



Code performance workshop

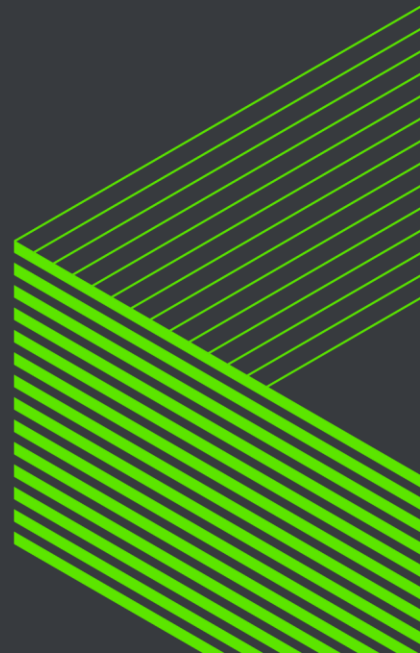
The TL;DR version



White Star Software
www.wss.com

agenda

- Intro
 - This is the short version of the Code Performance Workshop
- Compile-listing ain't enough
- Network effects
- Unnecessary work
- Conclusion



your hosts

- One named Peter
- One named Paul



one named Peter (Judge)



pjudge@progress.com

Software Architect*

@ Progress since 2003

Integration-y stuff – Authentication Gateway, HTTP-Out, Corticon et al

OE Best Practices / OERA / AutoEdge / CCS

4GL since 1996

* Aka programmer who knows PowerPoint

one named Paul (Koufalis)

pk@wss.com

Progress DBA and UNIX admin since 1994

Expert consulting related to technical aspects
of Progress and OpenEdge

Wide range of experience

Small 10 person offices to 2500+ concurrent users

AIX, HPUX, Linux, Windows...if Progress runs on it,
I've worked on it

Single malts and American bourbons





White Star Software
www.wss.com

- The oldest and most respected independent DBA consulting firm in the world
- Four of the world's top OpenEdge DBAs
- Author of ProTop, the #1 FREE OpenEdge Database Monitoring Tool
 - <http://dashboard.dbappraise.com>

compile-listing ain't enough

- Everybody knows that there's a problem: "It's slow". Now what?
 1. Find the source of the problem
 1. Reproduce the issue or at least follow it *live* - **HOW ?**
 2. Identify the errant code – **HOW ?**
 3. Fix it. That's the easy part

Where are you hiding Mr. Problem?

"The system is slow" or better yet "it's slow". Sound familiar?

Three ways to identify problems

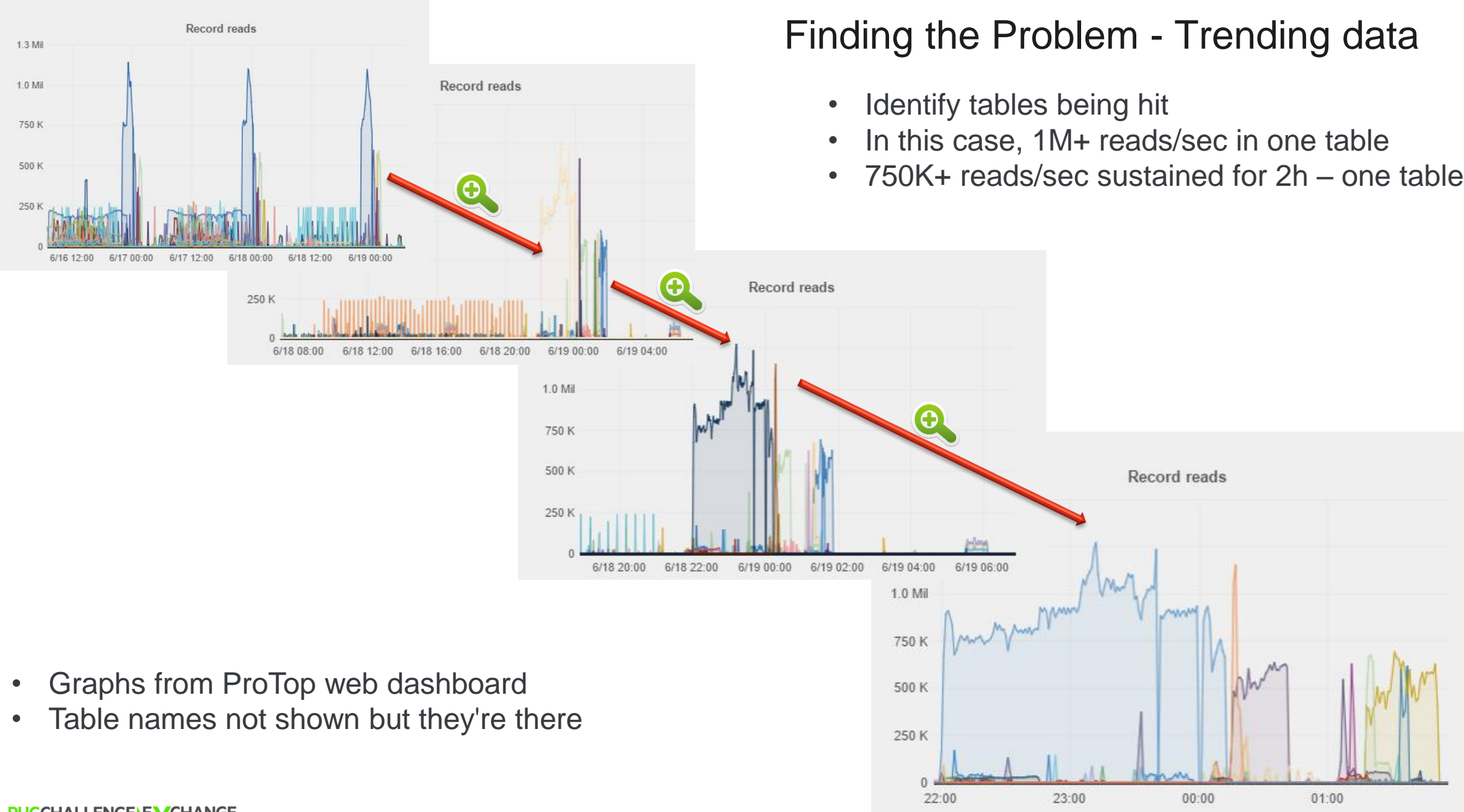
1. The user actually tells you what he was doing
 - Go buy a lottery ticket
2. Trend data
 - Conceptually easy (more later)
3. Sit in front of a screen 24/7 and watch the system

Finding the Problem - Trending data

- In a nutshell: there's a LOT of data
- Tools like ProTop and OE Management trend global data
 - For example: Index and table usage statistics
- Trending data at the user level is daunting
 - Ex.: 500 users, 500 tables, 1500 indexes => 1M data points per sample
 - Most of them zero
 - How often do you sample? Every minute? 5 minutes?
 - "Don't store the zeros" - sure – but there is a CPU cost to read the million samples and discard them
 - Same idea for CSC (Client Statement Cache)

Finding the Problem - Trending data

- Identify tables being hit
- In this case, 1M+ reads/sec in one table
- 750K+ reads/sec sustained for 2h – one table



- Graphs from ProTop web dashboard
- Table names not shown but they're there

Finding the problem – from trending data to source code

- You have to play detective
- Repeating pattern tells me it's a nightly batch
- If necessary, login at 10:00pm and start monitoring in real time
- Processes consuming excessive CPU or disk I/O
 - Use nmon (Linux, AIX) or glance (HPUX) or Windows PerfMon
- Processes doing excessive database reads/writes
 - Use VST data (more later)

Identify bad code

- Focus on 3 tools to help you identify problems:
 - VSTs
 - LOG-MANAGER
 - -zqil
- Profiler is a great tool but we don't have time

Using VST Data

- Assumption for Part 1: Problem is query-related
 - Peter will talk later about unnecessary work in your code
- `_UserTableStat / _UserIndexStat` VSTs
 - Which process is hitting what tables/indexes
 - Calculate volume but don't ignore rate and relative size
 - 1K reads in 1M records may be normal
 - 10M reads in 10K records probably is not
 - 750K reads/sec is probably a tad high
- CSC: What code is running
 - Turn on Client Statement Cache to follow in real time
 - Info in `_Connect` VST



- Collect data before and after the problematic code and subtract
- Do a little math with `etime()` to calculate rates

```
find _connect where _connect-pid = 12345.
```

```
for each _UserTableStat where _UserTableStat-conn = _connect-usr:  
  find _file no-lock where _file-number = _UserTableStat-id.  
  displ _file-name _UserTableStat-read.  
end.
```

```
for each _UserIndexStat where _UserIndexStat-conn = _connect-usr:  
  find _index no-lock where _idx-num = _UserIndexStat-id.  
  displ _index-name _UserIndexStat-read.  
end.
```



ProTop Free
does it all for you

Using VST Data

```

..... Table Activity .....
.  Tbl# Area# Table Name          RM Chain   #Records   Turns   Create   Read v   Update   Delete   OS Read .
.  >  790   20 s_crm-valid-queue          60        1193966   0.16      0    194453     0       0       0 .
.    936   18 wm-send                      16         5368656   0.00      0       341     0       0       0 .
.    782   20 s_crm-crm-field              60          48       0.52      0       25     0       0       0 .
.    787   20 s_crm-lat-field              60          48       0.36      0       17     0       0       0 .
.    849   22 wb_dept-user                60         7088     0.00      0       12     0       0       0 .
.    798   20 s_param                      39         1979     0.00      0        8     0       0       0 .
.    169   20 cost-factor                  60          18       0.39      0        7     0       0       0 .
.    528   18 prod-exp                     37        217216   0.00      0        7     0       0       0 .
.    557   18 product                      1355       214159   0.00      0        7     0       0       0 .
.    654   142 so-pick-d                  383       28455997  0.00      0        7     0       0       0 .
.     1    22 alternate                    58         488     0.00      0        0     0       0       0 .
.     2    18 am-list                       60          2     0.00      0        0     0       0       0 .
.     3     8 am-list-d                     4           0     0.00      0        0     0       0       0 .
..... Index Activity .....
.  Idx# Area# Index Name          Lvl#   Blocks Util Idx Root   Create   Read v   Split   Delete BlkDl Note .
.  >  1744   21 s_crm-valid-queue.s_crm-changes  3       1,872  90%   35839     0    194,473     0       0       0 .
.    1964   19 wm-send.WMS-ID                3       6,353  94%   13695     0       943     0       0       0 U .
.     3     6 _Field._Field-Name             0         0    0%    68352     0       621     0       0       0 U .
.    1953   21 wm-pick.WMS-ID                2         508  96%   41599     0       149     0       0       0 U .
.    1456   19 so-pick.ctrl-machine          3         969  90%    9535     0        68     0       0       0 .
.    1463   19 so-pick.shipped              2         292  98%    9983     0        65     0       0       0 .
.     4     6 _Field._Field-Position         0         0    0%   68416     0        51     0       0       0 U .
.     6     6 _Index-Field._Index/Number     0         0    0%   68544     0        36     0       0       0 PU .
.    1734   21 s_crm-crm-field.s_crm-crm-field  1         1    6%   35263     0        26     0       0       0 PU .
.    1816   23 wb_dept-user.wb_user          2         36  50%   13439     0        23     0       0       0 .
.     5     6 _Index._File/Index            0         0    0%   68480     0        11     0       0       0 PU .
.     2     6 _Field._File/Field            0         0    0%   68288     0         2     0       0       0 PU .
.    1196   121 prod-exp-loc-d.prod-exp-loc-d  4       147,490  70%    127     0         0     0       0       0 PU .
..... User IO Activity .....
.  Ustr# Tenant Name          PID    Flags Blk Ac v   OS Rd   OS Wr   Hit%  Rec Lck  Rec Wts  Line# Program Name .
.  >  773     0 trarm                    48139  SXB*   390366     0     0  100.00%  194492     0     582 crmim/apply.p .
.    778     0 xuycata2                  13742  SX     1411     0     0  99.99%     1     0     305 so/bmonaut1.p .
.    772     0 xc11wt2                   21791  SXB    1291     0     0  100.00%   169     0     -1  wm/senddata.p .
.    783     0 mlwt2                     6278   SXB    200     0     0  100.00%     0     0     -1  wm/senddata.p .

```


LOG-MANAGER

- A terribly underused but awesomely amazing tool
- Allows you to leave debug messages in your code
 - No more `/* Message here vValue. */`
- Create some secret hotkey sequence to activate
 - I.e. you can turn it on in production for one user
- Writes detailed info to a log file
- As easy as ...

```
assign log-manager:logfile-name      = "c:\temp\wshop.log"  
      log-manager:logging-level     = 3  
      log-manager:log-entry-types   = "4GLTrace,4GLTrans,QryInfo".
```

```
/* Writing your own messages is easy too ... */  
log-manager:write-message(  
    string(LogLevelEnum:WARN) + ': ' + pcMessage,  
    pcMessageGroup).
```

LOG-MANAGER

for each order where order-num > x:

Type: FOR Statement

Client Sort: N

Scrolling: N

Table: wshop.Order

Indexes: Order-Num

Query Statistics: Bad1 logmgr.p line 23

QueryId: 101299360

DB Blocks accessed:

wshop : 15599

DB Reads:

Table: wshop.Order : 4557

Index: Order.Order-Num : UNAVAILABLE

wshop.Order Table:

4GL Records : 3399

Records from server: 3399

Useful: 3399

Failed: 0

Select By Client: N

16K DB reads

4557 fragments

3399 records

LOG-MANAGER

for each order fields(order-num) where order-num > x:

Type: FOR Statement

Client Sort: N

Scrolling: N

Table: wshop.Order

Indexes: Order-Num

Query Statistics: Bad1 logmgr.p line 23

QueryId: 101299360

DB Blocks accessed:

wshop : 11333

DB Reads:

Table: wshop.Order : 3400

Index: Order.Order-Num : UNAVAILABLE

wshop.Order Table:

4GL Records : 3399

Records from server: 3399

Useful: 3399

Failed: 0

Select By Client: N

11K DB reads

3400 fragments

3399 records

for each order where terms = "net30" by terms:

Type: FOR Statement

Client Sort: Y

Scrolling: N

Table: wshop.Order

Indexes: Order-Num

Query Statistics: Bad3 logmgr.p line 45

QueryId: 35632288

Entries in result list: 15526

Time to build result list (ms): 169

DB Blocks accessed to build result list:

wshop : 252527

DB Reads to build result list:

Table: wshop.Order : 61300

Index: Order.Order-Num : UNAVAILABLE

wshop.Order Table:

Records from server: 15526

Useful: 15526

Failed: 0

Select By Client: N

Fields: Terms

Query Statistics: Bad3 logmgr.p

QueryId: 35632288

DB Blocks accessed:

wshop : 73703

DB Reads:

Table: wshop.Order : 20571

Index: Order.Order-Num : UNAVAILABLE

wshop.Order Table:

4GL Records: 15526

- Read all records (61K) PLUS another 20K fragments to return 15K records
- 320K total DB reads

- Unsupported and undocumented startup parameter
 - Aren't those the best!?!
- Writes detailed run-time index usage information to db.lg - yes db.lg
 - Do NOT use in prod please

- Tells you which index is used and how many fields deep

- Format is INDEX # LOWER-BOUND UPPER-BOUND TYPE

for each order:

- INDEX 20 0 0 (pu order-num)
- INDEX 37 1 1 (Ship-date + carrier)
- INDEX 36 2 2 EQUALITY (Carrier + ship-date)
- INDEX 35 3 3 EQUALITY (sales-rep+carrier+ship-date)
- **Now let's make it interesting**
- ... where ship-date=... and carrier=... and sales-rep GT ""
 - INDEX 36 2 2 EQUALITY (Carrier + ship-date)
- ... where sales-rep=... and carrier=... and ship-date LT ...
 - INDEX 35 2 3 (Sales-rep + Carrier + ship-date)
- ... where sales-rep =... and carrier NE... and ship-date LT ...
 - INDEX 23 1 1 EQUALITY (sales-rep)
- ... where sales-rep =... and carrier GT "" and ship-date LT ...
 - INDEX 35 2 1 (sales-rep + Carrier)

Network Effects

- Shared memory DB connections make programmers look like (geeky) rock stars
- Network connections are more like the morning after – not so pretty





Network Effects - Basics

MTU



-prefetch*



- Mm
- Mm
- Mm
- Mm
- Mm
- Mm



- Field lists: fields of record that are sent to the client
- Prefetch: multiple records per message (no-lock)
- Message: Unit of measure for 4GL network data transfers
 - Size controlled by –Mm database start-up parameter
- MTU (maximum transmission unit): the largest packet size that can be transmitted over a network
 - This is a network parameter, not a Progress parameter
- Server Parameters
 - -Mi / -Ma / -Mpb: min, max users per server and number of servers per broker
 - 4GL servers are round-robin single-threaded
 - SQL servers are multi-threaded

Network Effects - LOBS

- Thank <*insert favourite deity*> Progress doesn't send the LOB across the wire unless you ask for it
- The LOB field is really a separate entity to the record
- The *real* record only contains a pointer to the LOB
- The LOB may be in another storage area (and should be)
- When you access the LOB, the client requests it from the server



Network Effects – Legacy Parameters

- -Mm: message buffer size
 - Default 1024 is too small
 - Max 32600 doesn't seem to be warranted
 - 8192 is a nice sweet spot
- -Mi / -Ma: min/max users per server
 - The more users per server, the less time the server can dedicate to any one user
 - -Mi 1 -Ma 5 is a good start
 - -Mi 1 -Ma 1 if you have heavy duty users
 - AppServer agents

Network Effects – New Prefetch Parameters

- No-lock queries
- Forward only or scrolling
- 10.2B06 + and 11.1+

- -prefetchPriority: server defers poll for other requests while filling message
 - Current suggested value 100 records added before next poll
- -prefetchDelay: Fills first message. By default first message contains one record
 - In theory this is better. In practice the ms difference is not significant
- -prefetchNumRecs : How many records are stuffed in a message
 - 100 records is a good start (default is 16)
- -prefetchFactor: How full (%-wise) to fill a message
 - 90-100%

Maximum Transmission Unit (MTU)

- A network parameter set at the NIC level
- Enable on the routing infrastructure
- Default is 1500
- “Jumbo Frames” is typically 9000 bytes

- The advantage lies in the relative size of network header data
 - 1500 byte MTU: 1460 byte payload / 1538 byte total = 95% efficient
 - 9000 byte MTU: 8960 byte payload / 9038 byte total = 99% efficient

Monitoring Server Messages

- `_ActServer VST`
 - Key: `_server-ID`
 - Interesting fields: `_Server-ByteSent`, `_ServerMsgSent`, `_ServerRecSent`
 - Calculate send size (Bytes sent / messages sent) and compare to `-Mm`
- Bug: When a record is larger than `-Mm`, only the first msg is counted
 - I.e. if you send a 4K record and `-Mm` is 1024, only 1 msg and 1024 bytes sent recorded
 - Blobs sent in 32,000 byte chunks – each chunk increments `msgSent` by 1 and `byteSent` by `-Mm`
- ProTop Free:

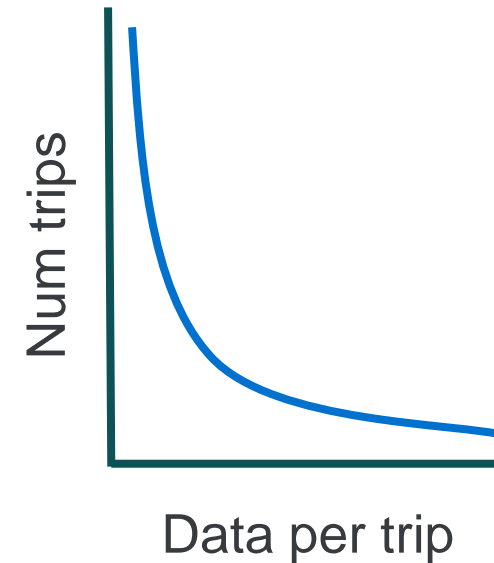
Server Activity																
Srv	Type	Port	Cnx	Max	QryRcvd	RecRcvd	MsgRcvd	rr/msg	RecSent	MsgSent	rs/msg	MB Sent	v	MB Rcvd	RcvdSz	SendSz
9999	Total	0	1	3	0	0	42	0.00	0	42	0.00	0.04		0.00	116	1020
1	Auto	3000	1	1	0	0	42	0.00	0	42	0.00	0.04		0.00	116	1020
4	Inact	0	0	0	0	0	0	0.00	0	0	0.00	0.00		0.00	0	0
3	Inact	0	0	0	0	0	0	0.00	0	0	0.00	0.00		0.00	0	0
2	Auto	3001	0	1	0	0	0	0.00	0	0	0.00	0.00		0.00	0	0

unnecessary work

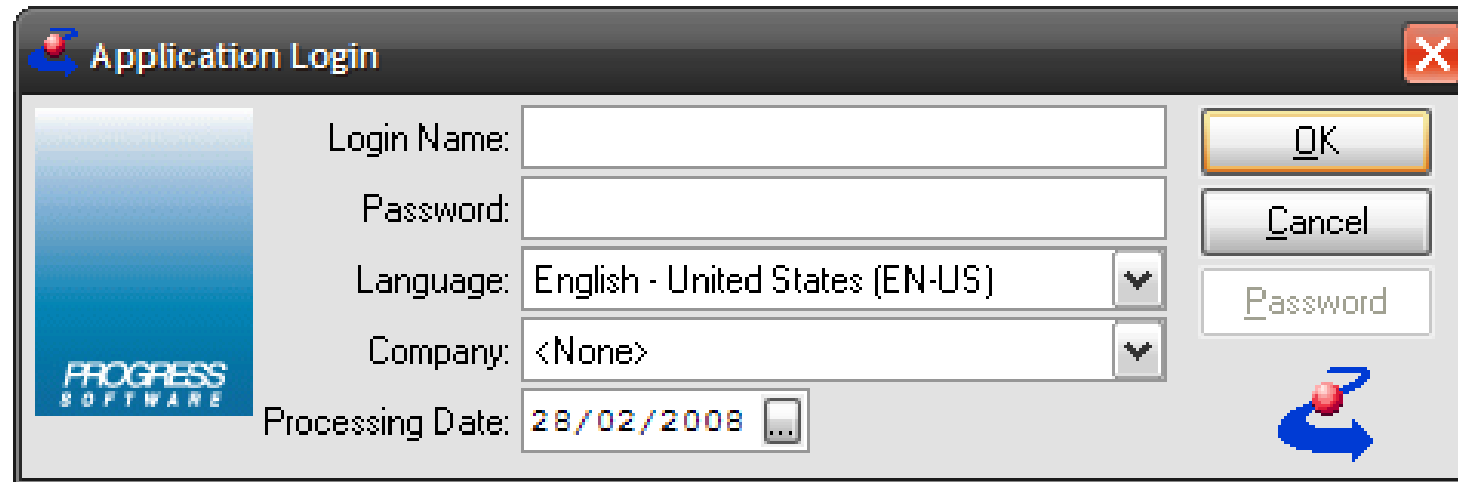
1. Too many calls
2. Too much stuff per call
3. Too many copies of data

data across the network

- Network typically major bottleneck
 - Makes other performance problems worse
- Number of roundtrips
 - Making a server connection has cost
- Data volume per roundtrip
- Network topography has impact ...
 - ... but usually out of our control



optimise network roundtrips



`loginWindow.w`

```
run getLoginLanguages()
```

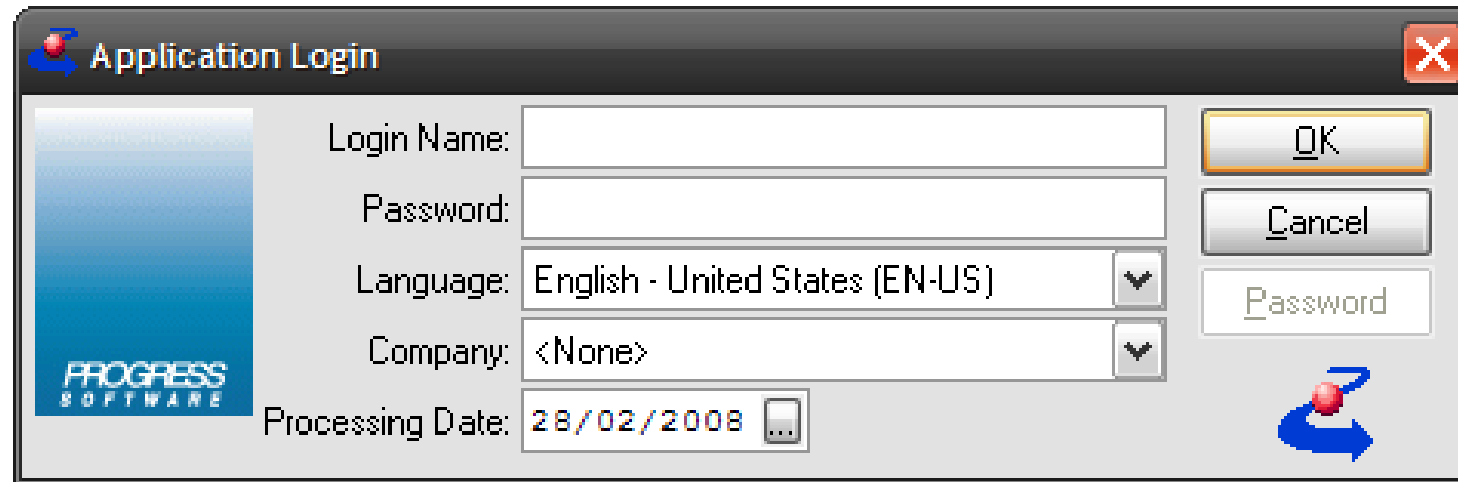
```
run getLoginCompanies()
```

```
run buildUI().
```

```
getLoginLanguages()
```

```
getLoginCompanies()
```

optimise network roundtrips



loginWindow.w

① run getLoginLanguages()

run getLoginCompanies()

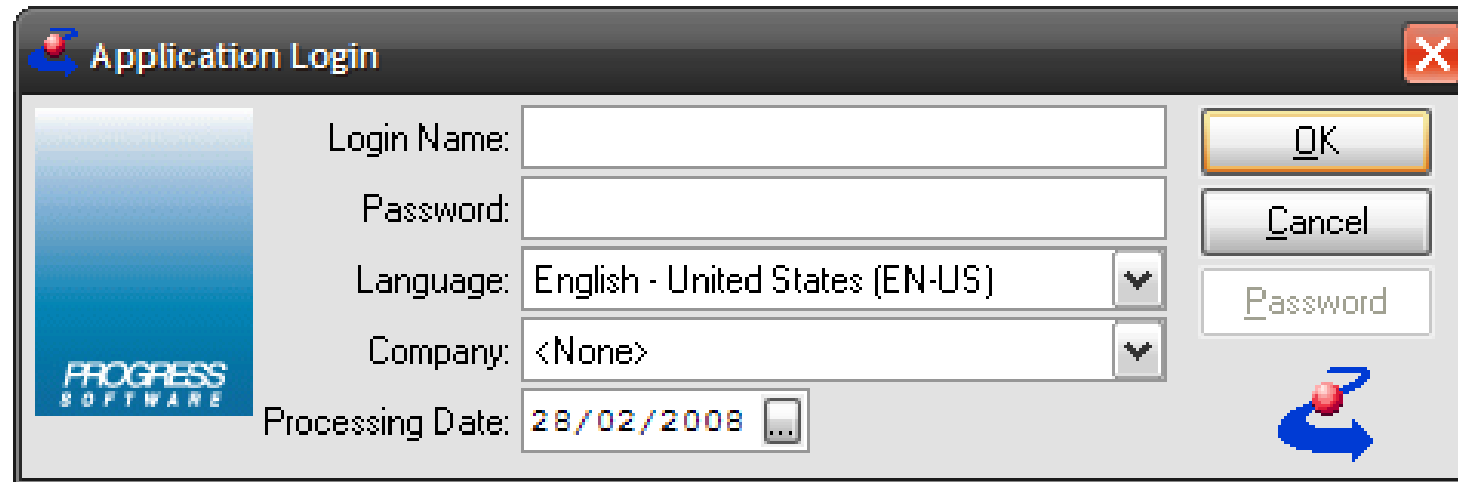
run buildUI().



getLoginLanguages()

getLoginCompanies()

optimise network roundtrips



loginWindow.w

```
run getLoginLanguages()
```

```
2 run getLoginCompanies()
```

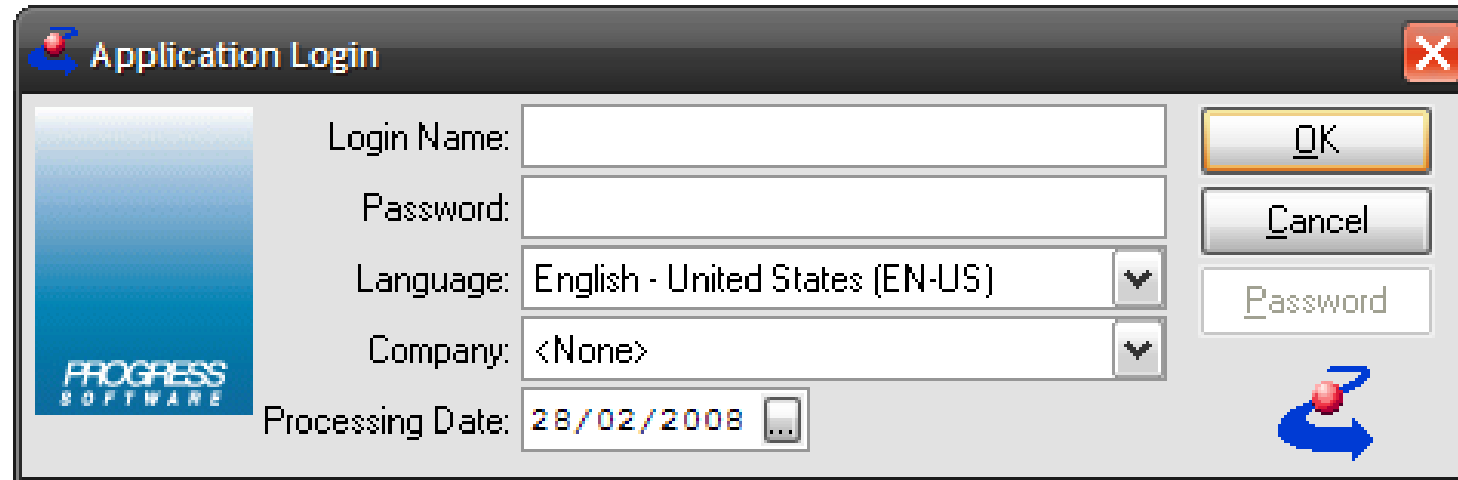
```
run buildUI().
```



```
getLoginLanguages()
```

```
getLoginCompanies()
```

optimise network roundtrips



loginWindow.w

1 run getUIData()

run buildUI().

getUiData()

run getLoginLanguages()

run getLoginCompanies()

no deep copies

```
define temp-table ttData ...  
  
run populateData (output table ttData).  
run showData (input table ttData).  
run getChanges (output table ttData).  
run saveChanges (input table ttData).
```

```
define temp-table ttData ...  
h = buffer ttData:handle.  
  
run populateData (output h).  
run showData (input h).  
run getChanges (output h).  
run saveChanges (input h).
```


no deep copies

```
define temp-table ttData ...  
  
run populateData (output table ttData).  
run showData (input table ttData).  
run getChanges (output table ttData).  
run saveChanges (input table ttData).
```

```
define temp-table ttData ...  
  
run populateData (output table ttData by-reference).  
run showData (input table ttData by-reference).  
run getChanges (output table ttData by-reference).  
run saveChanges (input table ttData by-reference).
```

dataset-handles are not really handles

- Yes, you define HANDLE variables to work with them
- But you can pass / call them as datasets
 - To a parameter defined as DATASET-HANDLE
 - To a parameter defined as DATASET
 - To a parameter defined as HANDLE
- Nice thing with DATASET-HANDLE is that if you receive them you can use static code against them as if you got a 'real' dataset

dataset-handles are not really handles

```
procedure fetch_data:
  def output param poLotsaData as JsonObject.

  def var hDataset as handle.
  def var oDataObject as Object.

  oDataObject = new CustomerData().
  oDataObject:GetData(
    output dataset-handle hDataset).

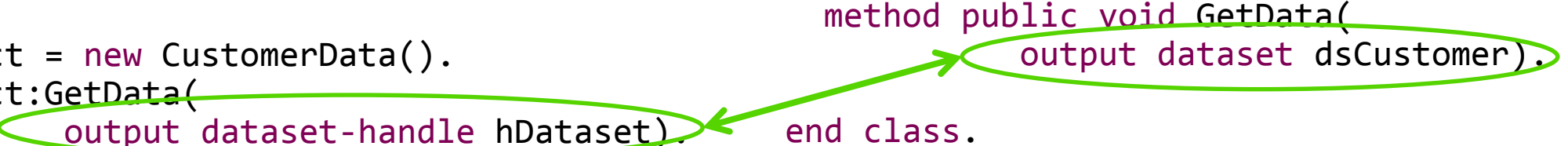
  poLotsaData:Read(hDataset).
  /* now has a property called dsCustomer */

  oDataObject = new EmployeeData().
  oDataObject:GetData(
    output dataset-handle hDataset).

  poLotsaData:Read(hDataset).
  /* now has a property called dsCustomer
  AND one called dsEmployee */
end procedure.
```

```
class CustomerData:
  define dataset dsCustomer for
  ttCustomer, ...

  method public void GetData(
    output dataset dsCustomer).
end class.
```



```
class EmployeeData:
  define dataset dsEmployee for
  ttEmployee, ttDepartment, ...

  method public void GetData(
    output dataset dsEmployee).
end class.
```

finding deep copies: find it

- Make sure you have a problem
 - LOG-MANAGER:LOG-ENTRY-TYPES = 'Temp-tables'
 - Data structure reference counts – datasets, buffers, objects, procedures, etc

```
define variable hDS as handle no-undo.  
define variable iLoop as integer no-undo.
```

```
hDS = session:first-dataset.  
do while valid-handle(hDS):  
  assign iCnt = iCnt + 1.  
  hDS = hDS:next-sibling.  
end.
```

```
TEMP-TABLE      Created TEMP-TABLE Headers (ID:4 NO-UNDO Indexes:1) OpenEdge.Ne  
TEMP-TABLE      Created TEMP-TABLE ConfigOption (ID:5 NO-UNDO Indexes:1) OpenEd  
TEMP-TABLE      Created TEMP-TABLE Registry (ID:6 NO-UNDO Indexes:1) OpenEdge.N  
TEMP-TABLE      Created TEMP-TABLE HeaderParameter (ID:7 NO-UNDO Indexes:1) Ope
```

finding deep copies: fix it

- Make value passing of tables/dataset the exception
 - Look for ANY and ALL temp-table, table-handle, dataset, dataset-handle calls without BY-REFERENCE

```
run get_data.p (  
    output dataset-handle hDataset  
    by-reference).
```

- When you have to make a deep copy, clean up after yourself

```
run get_data.p (  
    output dataset-handle hDataset).  
finally:  
    delete object hDataset.  
end finally.
```

- Desperate measures may be needed

```
hDS = session:first-dataset.  
do while valid-handle(hDS):  
    delete object hDS.  
    hDS = session:first-dataset.  
end.
```

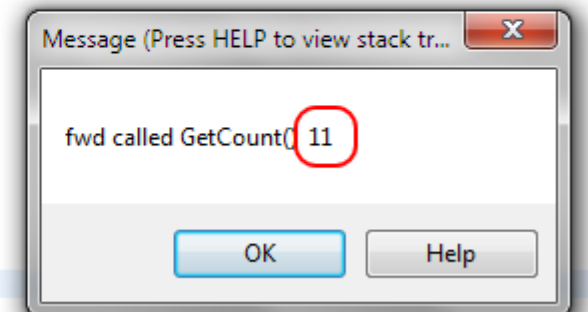
too many calls

- Make sure you evaluate functions only once

do i = 1 to udf() vs. do i = udf() to 1 by -1

- Functions are sadly not always idempotent
- Use arrays instead of delimited strings
- Stuff inside a loop
 - Record FINDs
 - RUN (INPUT-OUTPUT tt)

```
function GetCount returns integer (pcDirection as char, piCnt as int):  
  message pcDirection 'called GetCount() ' piCnt  
  view-as alert-box.  
  
  return 10.  
end function.  
  
def var iLoop as int.  
  
do iLoop = 1 to GetCount('fwd', iLoop) :  
end.  
  
do iLoop = GetCount('back', iLoop) to 1 by -1 :  
end.
```



- There are ~~no~~ few wizards. There is ~~no~~-barely any magic.
- Measurement is the only sure way to KNOW
 - That you have a problem
 - That you fixed the problem

PUGCHALLENGE  **EXCHANGE**
AMERICAS

- Unsupported tool introduced in V9
- Now supported in PDSOE 11.6 (maybe 11.5 ?)
- Example UI available in \$DLC/src/samples
 - I just wrote some simple ChUI/text code to do it myself
- Use the PROFILER handle
 - Ex.: PROFILER:ENABLED = TRUE.
- Outputs time spent in each line of code, number of executions...
- Sorry – no time to go into Profiler in detail

1 Essai #1

Date: 12/12/2007

Time: 19:24:14

Total Run Time: 12.4751

Compare

Code Block	Calls To	Avg Time	Tot Time	% Session	Cum Time	Calling Code Block	Calls From	% Session
report.p	1	11.715508	11.715508	93.911398	12.421169	ProfileCode PROFILER/profile.w	2	0.419942
cummulventes report.p	9,054	0.000065	0.584186	4.682829	0.584186			
ListTop5 report.p	9	0.013497	0.121475	0.973742	0.121475			
ProfileCode PROFILER/profile.w	2	0.026194	0.052388	0.419942	0.000000			
setProfilerOptions PROFILER/profile.w	1	0.000721	0.000721	0.005780	0.000721			
profiler_message PROFILER/profile.w	1	0.000249	0.000249	0.001996	0.000249			
assignFieldValues PROFILER/profile.w	1	0.000206	0.000206	0.001651	0.000206			
3-USER-INTERFACE-TRIGGER PRO	2	0.000037	0.000074	0.000593	0.000000			

Line	Exec Count	Avg Exec	Tot Time	Cum Time	Score#1	Score#2	Database	Table	FileName
31	3,674	0.002702	9.926014	9.926014	?	??			
32	11,611	0.000116	1.345663	1.345663	?	??			
25	1,118	0.000154	0.172549	0.172549	?	??			
0	1	0.115074	0.115074	12.421169	?	??			
33	9,054	0.000009	0.081806	0.665992	?	??			
34	9,054	0.000002	0.017669	0.017669	?	??			

```

27 IF FIRST-OF(country) THEN DO:
28     EMPTY TEMP-TABLE ttSales.
29 END.
30
31 FOR EACH order OF customer WHERE YEAR(order.orderdate) = cYear USE-INDEX orderdate:
32     FOR EACH orderline where orderline.ordernum = order.ordernum:
33         RUN cummulVentes.
    
```

- This is not a "Learn how to use indexes 101" workshop
- Go see *Proper Care and Feeding of an Index* by Mike Lonski

- With that said, here is a simplified version of the rules
 1. If there is a "CONTAINS" then use a word-index
 2. If an index is unique and all of it's components are used in an equality match, use that index
 3. Use the index with the most equality matches on SUCCESSIVE, LEADING INDEX components
 4. Use the index with the most active range matches on SUCCESSIVE, LEADING INDEX components
 5. Use the index with the most active sort matches
 6. Use the first index alphabetically
 7. Use the primary index

Sidebar – Multi-Index Use

- Where ... and ...
 - All components of each index involved in equality matches
 - No unique indexes
- Where ... or ...
 - Both sides of OR contain at least the lead component of an index
 - Equality or range matches

Sidebar - 4GL vs SQL = Rules vs Cost

- 4GL and SQL are two different animals
- The 4GL compiler uses rules to pick an index
- The SQL analyzer uses cost statistics to select the lowest cost path to the data
 - You need to run UPDATE STATISTICS
- CAVEAT: update statistics is buggy in earlier versions
 - Search the KB and the release notes
 - I have personally noticed issues up to 10.2B03
- TEST TEST TEST