"There's nothing as practical as good theory" (Kurt Lewin)

B-OSL - Object Oriented Specification Language for business - proposal (banking example)

OSL (Object Specification Language) contains business independent standard phrases, dedicated for definition any objects, and key-words applied in a given business-area and/or in global business environment (UNIVERSE).

B-OSL is a subset of OSL (Object Specification Language) dedicated to define business objects. Object is independent (self-contained) object having at least its own name, identifier, properties, behaviour and "history". Object class includes various types of objects (concrete and abstract, existent and non-existent, real and ideal, independent and dependent).
For banking typical objects are bank, headoffice, branch, customer, deposit-account, loan-accout, account manager, loan-credit line, product offered by bank, dealer...etc.

Description is related to the main object called SUBJECT (e.g. bank). UNIVERSE, AREA i SUBJECT could be treated as super-classes (at a top-hierarchy). Business classes within AREA are user-defined, to reflect SUBJECT behaviour and needs. Specification contains also executive/operational objects, like account manager, teller etc. and technical infrastructure objects (like servers, databases).

OSL NOTATION

<!...> <! comment >
<> <! container for main structural phrases: def, spec, commentaries. Object names and predefined key-words are written at Capital Letters. Non-business objects are written in Small Letters.>
=> <link - *to another system notation e.g. computer programs* >
<**def** ***object name*** > <**!** start of object definition >
</**def**> <**!** end of object definition >
<**spec** ***specification name*** > <**!** specification start >
</**spec** > <**!** specification end >
::= <**!** type assignment >
:= <**!** list assignment >
= <**!** value assignment >
: <**!** attribute assignment>
:: <**!** belongs to>
== <**!** equivalency >

{.....} **<!** { } delimiters >
(x,y,..) **<!** list >
YYYYYY.UUUUU(XXXXXX) **<!** qualified name of business object >
[name] **<!** executive/operational object >
(XXXX) **<!** business object >
keywords **<!** key-words are heritated by objects of lower class (rank) >

Environment (ENV)
ENV:=(REGULATIONS, INFRASTRUCTURE)
ENV.REGULATIONS:=(*Legal acts, resolutions, decisions ... etc.***)**
ENV.INFRASTRUCTURE:=(***Servers,OperationalSystems, TransactionalDataBases, DataWarehouses,NetworkManagementSystem, OrganizationalStructureOfCompany)***
Object specifications includes elements:
-id <! object identifier>
-**infoWindow** <! "window for communication with other object " >
-dataTable <! data table. >
-typ obiektu ObjectTypes : (eOBJECT<! elementary object e.g.a given customer >
- (ROLE)
- (OLH -Object Life History)
- (RELATIONS)
-dynamic characteristics(PROCESSses, transactions. events, mode of triggering etc.)
-mode of execution (EXECUTION_MODE:= modeRT <! real time >**,modeBAT<!** batch > **modeOND <!** on demand >**)**
- (DATA_FLOW)
- (CONTROL_FLOW)
Other OSL phrases:
trigger <! trigger (of PROCESSs or event) >**,**
generator <! generator of events (e.g. cashflow) >**,**
agent <! service object e.g. . agent of call-center >**,**
integrator <! integrator of complex object or set of objectsgt;
monitor <! tracer of PROCESSs >**,**
executor <! >**,**
participator <! participating object >,
owner, stockholder, customer, supplier; partner, employee,
component <! part of >
performance center <! > **)**
RELATIONS:=(*(activated by / activates, activated when/if,***

appearence depends on **<!** e.g. appearence of child object depends on existence of parent object >
assisted by, belongs to /is owned by , built from ,
calls <obiekt> (xxx,yyy) <! xxx input elements, yyy return elements >
consists of **<parts>** ***, contained in/contains, controlled by / controls,derived from,***
driven by transaction,driven by product ,driven by banking regulations,
driven by customer, driven by date, driven by schedule,driven by frequency
existence depends on **<!** condition of existance .... >
exists when/in/for, evaluated as critical/most wanted , included in ,
linked to ...by / links, matched/matches *<! e.g. transaction confirmation >*
refers to *<! direct relation > relates to, related by affinity, represented by /*
represents , involved in, shared by / shares, used by / uses)
STATE := (active,inactive,dormant,suspended,aborted,idle, lost)
STATUS := (generic, real, virtual)
dOBJECT<! dynamic object e.g. transaction >
iOBJECT<! informational object e.g. customer position >
vOBJECT < virtual object e.g. mirror Nostro accounts >**.**
dOBJECT ::= (EVENT,TRANSACTION,ACTION,PROCESSS)
<! EVENT -elementary atomic fact >
<! TRANSACTION - sequence of events aimed at shortterm goal e.g. account opening or closing, payment
>
<! OPERATION - e.g. monthly charge for account maintenance >
<! ACTION - complex activity e.g. defining provision for doubtful loans >
<! PROCESSS - sequence of actions and events with common aim, initiated by trigger e.g. end of day
PROCESSsing >
{CONTROL_FLOW
repetition :=(**iteration, single spiral, multiband spiral** <! spiral pattern is used in learning PROCESSses and
differs from iteration that each scroll could contains different mechanism and contents >)
activated .... BY ...with <initial-value> AT <time-point > WHEN ..
finished AT < > with <value> WHEN ...}
{ DATA FLOW
<.> from (<.>) <! data list... >
<.> to (<.>) <! data list >
<> dataGeneration Method <*name* >
<>dataEncoding Method <*name* >
<>dataCompressing Method <*name* >
}
{BODY ::= (Contents, Script, Metadata)

contents <! e.g. document contents, program code >
script <! operation list generated in object upon the pattern of its behaviour >
metadata <! e.g. in XML- body structure description >**}**

Simple example of B-OSL (OSL for banking) *(for illustration purpose only)*

```
<B-OSL-version 1.1>
<spec(BANKING)>

<def SUBJECT(MY_BANK)>
infoWindow: =(BankId , BusinessType <! uniwersal, commercial, retail> ,
country, bCurrency, FinancialYear, number of branches, RankingPosition, CurrencyLimits)
dataTables:= (CorrespondentBanks, Branches, Calendar-WorkingDays, bkAccountChart>
<! bkAccountChart - bookkeeping accounts chart>
<! bCurrency - base currency > </def>

<def UNIVERSE (INTERNATIONAL_BANKING)>
id :=BIC
keywords:= (IBAN<! International Bank Account Number > , ICC <! International Chamber of
Commerce > , BIC<! Bank Identification Code >
dataTables:= (LIBOR, OperatingCurrences >,
</def>

<def AREA(Banking)>
BUSINESS_MODULES::=(DEPOSITS, LOANS, CIF-INFO, MONEY-MARKET, PAYMENTS, DERIVATES,
SHARES)
keywords:=( BranchNr, CustomerId, AccountNr, Rate, Balance, tunParameter{<! tuning parameter >
)
dataTables := (Product)
procedures:= (NaliczOds, ...)
interest-rate refers to <interest rate table id >
OperationalObjects ::= [teller, AccountMgr, CustomerMgr, ProductMgr, trader]
{<! Main object classes >
CLASS ::= ( SUBJECT, PRODUCT, CURRENCY, LIMIT, ACCOUNT,
TRANSACTION,OPERATION,PROCESS,EVENT)
<! objects like CURRENCY, LIMIT, ACCOUNT-KS,TRANSACTION,OPERATION,PROCESS,EVENT are
included into specification of other objects ex..PRODUKT and themselves are subjects of object
definition>
PRODUCT ::= (ACCOUNT, DERIVATIVES, )
ACCOUNT::= (CURRENT, DEPOSIT, LOAN)
LIMIT ::= (COUNTRY, INDUSTRY, CUSTOMER, CURRENCY)}
{<! types of objects>
eOBJECT ::= (CUSTOMER,BANK,ACCOUNT,CURRENCY, DERIVATIVES) <! elementary objects>
iOBJECT::= (CUSTOMER-POSITION, MONTHLY-BALANCESHEET) <! information objects>
}
{ <! Object classification>
SUBJECT : (coSUBJECT <! company> , prSUBJECT <! person >)
BANK: (dmBANK <! domestic bank <! , frBANK <! foreign bank > , corBANK <! correspondent >)
ACCOUNT: (bsACCOUNT <! balance-sheetACCOUNT> ,nbsACCOUNT <! nonbalance-sheetACCOUNT >
TRANSACTION:
(rtTRANSACTION <! realtime TRANSACTION >,

eodOPERATION <! OPERATION generated at end of day procedure>)}
eomOPERATION <! OPERATION generated at end of month>)}
eoyOPERATION <! OPERATION generated at end of year>)}
eoppOPERATION <! OPERATION generated at end of product period>)}
EVENT: ( in-ev <! trigger> , end-ev <! ending> , pn-ev <! pending > , dl-ev <! deleting >)} rv-ev <!
reversing>
</def>

<def ACCOUNT>
infoWindow :=(owner, co-owner
MinBalance,actualBalance,HistoryStatement)
Relates to CUSTOMERid
rtTRANSACTION := (Open, Quit, Cash-in, Cash-out, transfer)
eomTRANSACTION:= (printStatement)
tunParameter := (interest-rate, period, ...) </def>

<def ACCOUNT.CURRENT> id::= (AccountNr)
exists for prSUBJECT <! for private persons >
initiated by FirstPayment <! cash-in, transfer >
ACCOUNT.CURRENT::infoWindow :=(DebetLimit , repPAYMENTS, StatementType, VisaCardId)

calls ArchEvent(5y,eoy) <! EVENT of deleting from archive TRANSACTION older than od 5 years >)
</def>
</def AREA>
</spec>
```

Concluding remarks

B-OSL specification is a descriptive schema with links to information technology. It could be a good base for comparative analysis of banking solutions (if all of them are specified in the same notation). Going further, if the implementation of this language is equipped with computer generator of tables and graphs, the "image" of business object is more readable.