Putting it all together: experiences with stored procedures, triggers, and XML on DB2 v8 for z/OS

Peter Vanroose

ABIS Training & Consulting

Nationale GSE-conferentie “The Next Step”

Zeist, 29 Oktober 2008

Goal

- describe our experiences with
  - setting up a complex end-to-end application
  - modular, service-oriented architecture
  - XML as data interface
  - DB2 stored procedure as API
  - history tables at the database end
  - using DB2 triggers to maintain history
- choices made & possible alternatives
- learn from our mistakes

Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

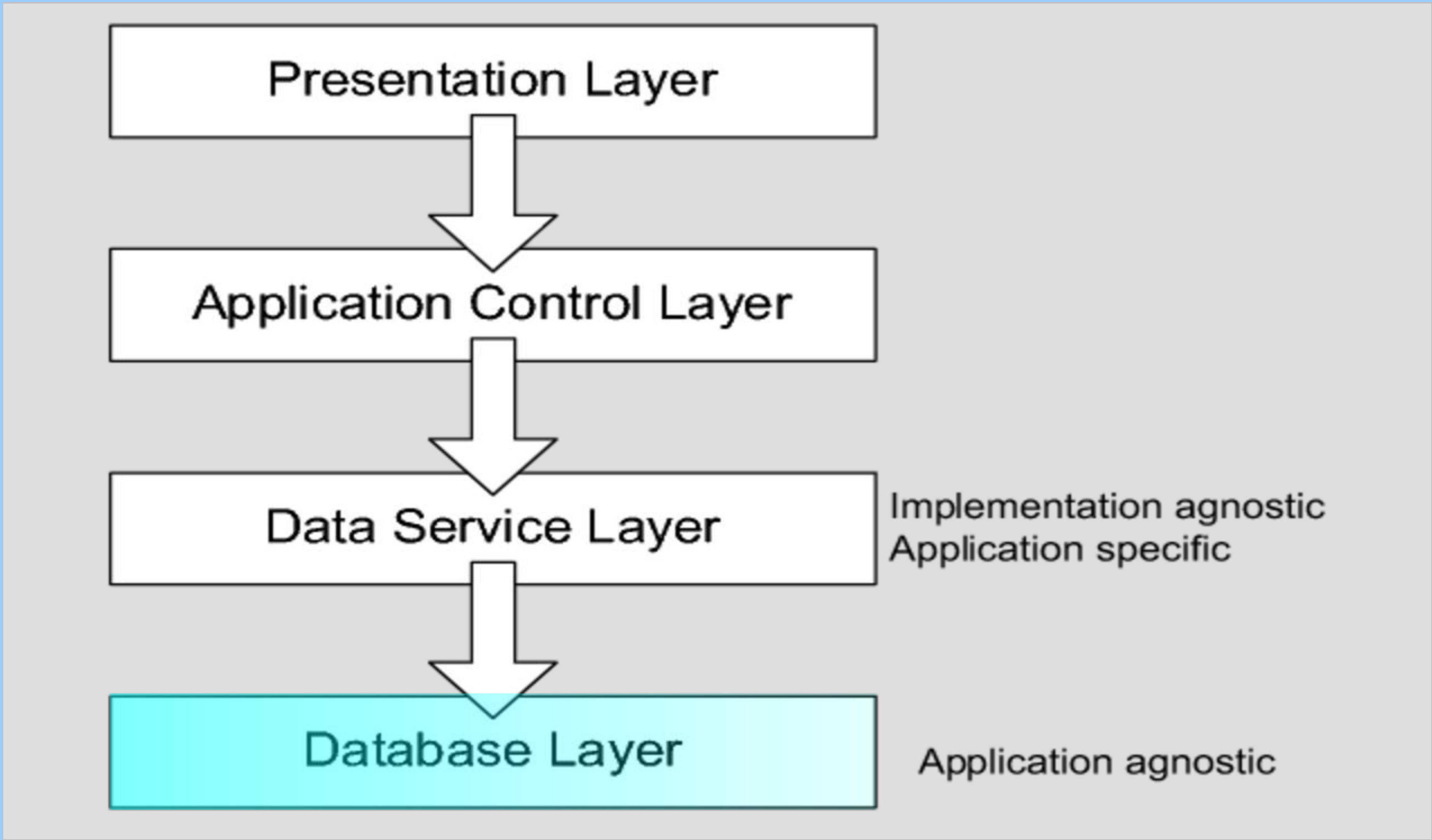
The business problem

- ABIS: course sessions – open inscriptions
- Notify enrollees of session modifications:
  - change of date, location, language
  - cancellation of session
  - notification of enrolment / cancellation / move
  - cancellation: give alternatives
- Goal: automate notification mails
  - new enrolment
  - any session change
  - notify both enrollee and contact person
  - allow for manual intervention

Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

Service-Oriented Architecture



Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

Database design: history table

- Loosely based on `SYSIBM.SYSCOPY`
- Contains “change” rows
- Goal: “log all changes”
  - able to reconstruct any previous DB state
  - avoid redundancy ==> contains no current info
  - generic: usable for other (future) applications

```
CREATE TABLE enrolhist (
  eh_seno     INTEGER     NOT NULL,
  eh_eno      SMALLINT    NOT NULL WITH DEFAULT,
  ehtimestamp TIMESTAMP   NOT NULL WITH DEFAULT,
  eh_eccode   CHAR(1)     NOT NULL,
  eholdval    VARCHAR(64),
  PRIMARY KEY (eh_seno, eh_eno, ehtimestamp, eh_eccode),
  FOREIGN KEY (eh_seno)         REFERENCES sessions(seno)          ON DELETE CASCADE  ,
--FOREIGN KEY (eh_seno, eh_eno) REFERENCES enrolments(e_seno, eno) ON DELETE CASCADE  ,
  FOREIGN KEY (eh_eccode)       REFERENCES enrolhistcases          ON DELETE RESTRICT )
```

Database design: history table

- Design choices:
  - enrolment-specific history: (eh_seno,eh_eno)

eh_eccode = 'J'	(new enrolment)
eh_eccode = 'E'	(enrolment cancellation info)
eh_eccode = 'P'	(person changed for enrolment)
eh_eccode = 'W'	(enrolment removed)

  - session-specific history: eh_eno = 0

eh_eccode = 'I'	(new session)
eh_eccode = 'D'	(session date changed)
eh_eccode = 'L'	(session language changed)
eh_eccode = 'O'	(session location changed)
eh_eccode = 'C'	(session cancellation info)
eh_eccode = 'U'	(session duration change)
eh_eccode = 'N'	(session instructor change)

Database design: history table

```
SELECT * FROM enrolhistcases ;
```

```
ECCODE  ECTEXT
---------+---------+---------+---------+---------+---------+---------+-
C       session Cancellation change (secancel)
D       session start Date (sesdate) changed
E       Enrolment (ecancel) change
I       Insertion (addition) of a new session
J       Insert (creation) of a new enrolment
L       session Language change (selang)
M       execution of the MailNoti procedure
N       session iNstructor changed
O       session lOcation (seloc_cono) changed
P       Update (change) of the Person number (estud_pno) of an enrollee
R       session date Range changed (see rgdate entries)
U       session dUration (sedur) changed
W       enrolment Wiped out (deleted)
X       ONLY FOR TEST PURPOSES (MAILNOTI)
```

Existing database tables

- sessions: one row per “course instance”

```
DECLARE sessions TABLE (
  seno       INTEGER    NOT NULL  PRIMARY KEY,
  sesdate    DATE,      -- start date
  selang     CHAR(1)    NOT NULL,  -- blank or 'N' or 'E' or 'F'
  secancel   CHAR(1)    NOT NULL,  -- blank or 'C'
  seloc_cono INTEGER(4)           REFERENCES compnos,
  seroom     CHAR(10),
  se_cno     SMALLINT   NOT NULL  REFERENCES courses,
  sedur      DECIMAL(3,1)
) ;
```

- enrolments: one row per session inscription

```
DECLARE enrolments TABLE (
  e_seno       INTEGER    NOT NULL  REFERENCES sessions,
  eno          SMALLINT   NOT NULL,
  ecancel      CHAR(1)    NOT NULL,  -- blank or 'C' or 'V'
  econtact_pno INTEGER              REFERENCES persons,
  estud_pno    INTEGER              REFERENCES persons,
  PRIMARY KEY (e_seno,eno)
) ;
```

History table – queries

- What was the database state about enrolment (seno,eno) at time instant "*ts*"?
  - already enrolled? cancelled?

```
SELECT eh_eccode, COALESCE(eholdval, ecancel)
FROM enrolments LEFT OUTER JOIN
     (SELECT *  FROM enrolhist
      WHERE  eh_eccode IN ('E','J')   -- 'E': cancel info; 'J': inscription
        AND  ehtimestamp >= :ts     ) eh ON eh_seno=e_seno AND eh_eno=eno
WHERE e_seno = :seno AND eno = :eno
ORDER BY ehtimestamp ASC
FETCH FIRST ROW ONLY
```

possible output:

• no history entries found	==> returns (NULL,current ecancel)
• one "E" entry found	==> returns ('E', eholdval)
• several "E" entries found	==> returns ('E', oldest eholdval)
• nonexisting at :ts	==> returns ('J', blank)

History table – queries

- What was the database state about enrolment (seno,eno) at time instant “*ts*”?
  - session info changed? (1) language:

```
SELECT COALESCE(eholdval, selang)
FROM sessions LEFT OUTER JOIN (SELECT *  FROM enrolhist
                               WHERE  eh_eccode = 'L'
                                 AND  ehtimestamp >= :ts ) eh
     ON eh_seno=seno AND eh_eno=0
WHERE seno = :seno
ORDER BY ehtimestamp ASC
FETCH FIRST ROW ONLY
```

    possible output:

    - no history entries found ==> returns current selang
    - one “L” entry found ==> returns eholdval
    - several “L” entries found ==> returns oldest eholdval

  - session info changed? (2) date, location, ...:
    similarly, with `eh_eccode = 'D'` or `'O'` or `...`

History table – queries

- What changed since time instant “*ts*”?

```
SELECT sessions.*, enrolments.*, eholdval
FROM enrolments INNER JOIN sessions ON e_seno=seno
     INNER JOIN enrolhist ON eh_seno = e_seno AND eh_eno IN (eno,0)
WHERE  eh_eccode <> 'I'  -- 'I' is “new session”
  AND  ehtimestamp <= :ts
ORDER BY seno, eno, eh_eccode, ehtimestamp
```

output:

- to be interpreted/grouped per (seno, eno, eh_eccode)
- only first row per group is useful (programming logic to filter)
- eccode = 'I' ==> “new session”; other entries not relevant
- eccode = 'J' ==> “new enrolment”; other entries not relevant
- eccode = 'D' ==> first eholdval is old date, sesdate is new date
- Similarly for 'L' (language), 'O' (location), 'C' (session cancel info), 'E' (enrolment cancellation info), ...

==> added denormalization (column “ehnewval”)
to simplify interpretation of history table

History table – queries

- What changed since last **notification**?
  ==> “M” entries in enrolhist, per (seno,eno)
  (automatically inserted, see further)

```
SELECT sessions.*, enrolments.*, eholdval
FROM enrolments enro INNER JOIN sessions sess ON e_seno=seno
     INNER JOIN enrolhist eh ON eh_seno = e_seno AND eh_eno IN (eno,0)
WHERE  eh_eccode NOT IN ('M','I')
  AND  NOT EXISTS (SELECT 1
                   FROM   TPVENROLHIST
                   WHERE  eh_eccode = 'M'  -- 'M' is “last notification”
                     AND  eh_seno = eh.eh_seno
                     AND  eh_eno  = enro.eno
                     AND  ehtimestamp > eh.ehtimestamp)
ORDER BY seno, eno, eccode, ehtimestamp
```

output:

- to be interpreted/grouped per (seno, eno, eccode)
- only first row per group is useful (programming logic to filter)

AFTER triggers

- guarantee history table always up-to-date
- on every change of sessions & enrolments

==> need DB2 *triggers*

- *after* every update / insert / delete of sessions & enrolments tables

```
CREATE TRIGGER eh1
  AFTER UPDATE OF selang ON sessions
  REFERENCING OLD AS O NEW AS N     FOR EACH ROW MODE DB2SQL
  WHEN (N.seno = O.seno AND N.selang <> O.selang)
   INSERT INTO enrolhist(eh_seno, eh_eccode, eholdval, ehnewval)
   VALUES(O.seno, 'L', O.selang, N.selang) ;
```

need similar triggers for any other event, e.g.:

```
CREATE TRIGGER eh2
AFTER INSERT ON sessions
REFERENCING NEW AS N    FOR EACH ROW MODE DB2SQL
 INSERT INTO enrolhist(eh_seno,eh_eccode) VALUES(N.seno, 'I') ;
```

note importance of useful choice of defaults in enrolhist!

AFTER triggers

- Careful with eholdval's data type “VARCHAR”:

```
CREATE TRIGGER eh4
  AFTER UPDATE OF seloc_cono ON sessions
  REFERENCING OLD AS O NEW AS N
  FOR EACH ROW MODE DB2SQL
  WHEN (N.seno = O.seno AND N.seloc_cono <> O.seloc_cono)
   INSERT INTO enrolhist(eh_seno,eh_eccode,eholdval,ehnewval)
   VALUES (O.seno, 'O', COALESCE(CAST(O.seloc_cono AS CHAR(5)),'') ,
                        COALESCE(CAST(N.seloc_cono AS CHAR(5)),'') ) ;
```

```
CREATE TRIGGER eh6
  AFTER UPDATE OF estud_pno ON enrolments
  REFERENCING OLD AS O NEW AS N
  FOR EACH ROW MODE DB2SQL
  WHEN (N.estud_pno <> O.estud_pno)
   INSERT INTO enrolhist(eh_seno,eh_eno,eh_eccode,eholdval,ehnewval)
   VALUES (O.e_seno, O.eno, 'P',
           COALESCE(CAST(O.estud_pno AS CHAR(5)),''),
           COALESCE(CAST(N.estud_pno AS CHAR(5)),'')) ;
```

BEFORE triggers

- useful to maintain RI for `eh_eno`, or
  to block certain updates of history table
- allowable updates *could* be:
  - removal / update / addition of “M” entries
  - change “M” to “X”
  - removal of all entries for a certain (seno,eno)

```
CREATE TRIGGER eh0
NO CASCADE BEFORE UPDATE ON enrolhist
REFERENCING OLD AS O NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN (O.eh_eccode <> 'M' OR N.eh_eccode NOT IN ('M','X'))
  SIGNAL SQLSTATE '70001' ('THIS UPDATE TO enrolhist IS DISALLOWED')
```

- returns `SQLCODE` = -438 when eccode≠ 'M'
- not yet in use

Triggers in DB2 – caveats

- possibly several triggers for same action, e.g. with different "`OF column-name`"
- order of execution = order of creation!
- careful with trigger cascading!
  ==> max. 16 levels (`SQLCODE` -724)
- belong in same UoW as triggering action
- `FOR EACH ROW` or `FOR EACH STATEMENT`
- body: `BEGIN ATOMIC ...;...;...; END` (statement delimiter must be changed)
- triggers are not fired with `LOAD`

Triggers in DB2: maintenance

- which triggers are active? ==> DB2 catalog

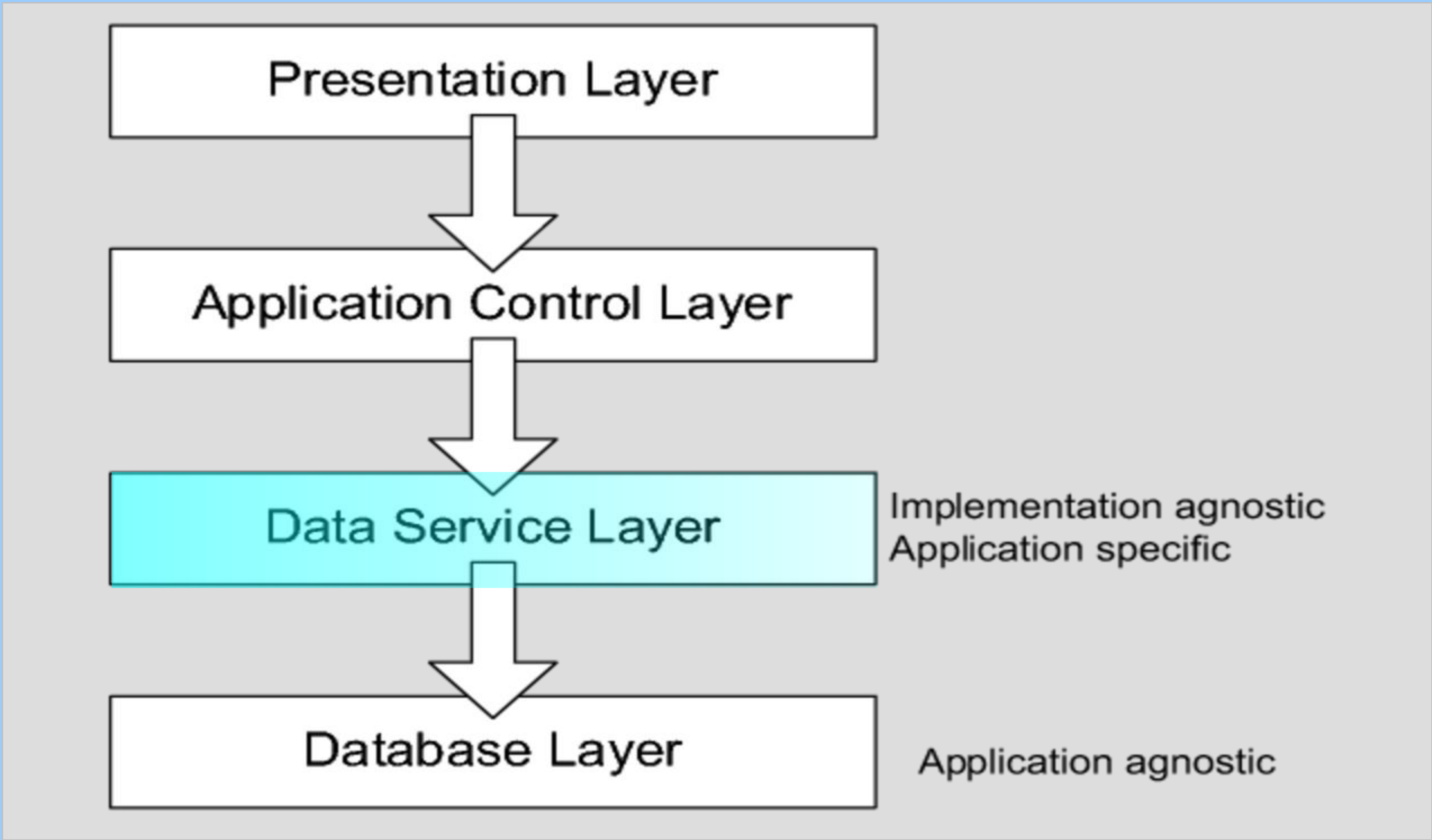
```
SELECT tbowner||'.'||tbname, seqno, text
FROM   sysibm.systriggers
WHERE  trigtime = 'A' AND trigevent = 'I' AND granularity = 'R'
ORDER BY tbowner, tbname, createdts, seqno
```

- unformatted output ==> “unreadable” (auto-formatting through REXX ?)
- trigger errors: not transparent to applic.
  - SQLCODE = -723
    - ==> “real” SQLCODE in error message
  - SQLCODE = -430
    - ==> program abend

Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

Service-Oriented Architecture



DB2 stored procedure (SP)

- to implement the Data Service Layer
- keeps DB program logic close to the data
- contains database application logic
  for a particular business application
  ==> “generate notification mails”
- authorization: is only access to data
- clean separation of DB and BI:
  - DB design details hidden in/behind the SP
  - interface API talks “business logic”

SQL in the SP

- one cursor
  ==> see before: (slide -8)
    - first row of group per (seno, eno, eh_eccode)
    - chronological order, since last “M” for (seno,eno)
- returns useful info for the confirmations:
  - at most one entry per (seno,eno)
  - only for future sessions
  - only when current ≠ previous notification
  - details:
    - name/email/language of student & contact person
    - session details (course, date, place, language)
    - old & new values for changed entities
    - list of future sessions for same course (when 'C')

Stored procedures in DB2

- “external” SPs: a two-level definition:
  - declaration in the DB2 catalog:

```
CREATE PROCEDURE schemaname.procname
(IN var1 TYPE1, ..., OUT var2 TYPE2, ..., INOUT var3 TYPE3, ...)
DYNAMIC RESULT SETS 0     -- no cursor is returned
EXTERNAL NAME 'MAILNOTI' -- name of the COBOL program
LANGUAGE COBOL          COLLID collection-name
PARAMETER STYLE GENERAL  -- do not return NULL ind., SQLSTATE, diagnostics
FENCED
MODIFIES SQL DATA       COMMIT ON RETURN NO
NO DBINFO                -- do not pass extra info (server name, UID, ...)
STOP AFTER 1 FAILURES    -- safeguard for runtime errors
WLM ENVIRONMENT WLM-name -- name of workload manager environment
```

  - implementation in e.g. COBOL; “normal” app.
- runs in separate address space; WLM
- to be called with SQL “`CALL`” statement
- input/output through CALL arguments
  of any SQL datatype (`VARCHAR`, `INT`, ...)

API design for SP

- BI driven
- must be simple to use (in `CALL` stmt)
  ==> no “result sets”; no large objects
- flexible interface
  ==> should be easy to modify API design
- interface choice: **XML**
  - SP returns single `VARCHAR(32767)` argument

```
CREATE PROCEDURE MAILNOTI
(OUT XMLtext VARCHAR(32767) CCSID EBCDIC)
DYNAMIC RESULT SETS 0 ...
```

  - XML specs described in an XMLSchema
  - versioned => synchronizing the applications

DB2 SP: caveats

- precompile, compile, bind appl. as usual
  - bind as package into collection
  - name must match SP declaration
- recompile and/or (re)bind indep. of SP
- never need to change SP object anymore
- SYSOUT goes to WLM output
  - more cumbersome debugging

DB2 SP: caveats (continued)

- runtime error ==> SP stopped
  - use DB2 command to re-activate

```
-DISPLAY PROC(schemaname.MAILNOTI)
     DSNX940I  =DB2A DSNX9DIS DISPLAY PROCEDURE REPORT FOLLOWS -
     ------- SCHEMA=schemaname
     PROCEDURE      STATUS ACTIVE QUED MAXQ TIMEOUT FAIL WLM_ENV
     MAILNOTI
                    STARTED     0    0    1       0    0 WLMname
     DSNX9DIS DISPLAY PROCEDURE REPORT COMPLETE
     DSN9022I  =DB2A DSNX9COM '-DISPLAY PROC' NORMAL COMPLETION
-START PROC(schemaname.MAILNOTI)
```

  - don't forget to first correct the error cause!

Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

XML

- Example output:

```
<?xml version="1.0" encoding="UTF-8"?>
<MailNotification>
     <Version>0.12</Version>
     <Entry id="21363-05">
          <Student>
               <Notify/>
               <PNO>23530</PNO>
               <FirstName>Marc</FirstName>
               <LastName>CRUYSMANS</LastName>
               <Email>marc.cruysmans@sdx.com</Email>
               <Language>N</Language>
               <Sex>M</Sex>
          </Student>
          <ContactPerson>
               <Notify/>
               <PNO>10859</PNO>
               <FirstName>Maria</FirstName>
               <LastName>DE RUITER</LastName>
               <Email>maria.deruiter@sdx.com</Email>
               <Language>N</Language>
               <Sex>F</Sex>
          </ContactPerson>
          <Session>
```

<continued...>

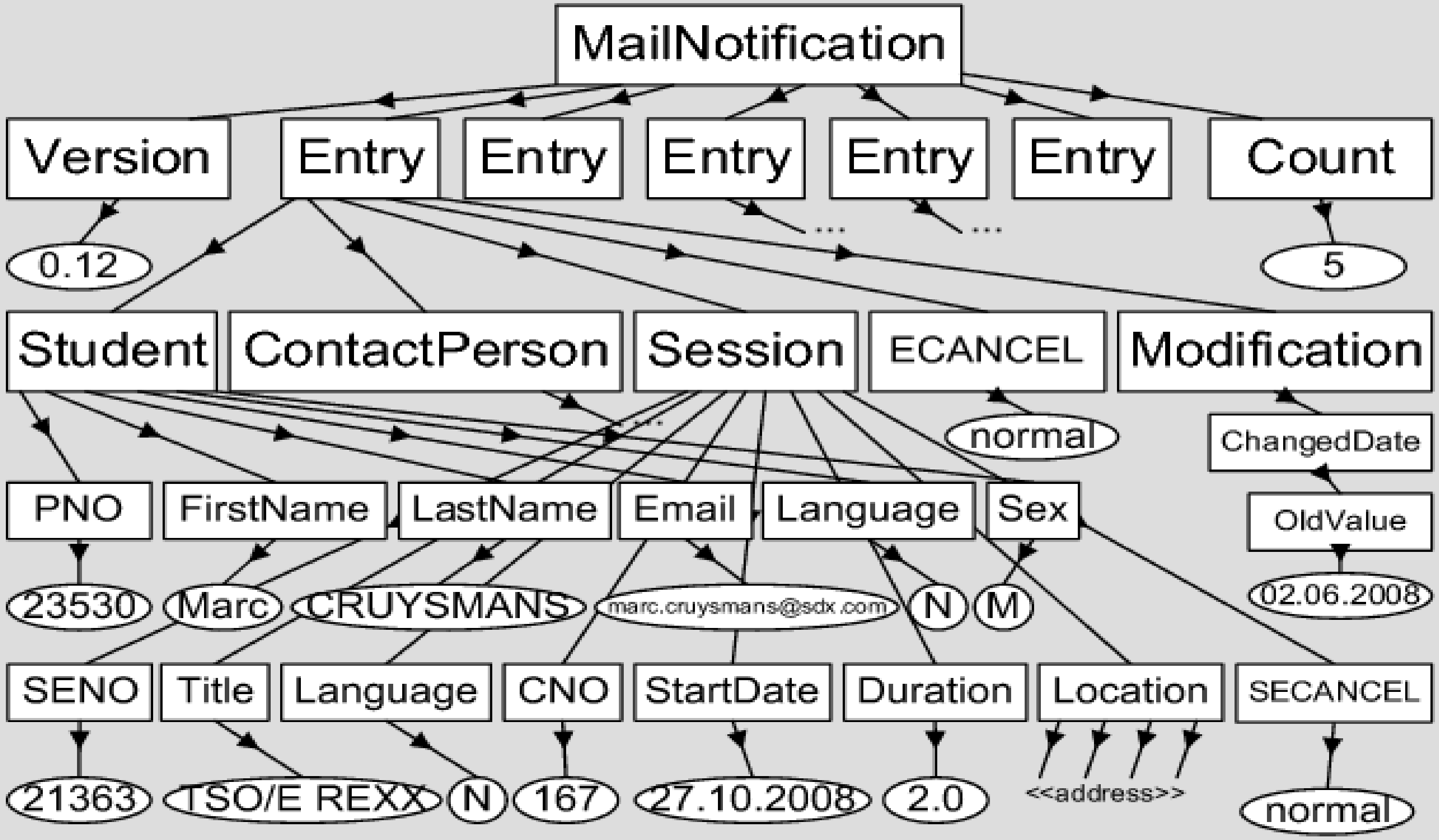
XML

- Example output (continued):

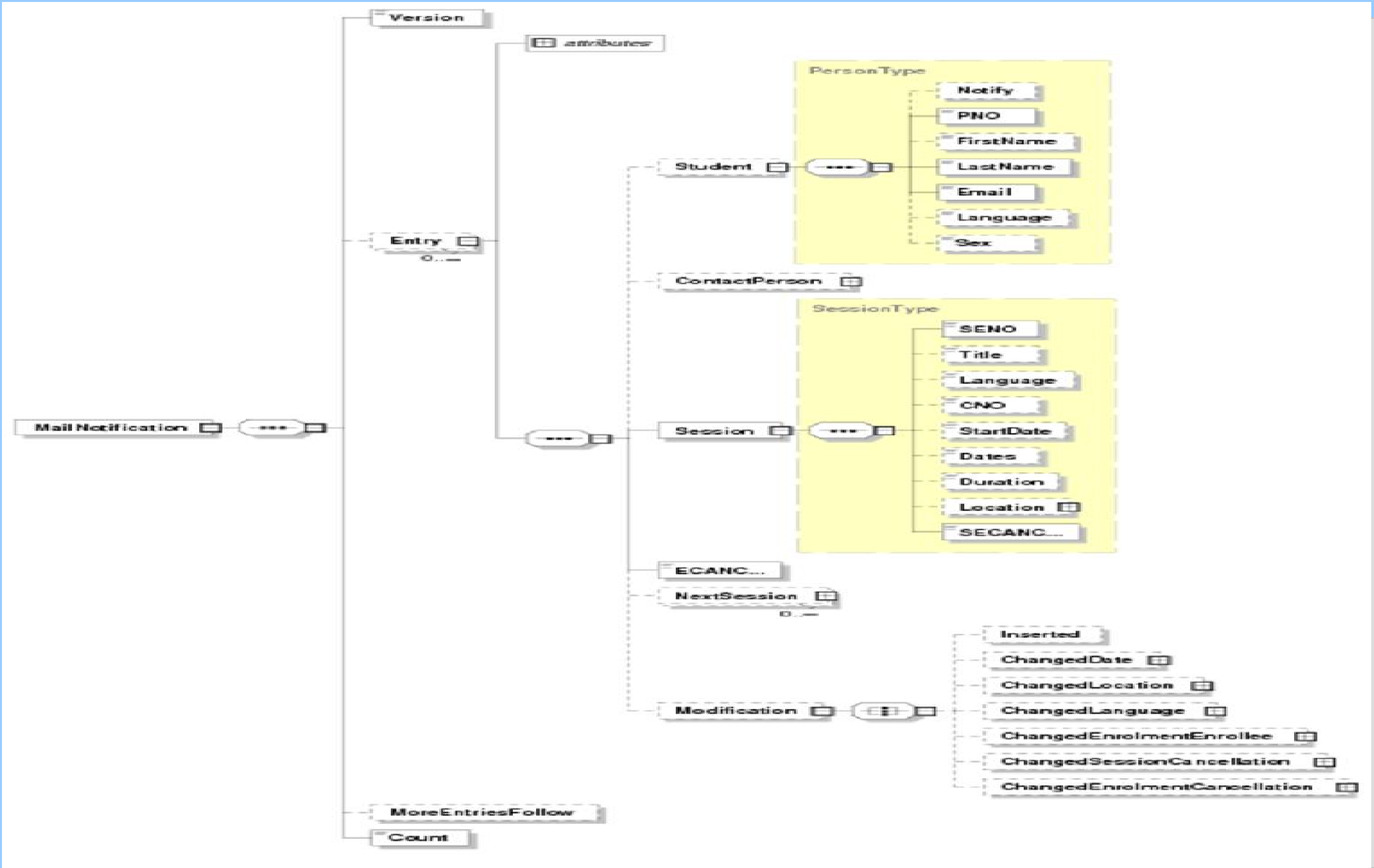
```
                <Session>
                <SENO>21363</SENO>
                <Title>TSO/E REXX</Title>
                <Language>N</Language>
                <CNO>167</CNO>
                <StartDate>27.10.2008</StartDate>
                <Duration>2.0</Duration>
                <Location>
                    <CONO>11866</CONO>
                    <CompanyName>ABIS TRAINING &amp; CONSULTING</CompanyName>
                    <Street>DIESTSEVEST</Street>
                    <StreetNumber>32</StreetNumber>
                    <ZIPCode>3000</ZIPCode>
                         <City>LEUVEN</City>
                </Location>
                <SECANCEL>normal</SECANCEL>
           </Session>
           <ECANCEL>normal</ECANCEL>
           <Modification>
                <ChangedDate><OldValue>02.06.2008</OldValue></ChangedDate>
                <Inserted/>
           </Modification>
     </Entry>
     <Entry> ........ </Entry>
     ...
     <Count>28</Count>
</MailNotification>
```

XML: structure

- document object model (DOM):



XML Schema



Version
attributes
PersonType
Notify
PNO
FirstName
LastName
Email
Language
Sex
Student
Entry
0..∞
ContactPerson
SessionType
SENO
Title
Language
CNO
StartDate
Dates
Duration
Location
SECANC...
MailNotification
Session
ECANC...
NextSession
0..∞
Inserted
ChangedDate
ChangedLocation
ChangedLanguage
ChangedEnrolmentEnrollee
ChangedSessionCancellation
ChangedEnrolmentCancellation
Modification
MoreEntriesFollow
Count

XML Schema

- Formal way to describe an XML structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
     <xs:element name="MailNotification">
          <xs:complexType>
               <xs:sequence>
                   <xs:element name="Version">
                       <xs:simpleType>
                           <xs:restriction base="xs:string">
                               <xs:enumeration value="0.12"/>
                           </xs:restriction>
                       </xs:simpleType>
                   </xs:element>
                   <xs:element name="Entry" minOccurs="0" maxOccurs="unbounded">
                       <xs:complexType>
                           <xs:sequence>
                               <xs:element name="Student"
                                           type="PersonType" minOccurs="0"/>
                               <xs:element name="ContactPerson"
                                           type="PersonType" minOccurs="0"/>
                               <xs:element name="Session" type="SessionType"/>
                               <xs:element name="ECANCEL">
                                   <xs:simpleType>
                                       <xs:restriction base="xs:string">
                                           <xs:enumeration value="normal"/>
                                           <xs:enumeration value="cancelled"/>
                                           <xs:enumeration value="moved"/>
                                       </xs:restriction>
                                   </xs:simpleType>
                               </xs:element>                 <continued...>
```

XML Schema (continued)

```
          <xs:element name="NextSession" type="SessionType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="Modification" minOccurs="0">
            <xs:complexType>
              <xs:all>
                <xs:element name="Inserted" minOccurs="0"/>
                <xs:element name="ChangedDate" type="ChangedEntryType" minOccurs="0"/>
                <xs:element name="ChangedLocation" type="ChangedEntryType" minOccurs="0"/>
                <xs:element name="ChangedLanguage" type="ChangedEntryType" minOccurs="0"/>
                <xs:element name="ChangedEnrolmentEnrollee" type="ChangedEntryType" minOccurs="0"/>
                <xs:element name="ChangedSessionCancellation" type="ChangedEntryType" minOccurs="0"/>
                <xs:element name="ChangedEnrolmentCancellation" type="ChangedEntryType" minOccurs="0"/>
              </xs:all>
            </xs:complexType>
          </xs:element>  <!-- Modification -->
         </xs:sequence>
         <xs:attribute name="id" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:pattern value="\d{5}-\d\d"/>
            </xs:restriction>
          </xs:simpleType>
         </xs:attribute>
       </xs:complexType>
     </xs:element>   <!-- Entry -->
     <xs:element name="MoreEntriesFollow" minOccurs="0"/>
     <xs:element name="Count" type="xs:integer"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>     <!-- MailNotification -->
```

XML Schema (continued)

```
   <xs:complexType name="PersonType">
      <xs:sequence>
           <xs:element name="Notify" minOccurs="0"/>
           <xs:element name="PNO" type="xs:integer"/>
           <xs:element name="FirstName" type="xs:string" minOccurs="0"/>
           <xs:element name="LastName" type="xs:string"/>
           <xs:element name="Email" type="xs:string"/>
           <xs:element name="Language" minOccurs="0">
                <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:enumeration value="N"/>
                        <xs:enumeration value="F"/>
                        <xs:enumeration value="E"/>
                        <xs:enumeration value="D"/>
                     </xs:restriction>
                </xs:simpleType>
           </xs:element>
           <xs:element name="Sex" minOccurs="0">
                <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:enumeration value="M"/>
                        <xs:enumeration value="F"/>
                        <xs:enumeration value=" "/>
                     </xs:restriction>
                </xs:simpleType>
           </xs:element>
      </xs:sequence>
   </xs:complexType> <!-- PersonType -->
   <xs:complexType name="ChangedEntryType">          (etc.)
   ...
 </xs:schema>
```

XML Schema: how to use

- functions as API description
- communication tool between developers
- use graphical software to manipulate
  - e.g. XmlSpy of Altova
  - see `http://www.altova.com/IBM_DB2_9_pureXML`

    ==> “strategic partnership” Altova & IBM
- easily allows for API versioning
- can auto-generate COBOL from Schema
  - By using XSLT

COBOL and XML

- need no help from DB2 to generate XML
  - fully supported in DB2 9 only ...
- use COBOL “STRING” command
  - flexible way to CONCAT text pieces
  - Enterprise COBOL compiler:
    mix with “XML GENERATE” command

```
MAIN.
    MOVE 1 TO SIZ
   STRING '<?xml version="1.0" encoding="ISO-8859-1"?>' NL
            DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    EXEC SQL  OPEN c  END-EXEC
    STRING '<MailNotification>' NL '<Version>0.11</Version>' NL
           DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    EXEC SQL  FETCH c INTO :array  END-EXEC
    PERFORM PROCESS-NEXT-ENROLMENT  UNTIL SQLCODE NOT = 0
   XML GENERATE XMD(SIZ:) FROM Qount COUNT IN CNT
    MOVE 'C' TO XMD(SIZ + 1 : 1)
    ADD CNT TO SIZ
    MOVE 'C' TO XMD(SIZ - 6 : 1)
```

COBOL and XML (continued)

```
PROCESS-NEXT-ENROLMENT.
    ADD 1 TO Qount
    MOVE NSENO TO SENO-DISP
    MOVE NENO  TO ENO-DISP
    STRING '<Entry id="' SENO-ENO '">' DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    IF NSTUPNO NOT = 0 THEN
      STRING '<Student>' DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
      IF NSTUNOTIFY = 'Y'
        STRING '<Notify/>' DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
      END-IF
      MOVE NSTUPNO TO PNO IN XML-GENERATE-VARS
      XML GENERATE XMD(SIZ:) FROM PNO IN XML-GENERATE-VARS COUNT IN CNT
      ADD CNT TO SIZ
      ...
    END-IF
    STRING '<Session>' DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    MOVE NSENO TO SENO IN XML-GENERATE-VARS
    XML GENERATE XMD(SIZ:) FROM SENO IN XML-GENERATE-VARS  COUNT IN CNT
    ADD CNT TO SIZ
    ...
    STRING '</Session>' DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    EXEC SQL  FETCH c INTO :array  END-EXEC
    STRING '</Entry>' NL DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    IF SIZ > 25000 THEN
       STRING '<MoreEntriesFollow/>' NL DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
       MOVE 100 TO SQLCODE
    END-IF
    .
```

COBOL and XML: caveats

- codepage issues:
  - receiving end expects UTF-8 or ISO-8859-1:

    ```
    STRING '<?xml version="1.0" encoding="ISO-8859-1"?>' NL
           DELIMITED BY SIZE INTO XMD WITH POINTER SIZ
    ```

    while COBOL generates EBCDIC!
  - EBCDIC has no std “newline” character

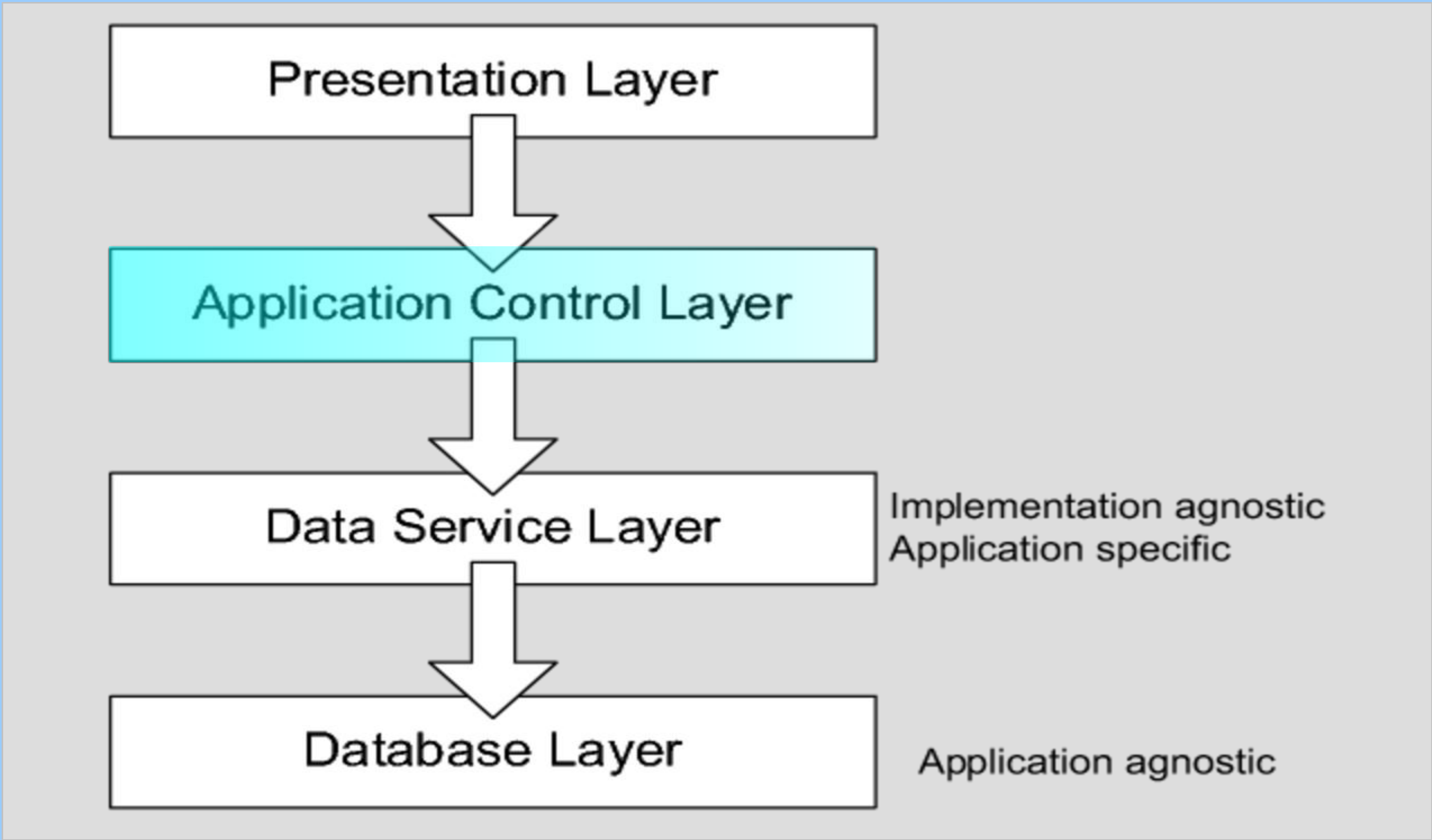
    (IBM: XML specs for “whitespace” will be extended)
  - How to “fake” newline (Unicode CP 10):

    ```
    ENVIRONMENT DIVISION.
    CONFIGURATION SECTION.
    SPECIAL-NAMES.
        SYMBOLIC CHARACTERS   NL   ARE   38.
    ```
- 32787 byte limit
- setting a `PIC S9(4) COMP` to 32000 ...

Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

Service-Oriented Architecture



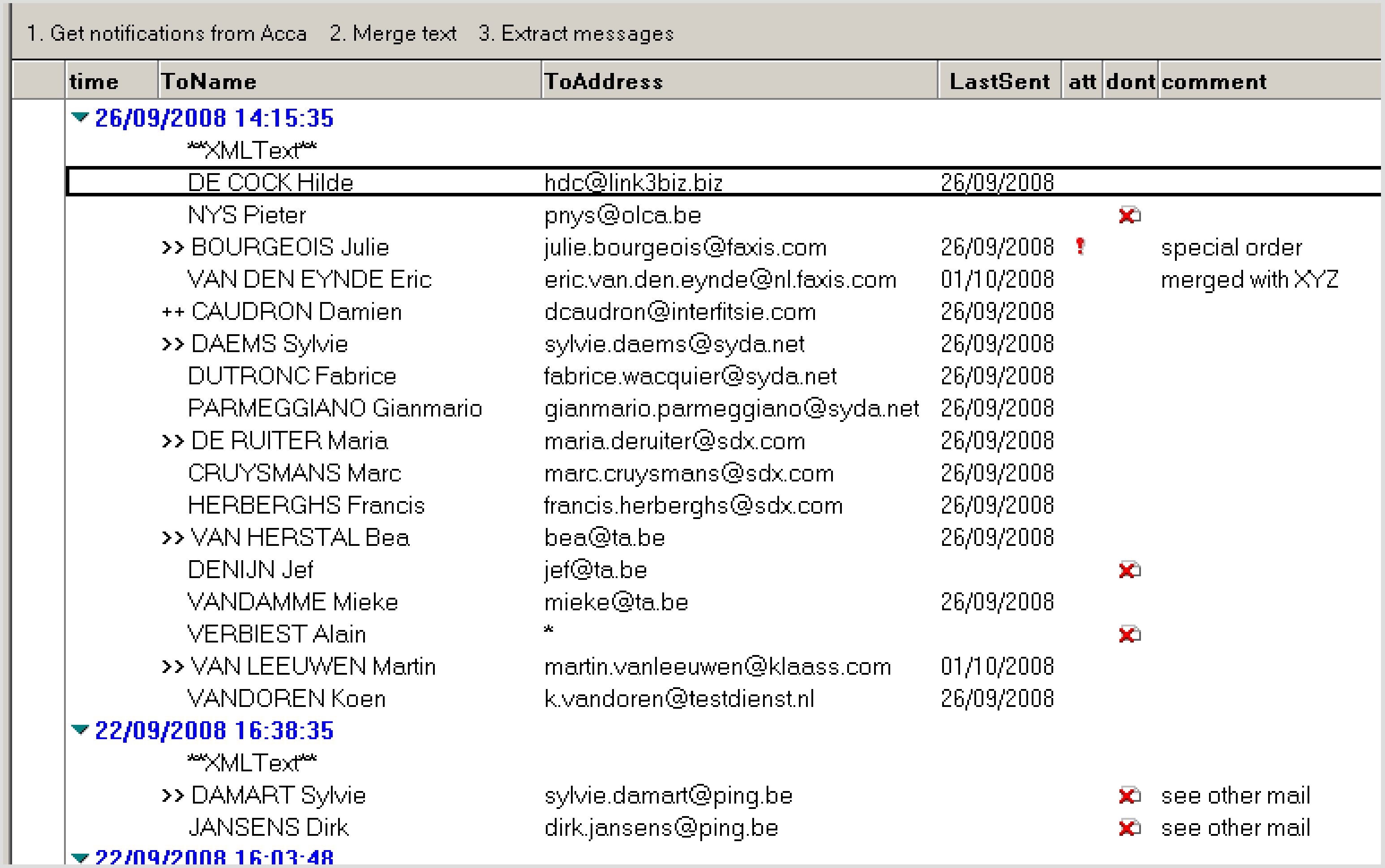
Lotus Notes mail server

- Domino LotusScript on mail server
  - accesses the SP with ODBC (SQL CALL)
  - XSLT to glue together 32000-byte pieces
  - XSLT to merge entries for same destination & to integrate with business logic (interpretation / highlighting / suppression ...)
- not using IBM Lotus Enterprise Integrator
  - no need for complex framework
  - earlier experience with LotusScript
- script triggered by user interface

Design challenges

- 2-phase commit
  - DB changed when SP run (“M” entries), but mail not yet sent
  - what if mail is returned “undeliverable”?
- DB2 connection needed on all clients, or just on the Lotus Notes server?

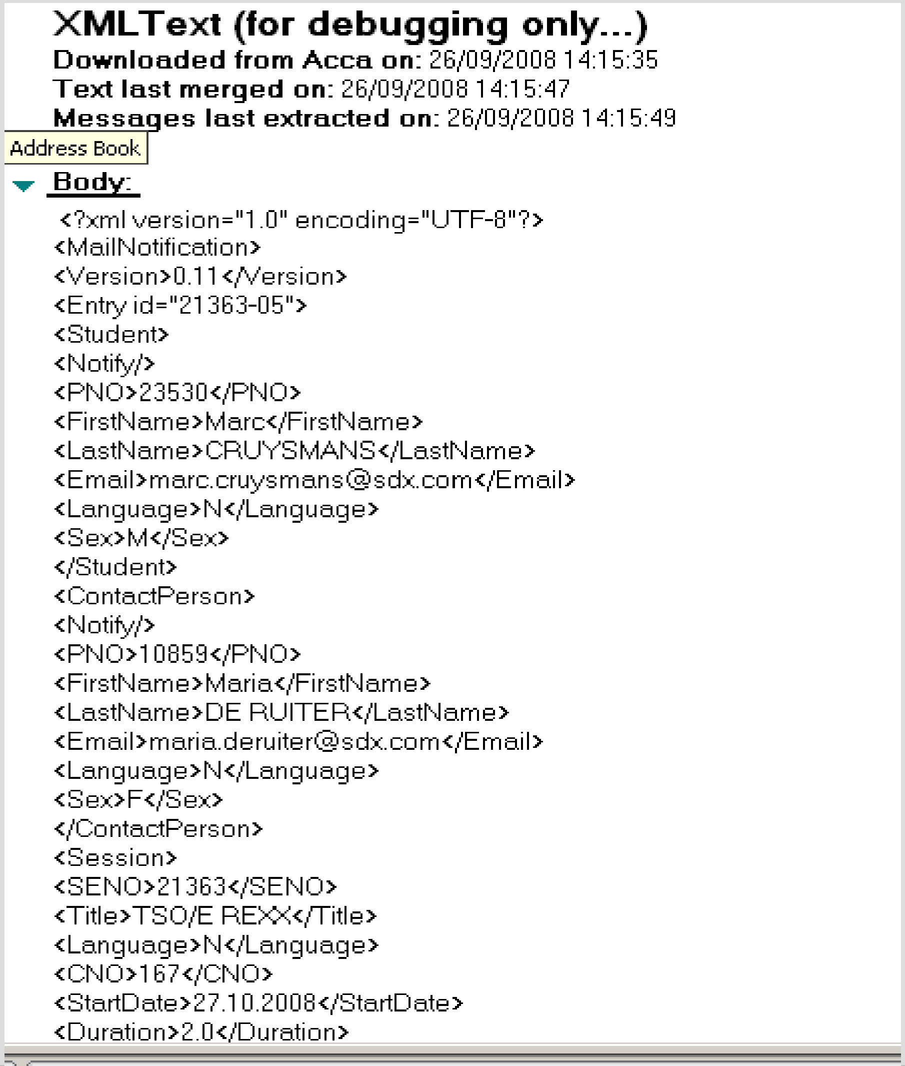
Debugging tool



1. Get notifications from Acca   2. Merge text   3. Extract messages

time	ToName	ToAddress	LastSent	att	dont	comment
26/09/2008 14:15:35						
	XMLText					
	DE COCK Hilde	hdc@link3biz.biz	26/09/2008			
	NYS Pieter	pnys@olca.be			✗	
	>> BOURGEOIS Julie	julie.bourgeois@faxis.com	26/09/2008	!		special order
	VAN DEN EYNDE Eric	eric.van.den.eynde@nl.faxis.com	01/10/2008			merged with XYZ
	++ CAUDRON Damien	dcaudron@interfitsie.com	26/09/2008			
	>> DAEMS Sylvie	sylvie.daems@syda.net	26/09/2008			
	DUTRONC Fabrice	fabrice.wacquier@syda.net	26/09/2008			
	PARMEGGIANO Gianmario	gianmario.parmeggiano@syda.net	26/09/2008			
	>> DE RUITER Maria	maria.deruiter@sdx.com	26/09/2008			
	CRUYSMANS Marc	marc.cruysmans@sdx.com	26/09/2008			
	HERBERGHS Francis	francis.herberghs@sdx.com	26/09/2008			
	>> VAN HERSTAL Bea	bea@ta.be	26/09/2008			
	DENIJN Jef	jef@ta.be			✗	
	VANDAMME Mieke	mieke@ta.be	26/09/2008			
	VERBIEST Alain	*			✗	
	>> VAN LEEUWEN Martin	martin.vanleeuwen@klaass.com	01/10/2008			
	VANDOREN Koen	k.vandoren@testdienst.nl	26/09/2008			
22/09/2008 16:38:35						
	XMLText					
	>> DAMART Sylvie	sylvie.damart@ping.be			✗	see other mail
	JANSENS Dirk	dirk.jansens@ping.be			✗	see other mail
22/09/2008 16:03:48						

Graphical user interface

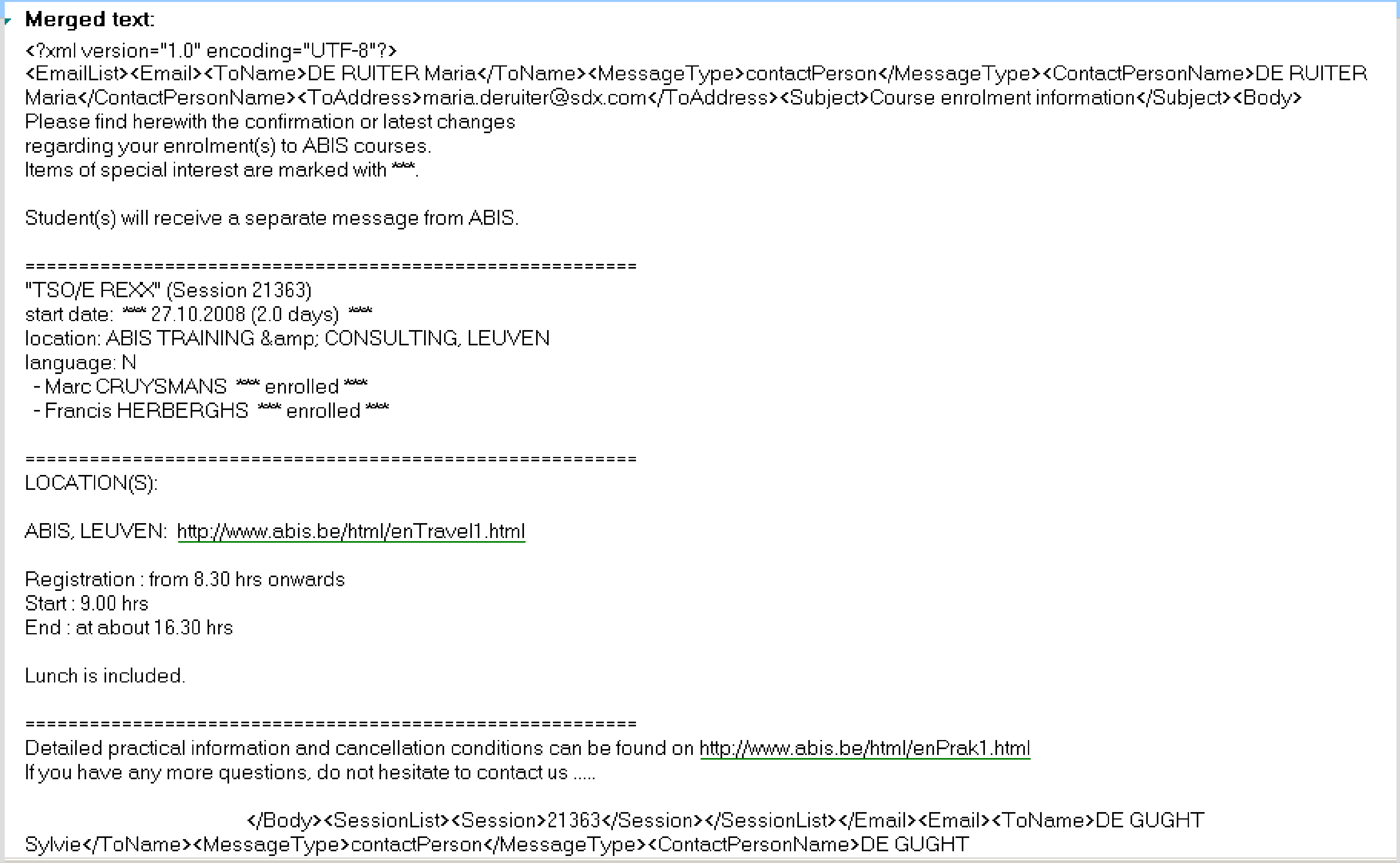


XMLText (for debugging only...)
Downloaded from Acca on: 26/09/2008 14:15:35
Text last merged on: 26/09/2008 14:15:47
Messages last extracted on: 26/09/2008 14:15:49

Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<MailNotification>
<Version>0.11</Version>
<Entry id="21363-05">
<Student>
<Notify/>
<PNO>23530</PNO>
<FirstName>Marc</FirstName>
<LastName>CRUYSMANS</LastName>
<Email>marc.cruysmans@sdx.com</Email>
<Language>N</Language>
<Sex>M</Sex>
</Student>
<ContactPerson>
<Notify/>
<PNO>10859</PNO>
<FirstName>Maria</FirstName>
<LastName>DE RUITER</LastName>
<Email>maria.deruiter@sdx.com</Email>
<Language>N</Language>
<Sex>F</Sex>
</ContactPerson>
<Session>
<SENO>21363</SENO>
<Title>TSO/E REXX</Title>
<Language>N</Language>
<CNO>167</CNO>
<StartDate>27.10.2008</StartDate>
<Duration>2.0</Duration>
```

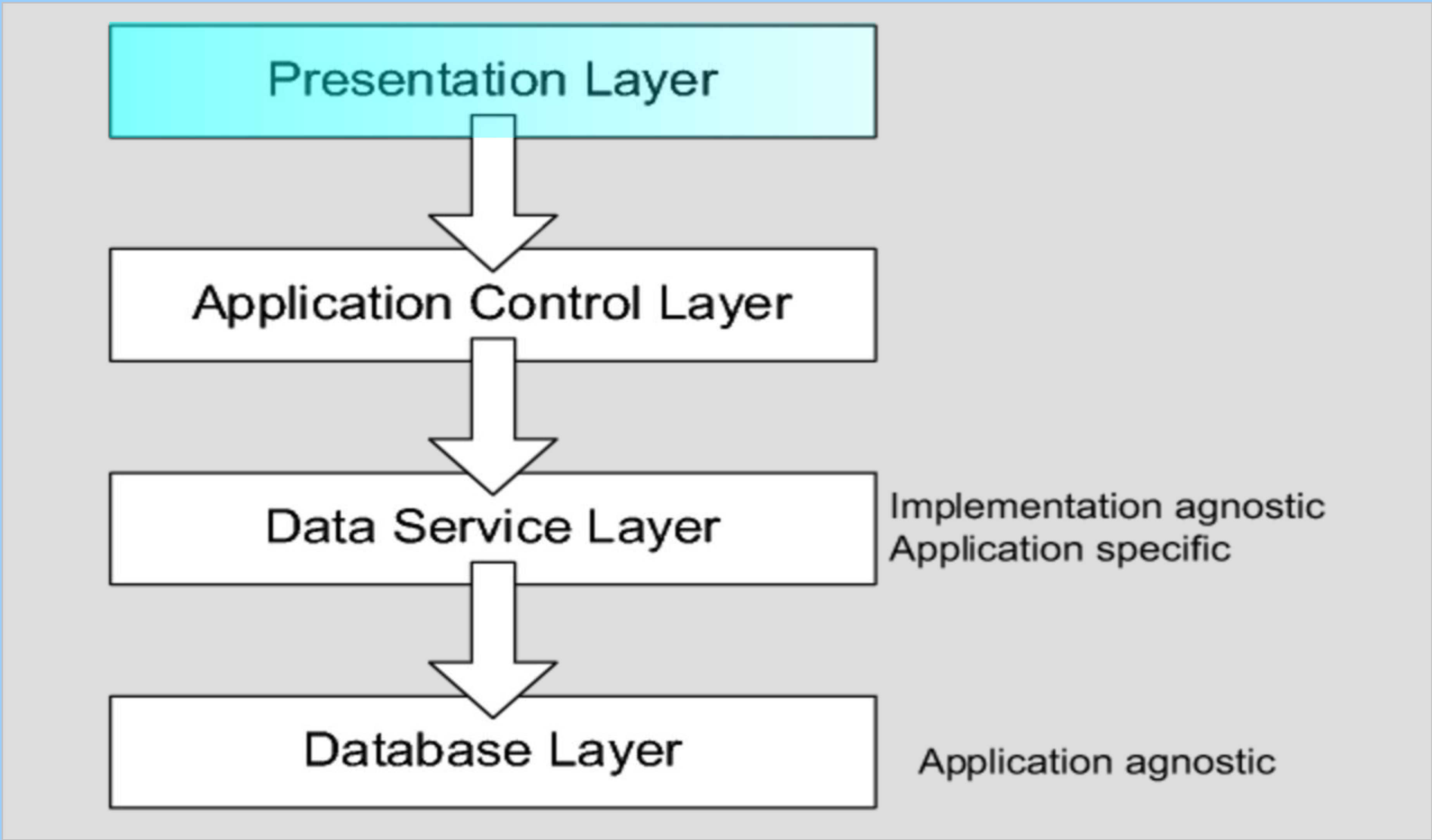
Graphical user interface



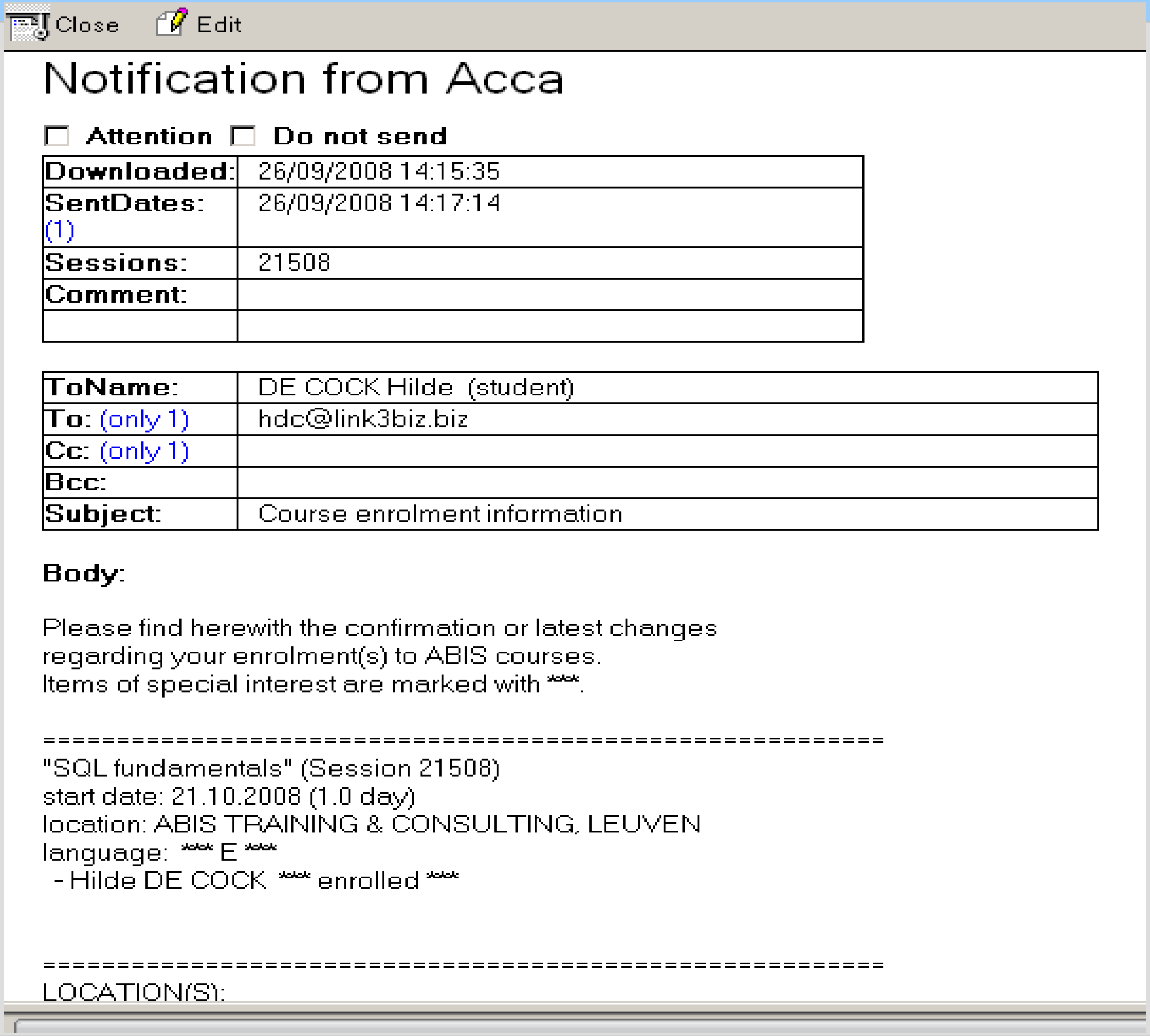
Merged text:

```
<?xml version="1.0" encoding="UTF-8"?>
<EmailList><Email><ToName>DE RUITER Maria</ToName><MessageType>contactPerson</MessageType><ContactPersonName>DE RUITER
Maria</ContactPersonName><ToAddress>maria.deruiter@sdx.com</ToAddress><Subject>Course enrolment information</Subject><Body>
Please find herewith the confirmation or latest changes
regarding your enrolment(s) to ABIS courses.
Items of special interest are marked with ***.

Student(s) will receive a separate message from ABIS.

============================================================
"TSO/E REXX" (Session 21363)
start date:  *** 27.10.2008 (2.0 days)  ***
location: ABIS TRAINING &amp; CONSULTING, LEUVEN
language: N
 - Marc CRUYSMANS  *** enrolled ***
 - Francis HERBERGHS  *** enrolled ***

============================================================
LOCATION(S):

ABIS, LEUVEN:  http://www.abis.be/html/enTravel1.html

Registration : from 8.30 hrs onwards
Start : 9.00 hrs
End : at about 16.30 hrs

Lunch is included.

============================================================
Detailed practical information and cancellation conditions can be found on http://www.abis.be/html/enPrak1.html
If you have any more questions, do not hesitate to contact us .....

                                </Body><SessionList><Session>21363</Session></SessionList></Email><Email><ToName>DE GUGHT
Sylvie</ToName><MessageType>contactPerson</MessageType><ContactPersonName>DE GUGHT
```

Service-Oriented Architecture



Graphical user interface



Notification from Acca

☐ Attention ☐ Do not send

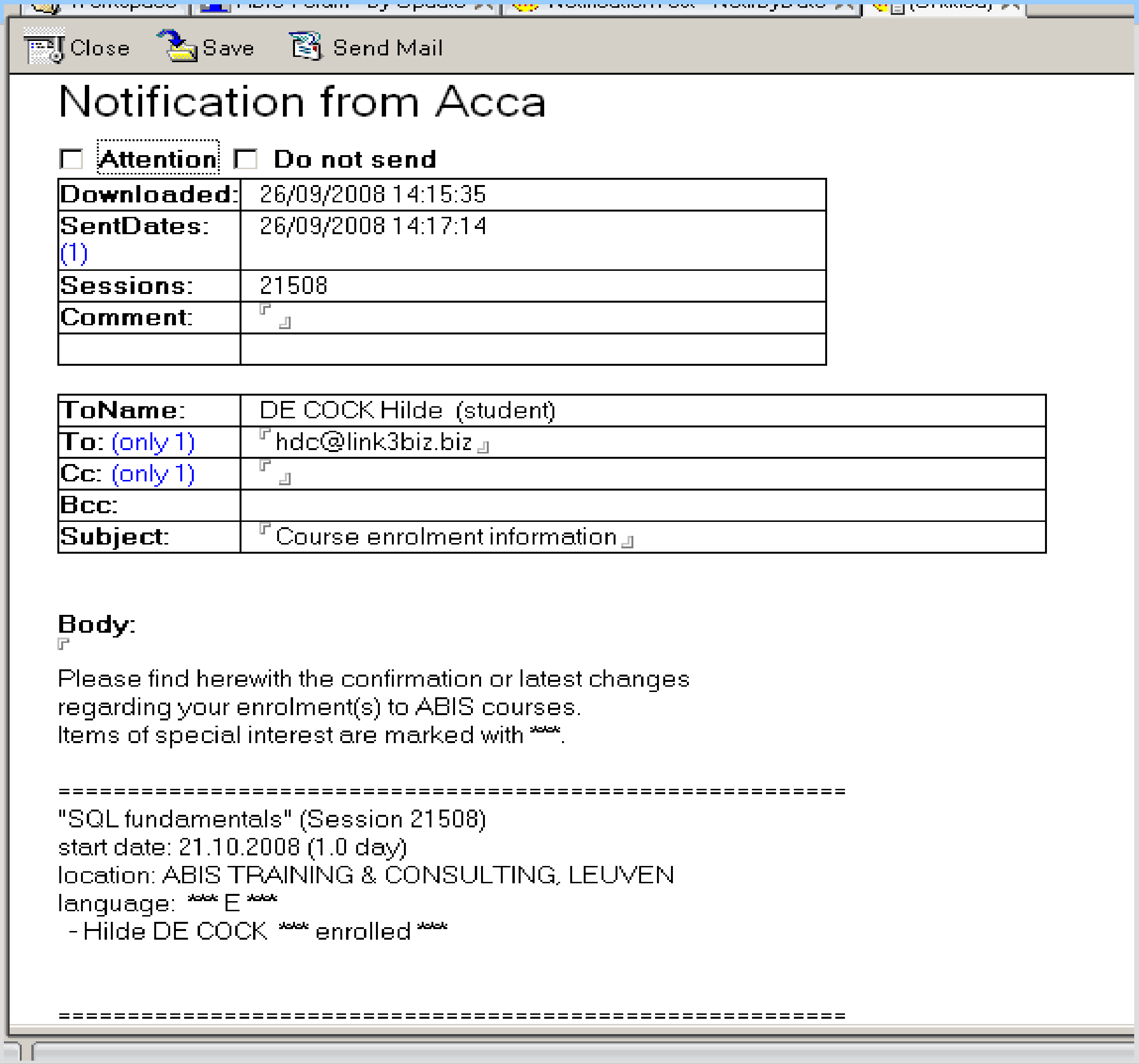
Downloaded:	26/09/2008 14:15:35
SentDates: (1)	26/09/2008 14:17:14
Sessions:	21508
Comment:	

ToName:	DE COCK Hilde (student)
To: (only 1)	hdc@link3biz.biz
Cc: (only 1)	
Bcc:	
Subject:	Course enrolment information

Body:

```
Please find herewith the confirmation or latest changes
regarding your enrolment(s) to ABIS courses.
Items of special interest are marked with ***.

=============================================================
"SQL fundamentals" (Session 21508)
start date: 21.10.2008 (1.0 day)
location: ABIS TRAINING & CONSULTING, LEUVEN
language:  *** E ***
 - Hilde DE COCK  *** enrolled ***


=============================================================
LOCATION(S):
```

Graphical user interface



Notification from Acca

☐ Attention ☐ Do not send

Downloaded:	26/09/2008 14:15:35
SentDates: (1)	26/09/2008 14:17:14
Sessions:	21508
Comment:	

ToName:	DE COCK Hilde (student)
To: (only 1)	hdc@link3biz.biz
Cc: (only 1)	
Bcc:	
Subject:	Course enrolment information

Body:

Please find herewith the confirmation or latest changes
regarding your enrolment(s) to ABIS courses.
Items of special interest are marked with ***.

```
============================================================
"SQL fundamentals" (Session 21508)
start date: 21.10.2008 (1.0 day)
location: ABIS TRAINING & CONSULTING, LEUVEN
language: *** E ***
 - Hilde DE COCK *** enrolled ***


============================================================
```

Agenda

- sketch of the business problem
- service-oriented architecture
- database design: history table + triggers
- DB2 stored procedures
- XML
- user interface design
- practical problems and solutions

Q & A

- ...

Putting it all together: experiences with stored procedures, triggers, and XML on DB2 v8 for z/OS

Peter Vanroose

pvanroose@abis.be

ABIS Training & Consulting
Leuven / Woerden
http://www.abis.be/

Nationale GSE-conferentie “The Next Step”

Zeist, 29 Oktober 2008