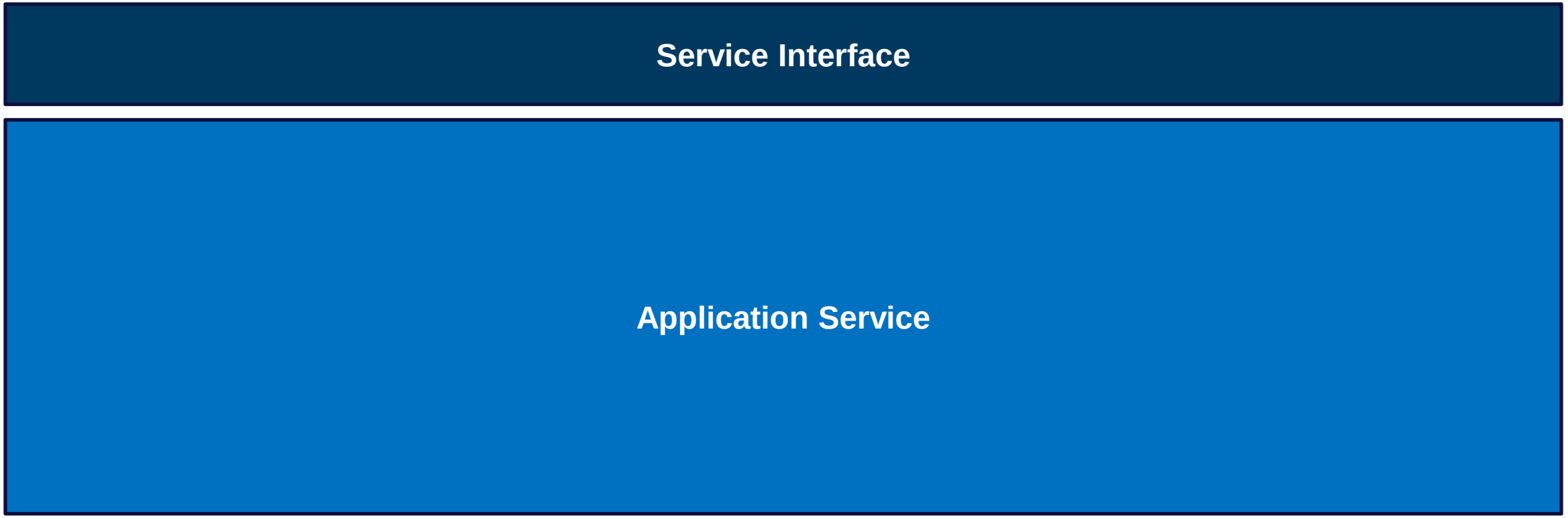
Top-down code-generalization

- Existing code considered to be closest to an application service
- First step is moving code from UI into an application service
- Simplifies automation during code-refactoring (almost statement by statement replaces)
- Further steps will improve code-reuse and single-concern by extracting code from application service into domain services
- Code de-duplication requires more design and guidelines
  - Into how many pieces do we cut the monolith?

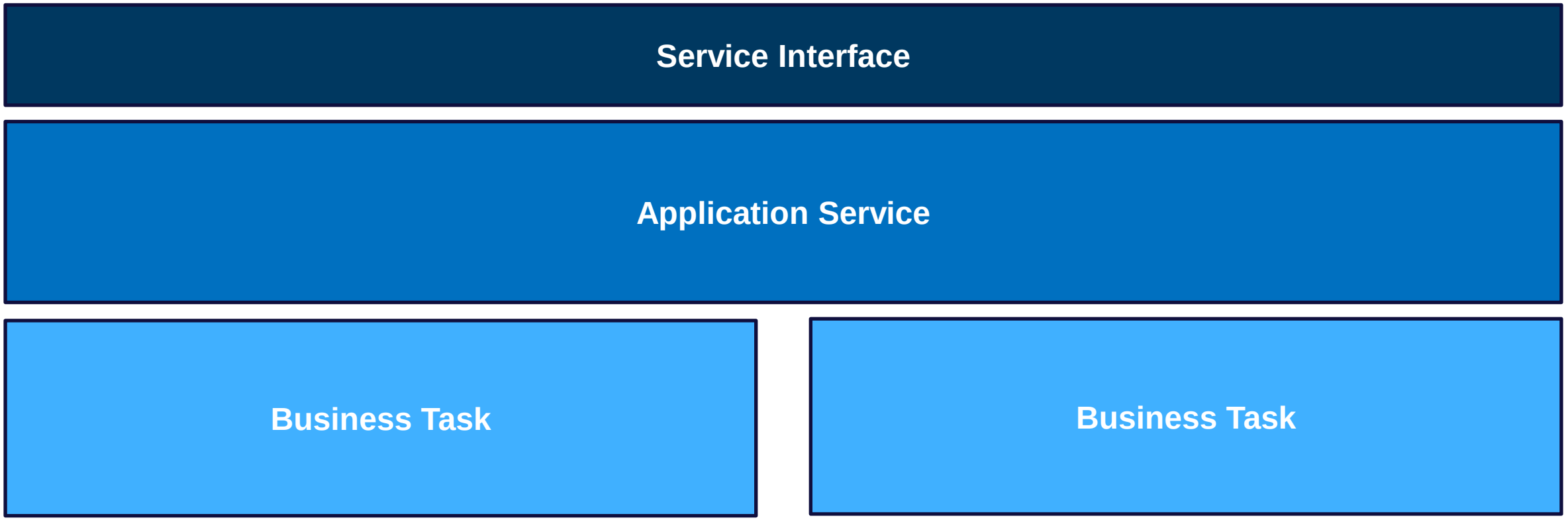
Top-down code-generalization



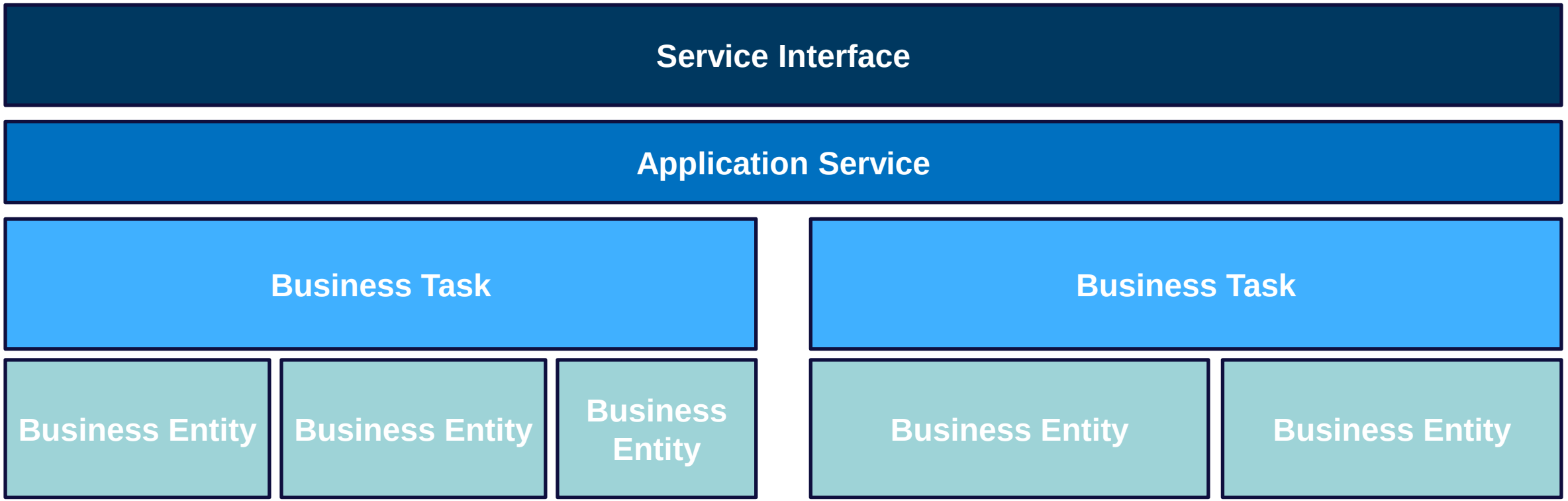
Top-down code-generalization



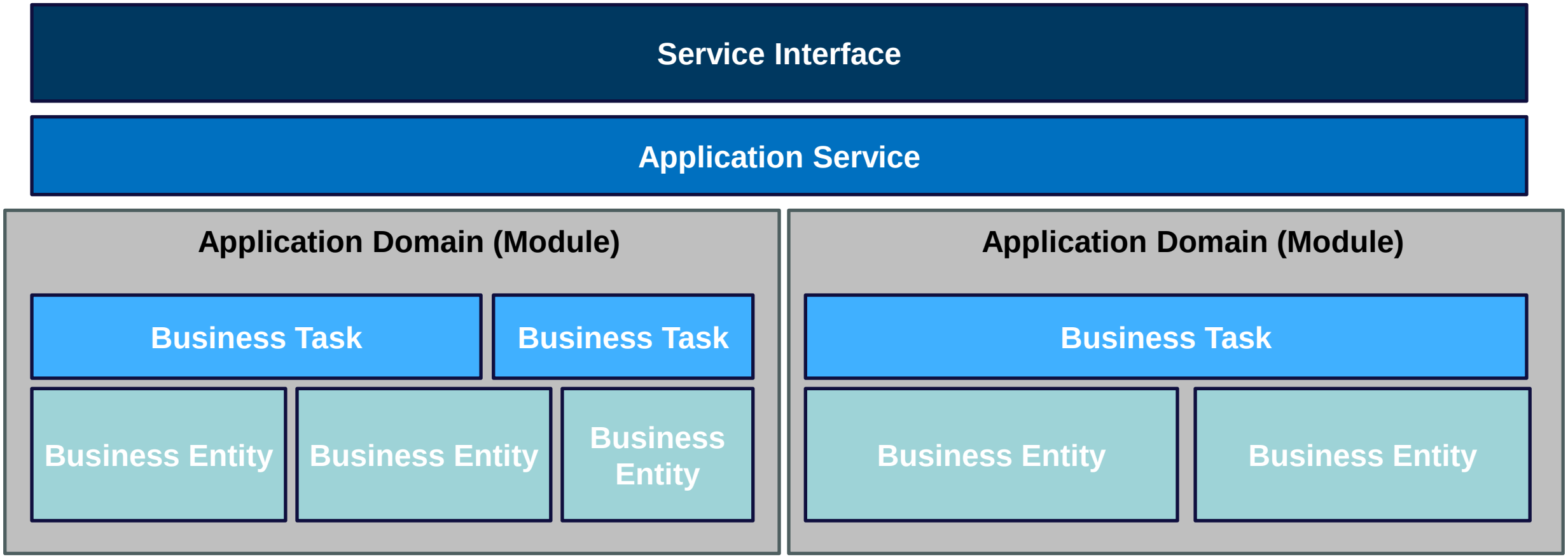
Top-down code-generalization



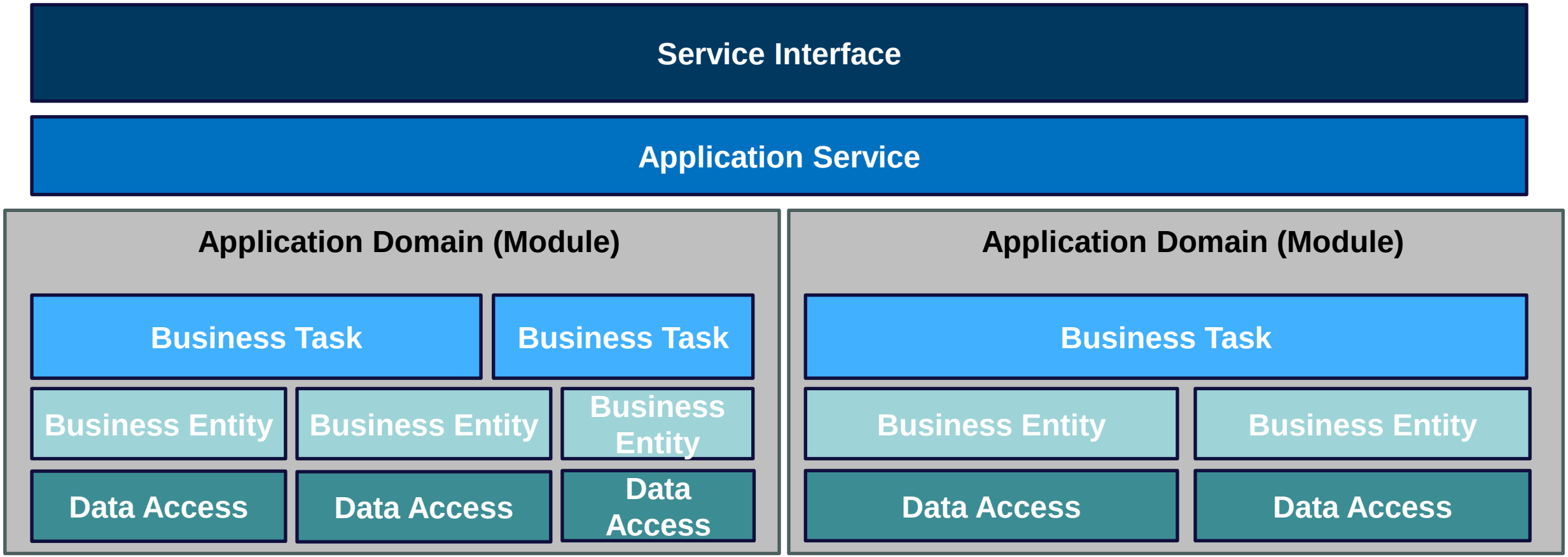
Top-down code-generalization



Top-down code-generalization



Top-down code-generalization



Demo

- Review different stages of the VALUE-CHANGED of OrderLine.ItemNum

c-win.w 9+

SmartComponentLibrary > Demo > Refactoring > SimpleTrigger > c-win.w > ...

Original GUI Trigger Block in .w File

```
228  ON LEAVE OF OrderLine.Itemnum IN FRAME DEFAULT-FRAME /* Item Num */
229  DO:
230      DEFINE VARIABLE iQty AS INTEGER NO-UNDO.
231
232      DEFINE BUFFER Item FOR Item.
233      DEFINE BUFFER Bin FOR Bin.
234
235      FIND Item WHERE Item.Itemnum = INPUT OrderLine.Itemnum
236          NO-LOCK NO-ERROR.
237
238      IF NOT AVAILABLE Item THEN DO:
239          MESSAGE "Item not available!" VIEW-AS ALERT-BOX ERROR .
240          RETURN NO-APPLY.
241      END.
242
243      DISPLAY Item.ItemName WITH FRAME {&FRAME-NAME} .
244
245      FOR EACH Bin WHERE Bin.Itemnum = INPUT OrderLine.Itemnum
246          NO-LOCK:
247          ASSIGN iQty = iQty + Bin.Qty.
248      END.
249
250      IF iQty < INPUT OrderLine.Qty THEN DO:
251          MESSAGE "Not enough items in stock!" VIEW-AS ALERT-BOX ERROR .
252          RETURN NO-APPLY.
253      END.
254  END.
```

AppServer Enabled Trigger Block in .w File

```
&ANALYZE-SUSPEND _UIB-CODE-BLOCK _CONTROL OrderLine.Itemnum C-Win
ON LEAVE OF OrderLine.Itemnum IN FRAME DEFAULT-FRAME /* Item Num */
DO:
    DEFINE VARIABLE iQty AS INTEGER NO-UNDO.

    DEFINE BUFFER ttItem FOR ttItem.
    DEFINE BUFFER ttBin FOR ttBin.

    // GET http://server:port/web/Items/42
    RUN get-item.p ON hAppServer (INPUT OrderLine.Itemnum) .
    FIND FIRST ttItem.

    IF NOT AVAILABLE ttItem THEN DO:
        MESSAGE "Item not available!" VIEW-AS ALERT-BOX ERROR .
        RETURN NO-APPLY.
    END.

    DISPLAY ttItem.ItemName @ Item.ItemName WITH FRAME {&FRAME-NAME} .

    // GET http://server:port/web/Bins?ItemNum=42
    RUN get-bin-for-item.p ON hAppServer (INPUT OrderLine.Itemnum) .

    FOR EACH ttBin NO-LOCK:
        ASSIGN iQty = iQty + Bin.Qty.
    END.

    IF iQty < INPUT OrderLine.Qty THEN DO:
        MESSAGE "Not enough items in stock!" VIEW-AS ALERT-BOX ERROR .
        RETURN NO-APPLY.
    END.
END.
```

Consultingwerk
software architecture and development

AppServer Enabled Trigger Block in .w File
Data-Access transparent through ORM Wrapper of the SmartComponent Library

```
234   ON LEAVE OF OrderLine.Itemnum IN FRAME DEFAULT-FRAME /* Item Num */
235   DO:
236       DEFINE VARIABLE iQty AS INTEGER NO-UNDO.
237
238       DEFINE VARIABLE oItem AS ItemDatasetModel NO-UNDO.
239
240       DEFINE BUFFER ttBin FOR ttBin.
241
242       // GET http://server:port/web/Items/42
243       oItem = NEW ItemDatasetModel (INPUT OrderLine.Itemnum) .
244
245       IF NOT oItem:Item:Available THEN DO:
246           MESSAGE "Item not available!" VIEW-AS ALERT-BOX ERROR .
247           RETURN NO-APPLY.
248       END.
249
250       DISPLAY oItem:Item:ItemName @ Item.ItemName WITH FRAME {&FRAME-NAME} .
251
252       // GET http://server:port/web/Bins?ItemNum=42
253       RUN get-bin-for-item.p ON hAppServer (INPUT OrderLine.Itemnum) .
254
255       FOR EACH ttBin NO-LOCK:
256           ASSIGN iQty = iQty + Bin.Qty.
257       END.
258
259       IF iQty < INPUT OrderLine.Qty THEN DO:
260           MESSAGE "Not enough items in stock!" VIEW-AS ALERT-BOX ERROR .
261           RETURN NO-APPLY.
262       END.
263   END.
```

Simple .p for AppServer with validation logic extracted from Trigger block

validate-item-num.p 9+

SmartComponentLibrary > Demo > Refactoring > SimpleTrigger > validate-item-num.p > ...

```
30
31  DEFINE INPUT  PARAMETER piItemNum  AS INTEGER   NO-UNDO.
32  DEFINE INPUT  PARAMETER piQty      AS INTEGER   NO-UNDO.
33  DEFINE OUTPUT PARAMETER pcItemName AS CHARACTER NO-UNDO.
34
35  DEFINE VARIABLE iQty AS INTEGER NO-UNDO.
36
37  FIND Item WHERE Item.Itemnum = piItemNum
38      NO-LOCK NO-ERROR.
39
40  IF NOT AVAILABLE Item THEN
41      UNDO, THROW NEW AppError ("Item not available!", 0).
42
43  ASSIGN pcItemName = Item.ItemName.
44
45  FOR EACH Bin WHERE Bin.Itemnum = piItemNum
46      NO-LOCK:
47      ASSIGN iQty = iQty + Bin.Qty.
48  END.
49
50  IF iQty < piQty THEN
51      UNDO, THROW NEW AppError ("Not enough items in stock!", 0).
52
```

OrderLinePresentationService.cls 8

SmartComponentLibrary > Demo > Refactoring > SimpleTrigger > OrderLinePresentationService.cls > Demo.Refactoring.SimpleTrigger.OrderLine

```
35  class Demo.Refactoring.SimpleTrigger.OrderLinePresentationService
40      method public UiControl ValidateItemNum (input-output dataset dsOrder,
41                                               poEventArgs as EventHandlerParameter):
42
43          define variable oUiControl as UiControl no-undo .
44          define variable iQty       as integer   no-undo.
45
46          oUiControl = new UiControl() .
47
48          find first eOrderLine.
49
50          find first Item where Item.Itemnum = eOrderLine.Itemnum
51              no-lock no-error.
52
53          if not available Item then do:
54              oUiControl:ValidationMessages:Add("eOrderLine", "ItemNum", "Item not available!").
55              oUiControl:Cancel = true.
56
57              return oUiControl.
58          end.
59
60          assign eOrderLine.ItemName = Item.ItemName.
61
62          for each Bin where Bin.Itemnum = eOrderLine.Itemnum
63              no-lock:
64              assign iQty = iQty + Bin.Qty.
65          end.
66
67          if iQty < eOrderLine.Qty then do:
68              oUiControl:ValidationMessages:Add("eOrderLine", "Qty", "Not enough items in stock!").
69              oUiControl:Cancel = true.
70              oUiControl:FocusFieldName = "Qty".
71          end.
72
73          return oUiControl.
74
75      end method.
```

Consultingwerk
software architecture and development

Application Service with generic server-side event handling approach of the SmartComponent Library

Demo

- Review different stages of migrating the „ship order“ trigger code
- Move out of GUI - into AppServer ready code
- Deal with questions using „MessageInteractionService“
- Throw Errors as Exception
- Introduce PreCheck method to reduce impact of undo’ing transactions on pending questions

w-orderentry.w 9+

SmartComponentLibrary > Demo > Refactoring > OrderEntry > w-orderentry.w > ...

```
202  ON CHOOSE OF MENU-ITEM m_Ship_Order /* Ship Order */
203  DO:
204      DEFINE VARIABLE lOk AS LOGICAL NO-UNDO.
205
206      DEFINE BUFFER OrderLine FOR OrderLine.
207      DEFINE BUFFER Item FOR Item.
208      DEFINE BUFFER InventoryTrans FOR InventoryTrans.
209
210      DO TRANSACTION ON ERROR UNDO, RETURN:
211          IF Order.OrderStatus <> "Ordered":U THEN DO:
212              MESSAGE "Order is not in the correct order status":T
213                  VIEW-AS ALERT-BOX ERROR.
214              RETURN "ERROR":U.
215          END.
216
217          IF Order.PromiseDate > TODAY + 7 THEN DO:
218              lOk = FALSE .
219
220              MESSAGE "Order is promised too far in the future. Are you sure you want to ship it now?":T
221                  VIEW-AS ALERT-BOX QUESTION BUTTONS YES-NO UPDATE lOk.
222
223              IF NOT lOk THEN
224                  RETURN "ERROR":U .
225          END.
226
```

```
227         FOR EACH OrderLine OF Order NO-LOCK:
228             FIND Item OF OrderLine NO-LOCK NO-ERROR .
229
230             IF NOT AVAILABLE Item THEN DO:
231                 MESSAGE "Item of order line" OrderLine.Linenum "is not available"
232                     VIEW-AS ALERT-BOX ERROR.
233                 RETURN.
234             END.
235
236             IF Item.Onhand < (OrderLine.Qty + Item.Allocated) THEN DO:
237                 lOk = FALSE .
238
239                 MESSAGE "Item" item.Itemnum item.Itemnum "is out of stock. Are you sure you want to continue?":T
240                     VIEW-AS ALERT-BOX QUESTION BUTTONS YES-NO UPDATE lOk.
241
242                 IF NOT lOk THEN
243                     RETURN "ERROR":U .
244             END.
245
246             FIND CURRENT OrderLine EXCLUSIVE-LOCK .
247
248             ASSIGN OrderLine.OrderLineStatus = "Shipped":U .
249
250             CREATE InventoryTrans.
251             ASSIGN InventoryTrans.Itemnum      = item.Itemnum
252                    InventoryTrans.Ordernum     = Order.Ordernum
253                    InventoryTrans.Qty          = OrderLine.Qty * -1
254                    InventoryTrans.TransDate    = TODAY
255                    InventoryTrans.WarehouseNum = iWarehouse.
256         END. /* for each OrderLine */
257
258         FIND CURRENT Order EXCLUSIVE-LOCK .
259
260         ASSIGN Order.OrderStatus = "Shipped":U
261                Order.ShipDate    = TODAY .
262
263         FIND CURRENT Order NO-LOCK .
264     END.
265 END.
```

OrderEntryApplicationService.cls 9+, E

Service.cls > Demo.Refactoring.OrderEntry.OrderEntryApplicationService > ShipOrder(↑INT piOrdernum, ↑INT piWarehouse, ↑Consultingwerk.Framework.MessageInteraction.ListQue

```
20  .ASS Demo.Refactoring.OrderEntry.OrderEntryApplicationService:
21
22    METHOD PUBLIC VOID ShipOrder (piOrdernum AS INTEGER,
23                                  piWarehouse AS INTEGER,
24                                  poQuestions AS ListQuestion):
25
26        DEFINE BUFFER OrderLine      FOR OrderLine.
27        DEFINE BUFFER Item           FOR Item.
28        DEFINE BUFFER InventoryTrans FOR InventoryTrans.
29
30        DEFINE VARIABLE oMessageService AS IMessageInteractionService NO-UNDO .
31        DEFINE VARIABLE oReply          AS MessageReplyEnum           NO-UNDO .
32
33        /* ASSIGN oMessageService = ... */
34
35        /* Fetch Database records - all buffers that are
36           expected to be available from outside this block*/
37        FIND Order WHERE Order.Ordernum = piOrdernum NO-LOCK NO-ERROR .
38        IF NOT AVAILABLE Order
39            THEN UNDO, THROW NEW Exception ("Order is not available":U, 0) .
40
41        DO TRANSACTION ON ERROR UNDO, RETURN:
42            IF Order.OrderStatus <> "Ordered":U THEN
43                UNDO, THROW NEW Exception ("Order is not in the correct order status":T, 0) .
44
45            IF Order.PromiseDate > TODAY + 7 THEN DO:
46                oReply = oMessageService:AskQuestion (poQuestions,
47                                                      "OrderEntryApplicationService.ShipOrder.IncorrectOrderStatus":U,
48                                                      ?,
49                                                      "Order is promised too far in the future. Are you sure you want to
50                                                      ?,
51                                                      ?,
52                                                      MessageTypeEnum:Question,
53                                                      MessageButtons:YesNo,
54                                                      MessageReplyEnum:ReplyNo).
```

```
        CASE oReply:
            WHEN MessageReplyEnum:Unanswered THEN
                RETURN .
            WHEN MessageReplyEnum:ReplyNo THEN
                UNDO, THROW NEW Exception ("The operation has been cancelled", 0) .
        END CASE .
    END.

    FOR EACH OrderLine OF Order NO-LOCK:
        FIND Item OF OrderLine NO-LOCK NO-ERROR .

        IF NOT AVAILABLE Item THEN
            UNDO, THROW NEW Exception (SUBSTITUTE ("Item of order line &1 is not available":T,
                                                   OrderLine.Linenum), 0) .

        IF Item.Onhand < (OrderLine.Qty + Item.Allocated) THEN DO:
            oReply = oMessageService:AskQuestion (poQuestions,
                                                  "OrderEntryApplicationService.ShipOrder.ItemOutOfStock":U,
                                                  BufferHelper:UniqueRecordIdentifier(BUFFER OrderLine:HANDLE),
                                                  SUBSTITUTE ("Item &1 &2 is out of stock. Are you sure you want
                                                               item.Itemnum,
                                                               item.ItemName),
                                                  ?,
                                                  ?,
                                                  MessageTypeEnum:Question,
                                                  MessageButtons:YesNo,
                                                  MessageReplyEnum:ReplyNo).

            CASE oReply:
                WHEN MessageReplyEnum:Unanswered THEN
                    RETURN .
                WHEN MessageReplyEnum:ReplyNo THEN
                    UNDO, THROW NEW Exception ("The operation has been cancelled":T, 0) .
            END CASE .
        END.
```

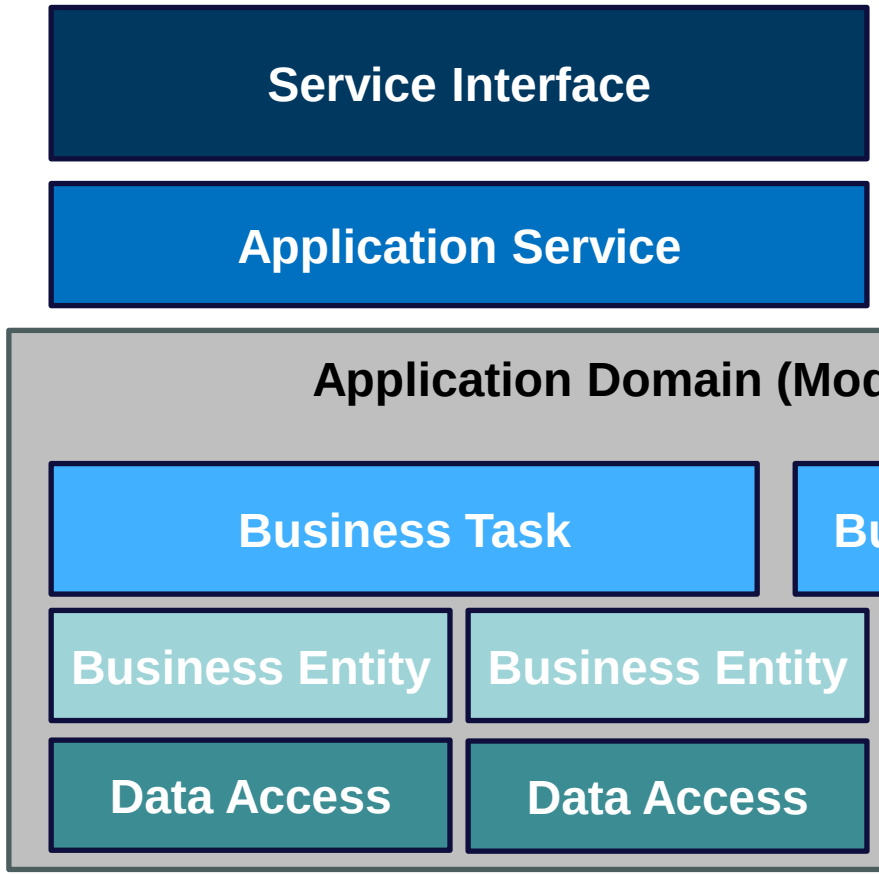
```
91
92              FIND CURRENT OrderLine EXCLUSIVE-LOCK .
93
94              ASSIGN OrderLine.OrderLineStatus = "Shipped":U .
95
96              CREATE InventoryTrans.
97              ASSIGN InventoryTrans.Itemnum      = item.Itemnum
98                     InventoryTrans.Ordernum     = Order.Ordernum
99                     InventoryTrans.Qty          = OrderLine.Qty * -1
100                    InventoryTrans.TransDate    = TODAY
101                    InventoryTrans.WarehouseNum = piWarehouse .
102         END. /* for each OrderLine */
103
104         FIND CURRENT Order EXCLUSIVE-LOCK .
105
106         ASSIGN Order.OrderStatus = "Shipped":U
107                Order.ShipDate    = TODAY .
108
109       FIND CURRENT Order NO-LOCK .
110     END.
111
112   END METHOD.
113
```

Benefits of top-down code-generalization

- First introduce service-ready component based on existing business logic
- Hide implementation details behind service interface
- Flow of business logic remains largely the same – **this will reduce risk**
- Component interface will allow
  - Use in modern user-interfaces
  - Implementation of unit-tests
- Unit tests will improve confidence when optimizing the code

Aspects of Top-Down code generalization

- Business Tasks and Business Entities should only deal with “their concern”
- Use factories or service managers – never directly new any application or domain business service object
- Only “allow” calls from top to bottom
- Services within a domain may call each other
- Services across domain boundary should use domain service interface



Further considerations

- Use parameter objects
- Separate screen-context from request parameters
- Selected warehouse may be screen context (might be a screen setting)
- Screen-context might be modified in UI and backend
- Selected order may be request-context (it’s the subject of ship order)

- Variables defined in the “definitions section” vs. parameters to internal procedures

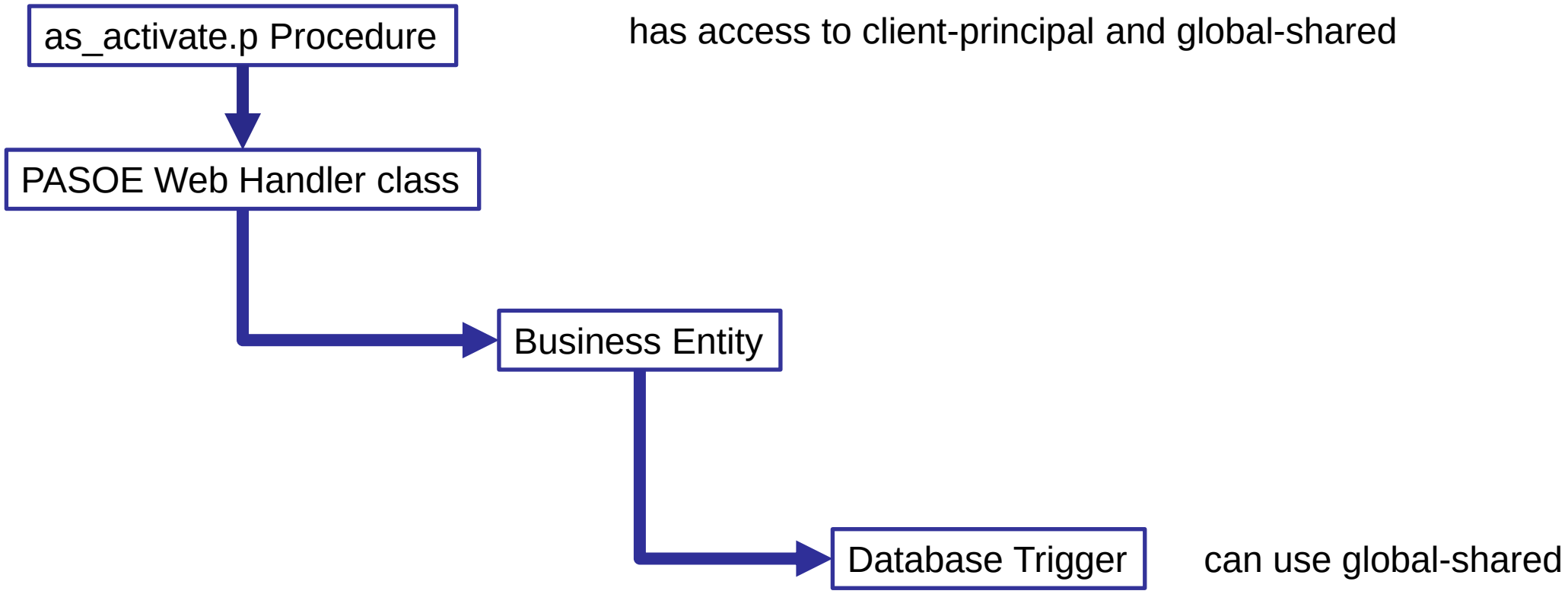
Agenda

- Modernization Process
- Application Architecture
- **Dealing with (GLOBAL) SHARED Variables**
- Dealing with messages or prompts
- Proparse
- Record Locking

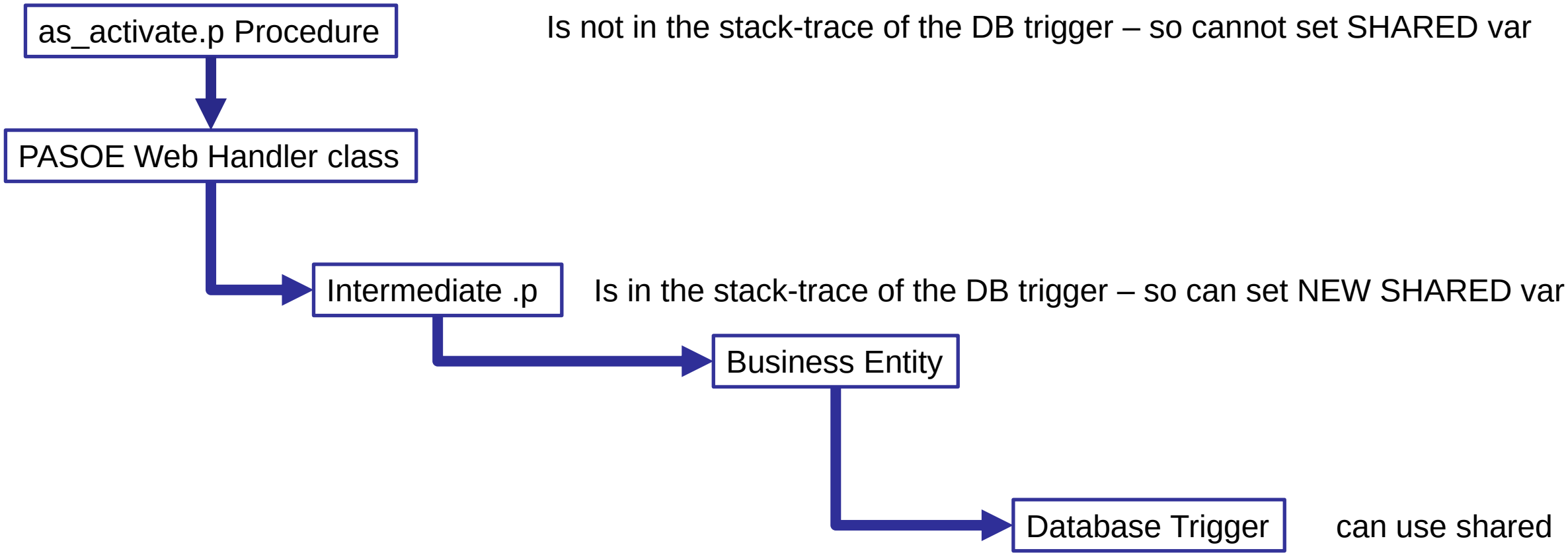
GLOBAL SHARED or SHARED variables …

- GLOBAL SHARED variables are less trouble
- SHARED variables should be reconsidered – many of them may be replaced with GLOBAL SHARED, usually a bad legacy
- Class based code (most new code, PASOE Web handlers) has NO access to any GLOBAL SHARED SHARED context

DB Trigger relying on a GLOBAL SHARED variable



DB Trigger relying on a SHARED variable



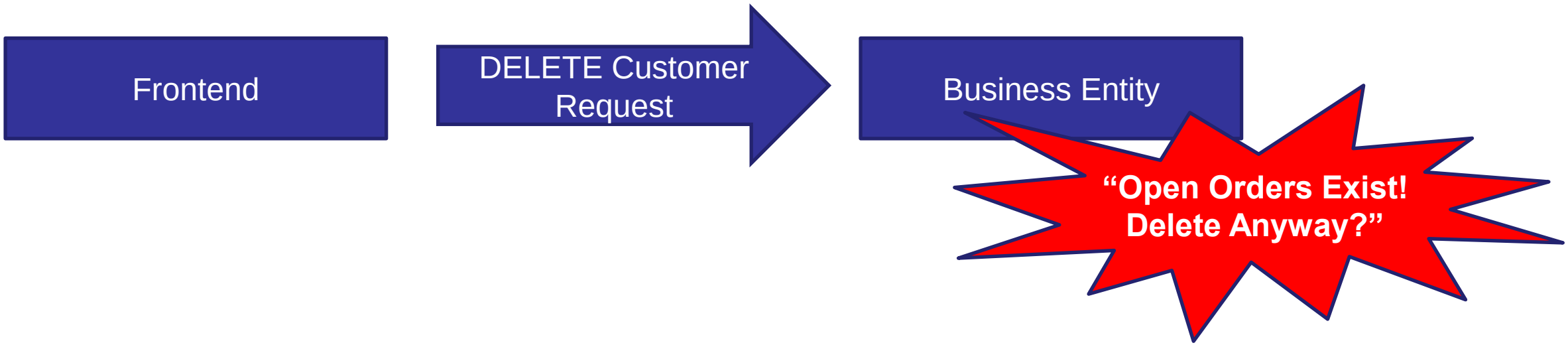
Agenda

- Modernization Process
- Application Architecture
- Dealing with (GLOBAL) SHARED Variables
- **Dealing with messages or prompts**
- Proparse
- Record Locking

Input blocking from the Backend

- Progress Application Server does not support Input Blocking on the UI
- Once AppServer is invoked, client waits for response
- Web technologies such as Socket.IO may be used to send messages from Backend to frontend
  - Back not vice-versa, no WAIT-FOR
- **When UI can foresee** that AppServer **may** require additional information when processing request, try adding this to the request
  - However UX should not be ignored. Too many irrelevant options confusion / annoying to users

Input Blocking, fat client ABL




```
IF CAN-FIND (FIRST Order OF Customer WHERE ....) THEN
    MESSAGE "Open Orders Exist! Delete Anyway?"
        VIEW-AS ALERT-BOX QUESTION BUTTONS YES-NO UPDATE response .
```

Input Blocking, fat client ABL


```
IF CAN-FIND (FIRST Order OF Customer WHERE ....)
   AND poRequest:DeleteOpenOrders = TRUE THEN …
```

Example challenge: Interaction between Back and

- Assumption: Existing Business Logic in large parts suitable as foundation for new application (functional and structural), especially validation
- Validation may also provide color coding to represent field status etc.
- Validation may have to prompt the user
- Web applications typically:
  Request (from browser) – Response (from server)
- No Input-Blocking (not possible to wait for user input in Business Logic)

Sample: Yes/No PROMPT in validation

- Demand is to keep the validation flow in major parts “as is“
- Validation may encounter question requiring user input: “Are you sure?” etc.

Sample: Yes/No PROMPT in validation

```
/* ---------- */
/* Verstorben */
/* ---------- */
if (date(Stamm.Todes_Dat:screen-value) <> ?) then do:
  /* Testen, ob Versicherter gerade eben verstorben ist. */
  if (EDIT_MODE = "UPDATE") then do:
    find Stamm no-lock where recid(Stamm) = MAIN_REC_ID.
    if (Stamm.Todes_Dat = ?) then do:
      /* Versicherter wurde soeben auf verstorben gesetzt. */
      run set_message_param(Stamm.Todes_Dat:screen-value).
      run user_warning("Der Versicherte ist am $1 verstorben. ~n~n" +
                       "Die zugehörigen Wohnadressen werden gesperrt.~n" +
                       "Überprüfen Sie, ob noch Revisionen vorgesehen sind~n" +
                       "und/oder Hilfsmittel zurückgenommen werden müssen.~n",
                       output continue).
      if not continue then return error.
    end.
  end.

end. /* if verstorben */
```

Sample: Yes/No PROMPT in validation

```
MSG = {Consultingwerk/get-service.i IMsg} .
SYS = {Consultingwerk/get-service.i ISys} .
MOD_ADD = {Consultingwerk/get-service.i IModAdd} .

if (eStammBefore.Todes_Dat = ?) then do:
    /* Versicherter wurde soeben auf verstorben gesetzt. */
    MSG:set_message_param(string (eStamm.Todes_Dat) /*:screen-value*/).

    continue = MSG:user_warning("Der Versicherte ist am $1 verstorben. ~n~n" +
                                "Die zugehörigen Wohnadressen werden gesperrt.~n" +
                                "Überprüfen Sie, ob noch Revisionen vorgesehen sind~n" +
                                "und/oder Hilfsmittel zurückgenommen werden müssen.~n",
                                this-object:GetClass():TypeName,
                                "eb09af84b1e2197b:4cb274e8:15608162bb6:-8000",
                                string (eStamm.SelfHdl)).

    if not continue then do:
        DatasetHelper:AddErrorString(buffer eStamm:handle, "_CANCEL") .
        return .
    end.

    /*if not continue then return error.*/
end.
```

Migration using MessageInteractionService API (SmartComponent Library framework)

- Backend – API maintains list of questions (unanswered and answered)
- Same API Call may ask a new question or return an existing answer
- Supports multiple questions per routine: Questions are flagged with e.g. a GUID identifying their location in code
- Support for multiple iterations (Loops, FOR EACH, …): Each question is also flagged with a records PUK value (GUID, combined key fields)

JSON Representation of the question

```
{
  "SerializedType": "Consultingwerk.Framework.MessageInteraction.Question",
  "MessageText": "Der Versicherte ist am 24\/12\/50 verstorben. \n\n
                  Die zugehörigen Wohnadressen werden gesperrt.\n
                  Überprüfen Sie, ob noch Revisionen vorgesehen sind\n
                  und\/oder Hilfsmittel zurückgenommen werden müssen.\n",
  "MessageButtons": "YesNo",
  "MessageReply": "Unanswered",
  "DefaultReply": "ReplyYes",
  "MessageID": "eb09af84b1e2197b:4cb274e8:15608162bb6:-8000",
  "MessageContext": "ac54bf82-56c4-bab2-2514-8e3d5c34775d"
}
```

Automation

- Migration of MESSAGE Statements into API calls can be automated using Proparse based tooling

Agenda

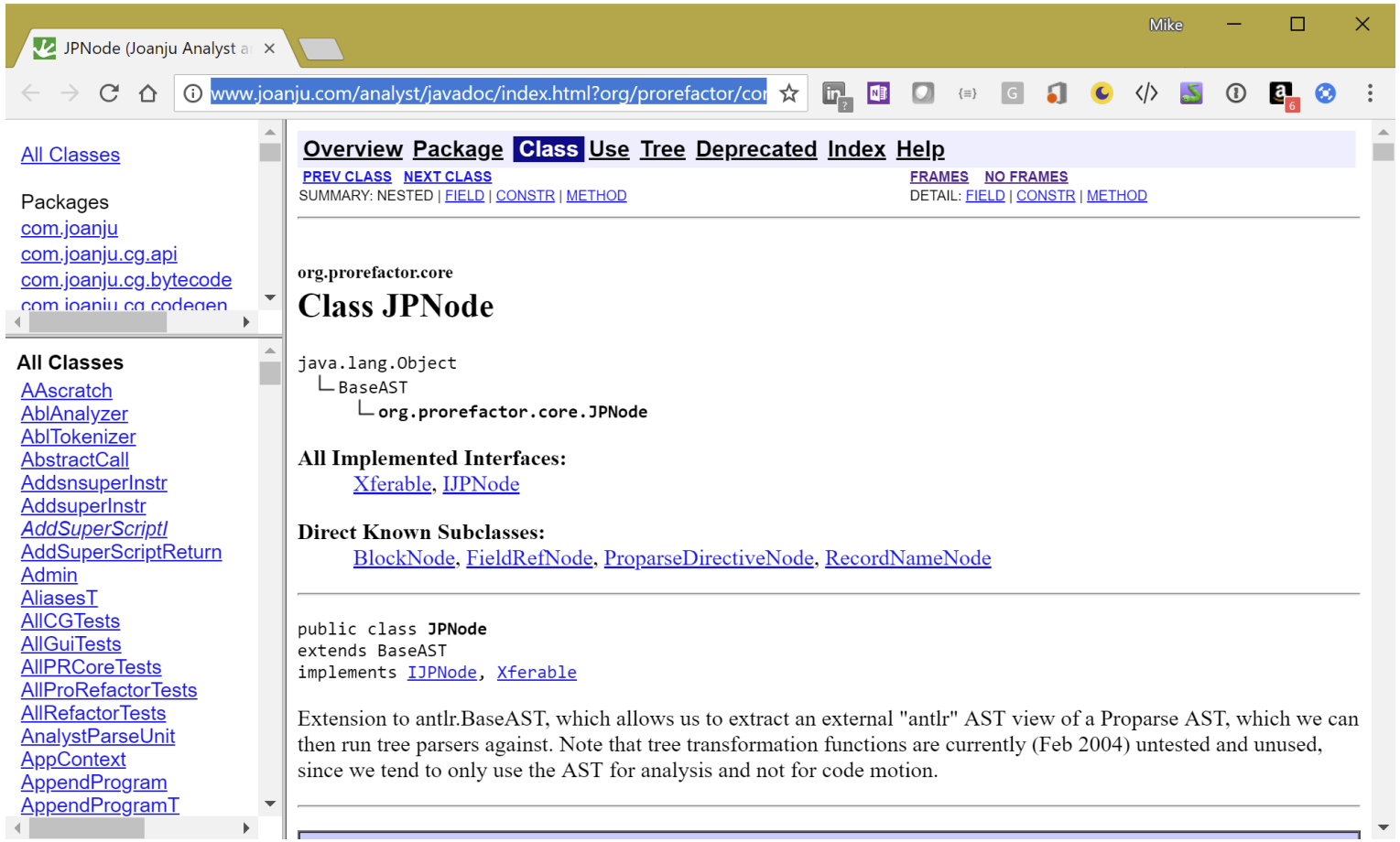
- Modernization Process
- Application Architecture
- Dealing with (GLOBAL) SHARED Variables
- Dealing with messages or prompts
- **Proparse**
- Record Locking

Source code parsing using Proparse

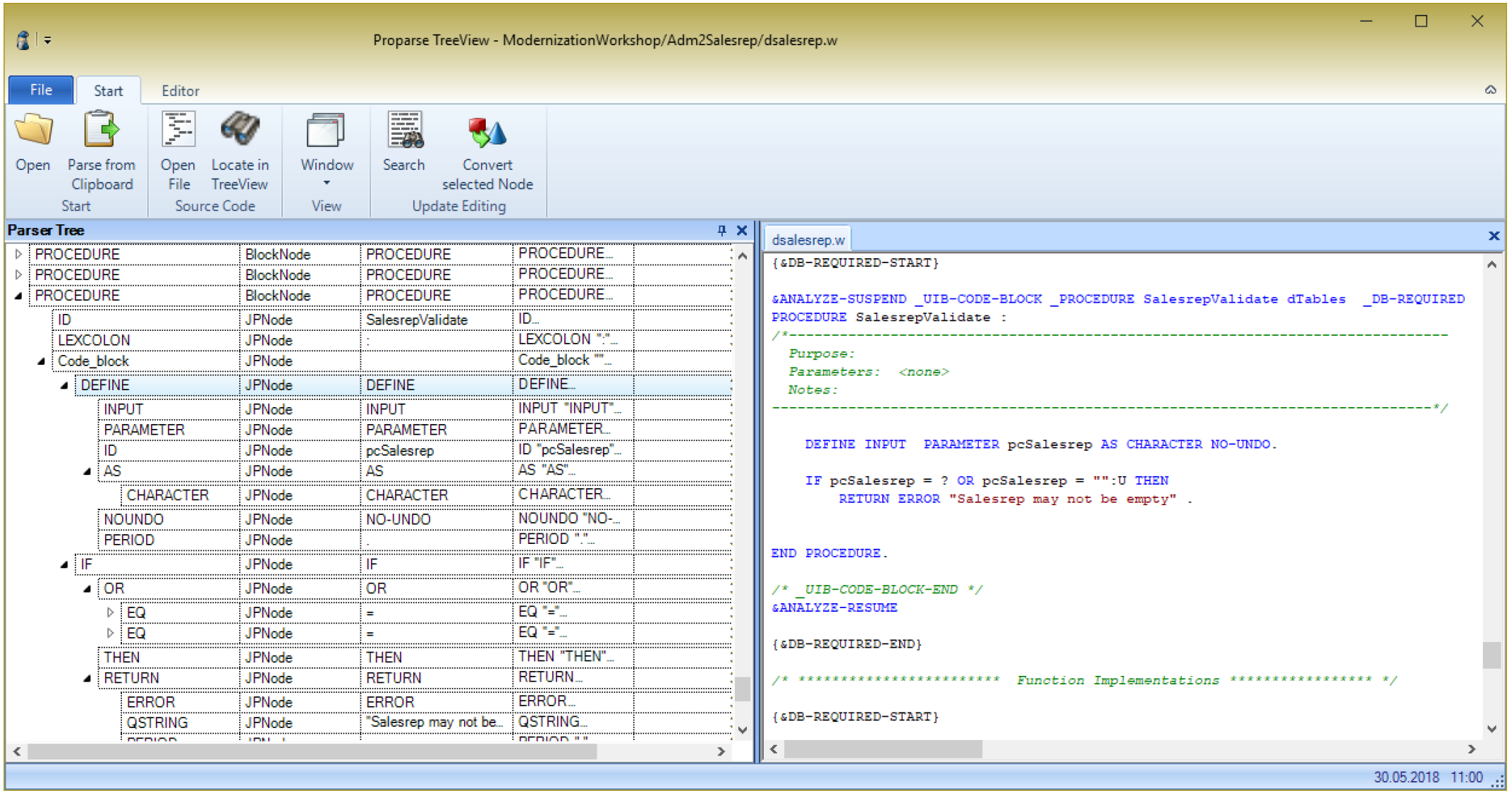
- ABL syntax parser, abstract view on ABL source code, based on ANTLR
- Eliminates the need for text based source code analysis
  - Resolves issues with line-breaks, abbreviated keywords, mixed order of keywords
- Open source
  - github.com/oehive/proparse
  - github.com/consultingwerk/proparse
  - github.com/riverside-software/proparse
- Actively maintained in various forks, support for 11.7 ABL syntax

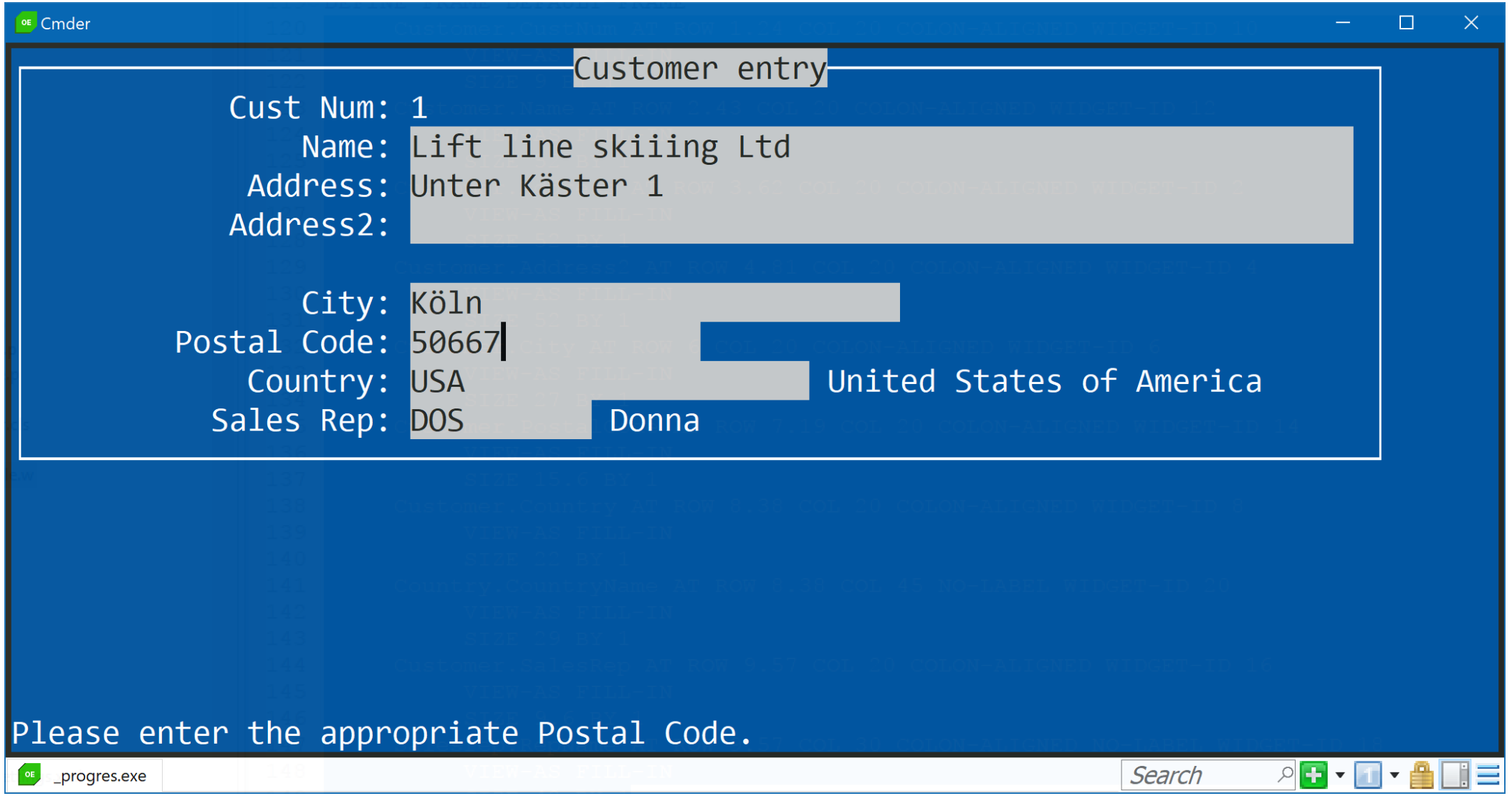
Proparse

- http://www.joanju.com/analyst/javadoc/index.html?org/prorefactor/core/JPNode.html



```
javafile = NEW java.io.File (pcFilename).

IF (NOT javafile:exists()) THEN
    UNDO, THROW NEW FileNotFoundException (pcFileName,
                                           SUBSTITUTE ("Could not find file: &1."{&TRAN}, pcFileName),
                                           0) .

IF cProparseCodepage > "" THEN DO:
    IF NOT Codepages:IsKnownCodepage (cProparseCodepage) THEN
        UNDO, THROW NEW InvalidValueException (cProparseCodepage, "ProparseCodepage":U) .

    oParseUnit = NEW ParseUnit(javafile, cProparseCodepage).
END.
ELSE
    oParseUnit = NEW ParseUnit(javafile).

pu:treeParser01().

DELETE OBJECT javafile .
```





UPDATE EDITING Blocks

```
DEFINE VARIABLE w-oldf AS CHARACTER NO-UNDO.

DO TRANSACTION:
    FIND CURRENT Customer EXCLUSIVE-LOCK .

    UPDATE {&ENABLED-FIELDS-IN-QUERY-DEFAULT-FRAME}
        WITH FRAME {&FRAME-NAME}
    blo-edit1:
    EDITING:

        READKEY.

        IF FRAME-FIELD <> "" THEN w-oldf = FRAME-FIELD.
        APPLY LASTKEY.

        IF FRAME-FIELD <> w-oldf OR GO-PENDING THEN
        DO:
            HIDE MESSAGE.

        /* ********** begin validation code ********** */
```

Single field validation within EDITING Block

```
IF w-oldf = "Salesrep" OR GO-PENDING THEN DO:

    FIND Salesrep WHERE Salesrep.SalesRep = INPUT Customer.SalesRep
        NO-LOCK NO-ERROR .

    IF NOT AVAILABLE Salesrep THEN DO:
        MESSAGE SUBSTITUTE ("Please enter a valid salesrep code. &1 is not a valid salesrep code.",
                            INPUT Customer.Salesrep) .
        NEXT-PROMPT Customer.Salesrep WITH FRAME {&frame-name}.
        NEXT blo-edit1.
    END.
    ELSE
        DISPLAY UPPER (Salesrep.SalesRep) @ Customer.SalesRep
                Salesrep.RepName WITH FRAME {&frame-name} .
END.
```

UPDATE EDITING Blocks

- Commonly used in TTY and early GUI applications
- Full of validation logic / Lookup functionality (locating foreign key descriptions)
- Tied to UI through “INPUT <fieldname>” references
- MESSAGE Statement used for error messages
- NEXT-PROMPT provides field that should receive input after error
- Record locked during duration of the UPDATE Statement

UPDATE EDITING Blocks

- Iterated for every keystroke or GO-PENDING
- When invoked on GO-PENDING, it's similar to a commit to a Business Entity
  - Validating all fields at once
  - Processing update when no validation error occurred
  - Returning validation error to user (with instruction of next field)
- Code flow in EDITING Block very similar to typical Business Entity validation

Business Entity Validation based on UPD EDITING

```
IF eCustomer.CustomerName = "" THEN DO:
     Consultingwerk.Util.DatasetHelper:AddErrorString (BUFFER eCustomer:HANDLE,
                                                       "Please enter customer name.",
                                                       "CustomerName":U) .
END.

FIND Salesrep WHERE Salesrep.SalesRep = eCustomer.SalesRep
    NO-LOCK NO-ERROR .

IF NOT AVAILABLE Salesrep THEN DO:
     Consultingwerk.Util.DatasetHelper:AddErrorString (BUFFER eCustomer:HANDLE,
                                                       SUBSTITUTE ("Please enter a valid salesrep code. &1 is
                                                       "SalesRep":U) .
END.
ELSE
    ASSIGN eCustomer.SalesRep = UPPER (Salesrep.SalesRep)
           eCustomer.RepName = Salesrep.RepName .

FIND Country WHERE Country.Country = eCustomer.Country
    NO-LOCK NO-ERROR .

IF NOT AVAILABLE Country THEN DO:
     Consultingwerk.Util.DatasetHelper:AddErrorString (BUFFER eCustomer:HANDLE,
                                                       "Please enter a valid country name",
                                                       "Country":U) .
END.
ELSE DO:
    ASSIGN eCustomer.Country = Country.Country .
    ASSIGN eCustomer.CountryName = Country.CountryName .
END .
```

Business Entity Validation based on UPD EDITING

- IF w-oldf OR GO-ENDING not required; Business Entity typically validates all fields at once
  - Removing at least one level of blocks in the code
- *“INPUT <fieldname>”* replaced with temp-table field reference
- *DISPLAY* statements replaces with update of temp-table field
- *MESSAGE/NEXT-PROMPT* statements replaced with API call to return validation message to the consumer of the Business Entity and control target field

Demo

- Proparse based migration of UPDATE EDITING Blocks into Business Entity Validation block

Agenda

- Modernization Process
- Application Architecture
- Dealing with (GLOBAL) SHARED Variables
- Dealing with messages or prompts
- Proparse
- **Record Locking**

Record Locking

- Record locking and Transaction concepts in the ABL within AppServer requests working as usual
- Legacy applications traditionally using pessimistic locking
- In an ABL fat-client with AppServer support scenario, ABL client and AppServer can lock records from each other …
- Different AppServer sessions serving the same client could lock records from each other …
- Minimum –lkwtmo of 10 seconds not ideal for AppServer requests

Record locking

- AppServer requests work better with optimistic locking – avoid record locking for long between AppServer request, detects update collisions when trying to update record (ProDataset before-image, time-stamp, etc.)

- Functional requirements may include record locking in distributed applications, e.g., ensure that Order header is not modified while updating or processing Order lines or related data
  - May be required to ensure record integrity
  - Item prices dependent on Terms in Order header

Soft record locks

- Alternative to record locks and transactions _can_ be soft locks
- Database table (e.g. SmartLock) with
  - Session Identifier
  - User Identifier
  - Resource Identifier
    - Database Table name and PUK values, “sports2000.Order” and “42”
    - Logical resource name: “month end processing 09/2024”
  - Lock time-out
- Time-out used to avoid eternal locks, alternative to back out locks on client disconnect

Soft lock API

- Acquire lock: Obtain record lock
  - Verify no other session is holding a lock record for the resource
  - Create lock record
  - Update existing lock record to refresh time-out
  - Return TRUE/FALSE or throw error
- Release lock:
  - Verify this session is holding the lock
  - Delete lock record
- Release all session locks:
  - Delete all lock records of a session on disconnect

Soft Lock Support

- Implement a scheduler job (e.g. all 15 minutes) to wipe out all expired lock records
- Consider implementing soft lock API also in legacy application as required to improve interoperability
- Consider simplified API for soft lock for legacy application, e.g. avoid need to introduce OO code there

Soft Lock Session Identifier

- For GUI/TTY Sessions, a GUID is suitable
- Authenticated /web requests receive a Session ID through the client-principal

Demo

- Review SmartLock API

```
/**
 * Purpose: Acquire logical application lock for a record
 * Notes:   This method is only implemented by SmartLockService
 *          Tries to create or update a SmartLock record
 * @param pcTableGuid Reference to unique record in SmartTable
 * @param pcKeyValues Values of a unique key field(s)
 * @param piLockDuration How long the lock may be hold in seconds
 * @param plThrowOnAlreadyLocked Logical indication to throw a record ... or not
 * @return True if lock is given else false
 */
METHOD PUBLIC LOGICAL AcquireLock (pcTableGuid AS CHARACTER,
                                   pcKeyValues AS CHARACTER,
                                   piLockDuration AS INTEGER,
                                   plThrowOnAlreadyLocked AS LOGICAL).
```

Questions

Consultingwerk

software architecture and development