YBS GROUP

YBS
ORACLE FORMS
APPLICATION STRATEGY
IN A SOA WORLD

Created by: Graham Brown, Application Architecture Manager

AGENDA

- Background to Yorkshire Building Society
- History of YBS Oracle Forms Application
- Strategic Roadmap
- Analysing the application using PITSS
- The journey to SOA supported by PITSS
- Next steps for YBS

BACKGROUND TO YORKSHIRE BUILDING SOCIETY

YORKSHIRE BUILDING SOCIETY

- Financial Services organisation – Mortgages and Savings our key business
- Strong Mutuality agenda – everything we do is for the benefit of our members
- Between 2008 and 2012, we went through significant Merger and Acquisition activity
- YBS Group now contains multiple brands..
- 3.5m Customers
- £37.6bn Assets
- Over 4,500 Employees

HISTORY OF YBS ORACLE FORMS APPLICATION

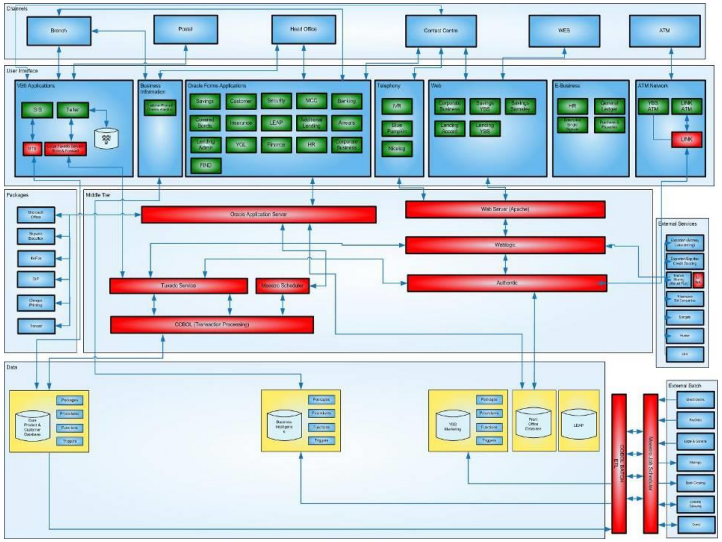
FORMS APPLICATION - HISTORY

- Initially developed during the late 90's – driven by the need to move from a mainframe to overcome Y2K issues
- Developed using Oracle Forms 4.5 Client/Server – initial module definitions generated from Oracle Designer 2000
- Migrated through versions – currently 10gR2 - plans for 11gR2 this year
- Mid-tier forms – adopted for business rather than technology reasons
- Evolved into a large application – 000's of modules
- Large supporting database – again, 000's objects
- Application is not just Oracle Forms – Also utilise lots of COBOL (overnight processing), Java (Web channel), MS VB/.Net (Branch channel)

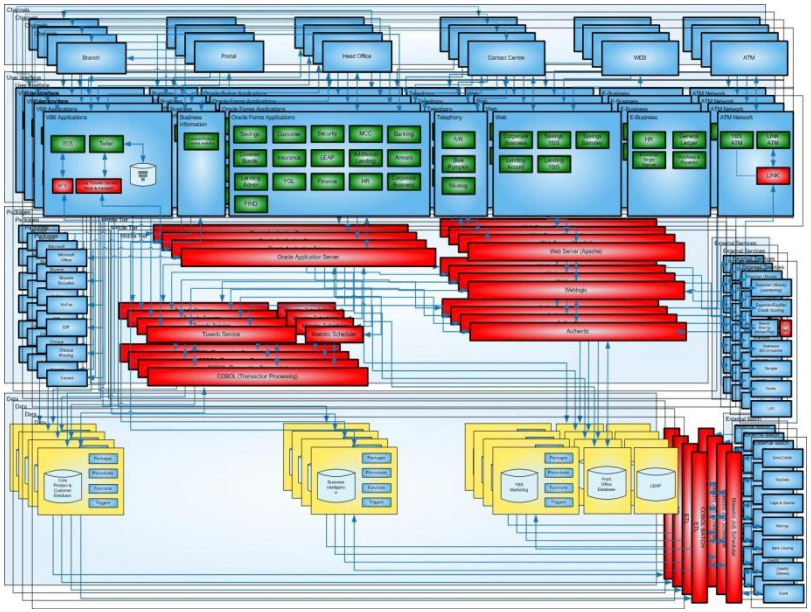
STRATEGIC ROADMAP

BRAND CLONING

- Single Brand architecture



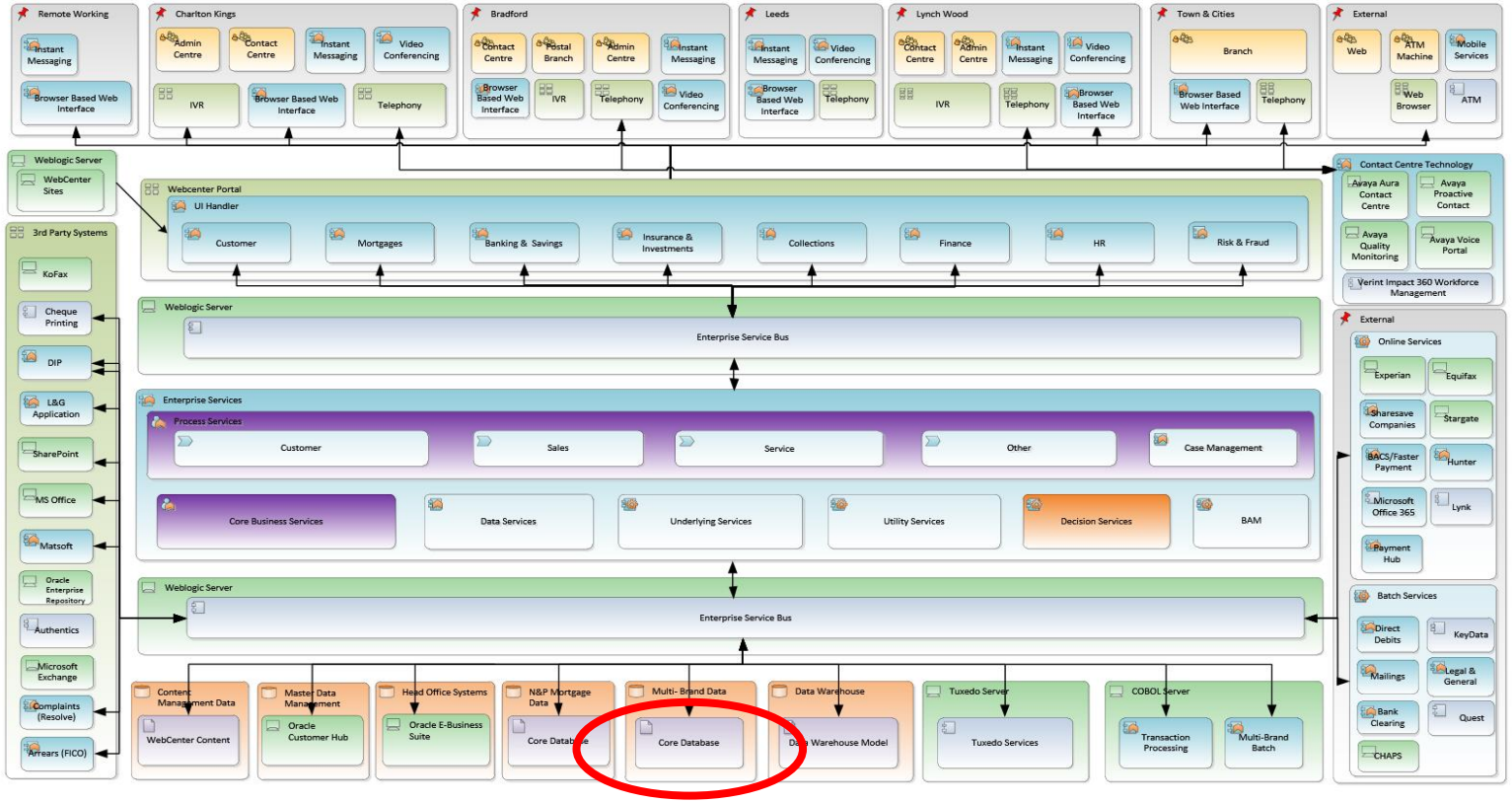
- Multi – Brand cloned architecture



- Our architecture was starting to constrain our business agility!

STRATEGIC ROADMAP

- Organisational growth aspirations - £50bn assets 5 million customers within 5 years
- Re-engineering not just technology but Business and associated processes
- Future state architecture – SOA and BPM. Improve IT agility to satisfy business requirements



- Centred around a single, brand aware database
- No appetite to do a wholesale conversion of our Forms to ADF (yesterdays application in todays technology)

WHY PITSS?

We faced a number of challenges to be able to deliver our strategy:

- Declining pot of Subject Matter Experts – both Technical & Business who understand the existing application functionality
- Minimal system documentation for the existing applications – it has evolved over the last 18 years
- Needed a mechanism for identifying business logic and enabling informed decisions to be made to help move us in the strategic direction
- Choice of doing this manually – slow, resource heavy, and possibly inaccurate

or we look for tooling to assist

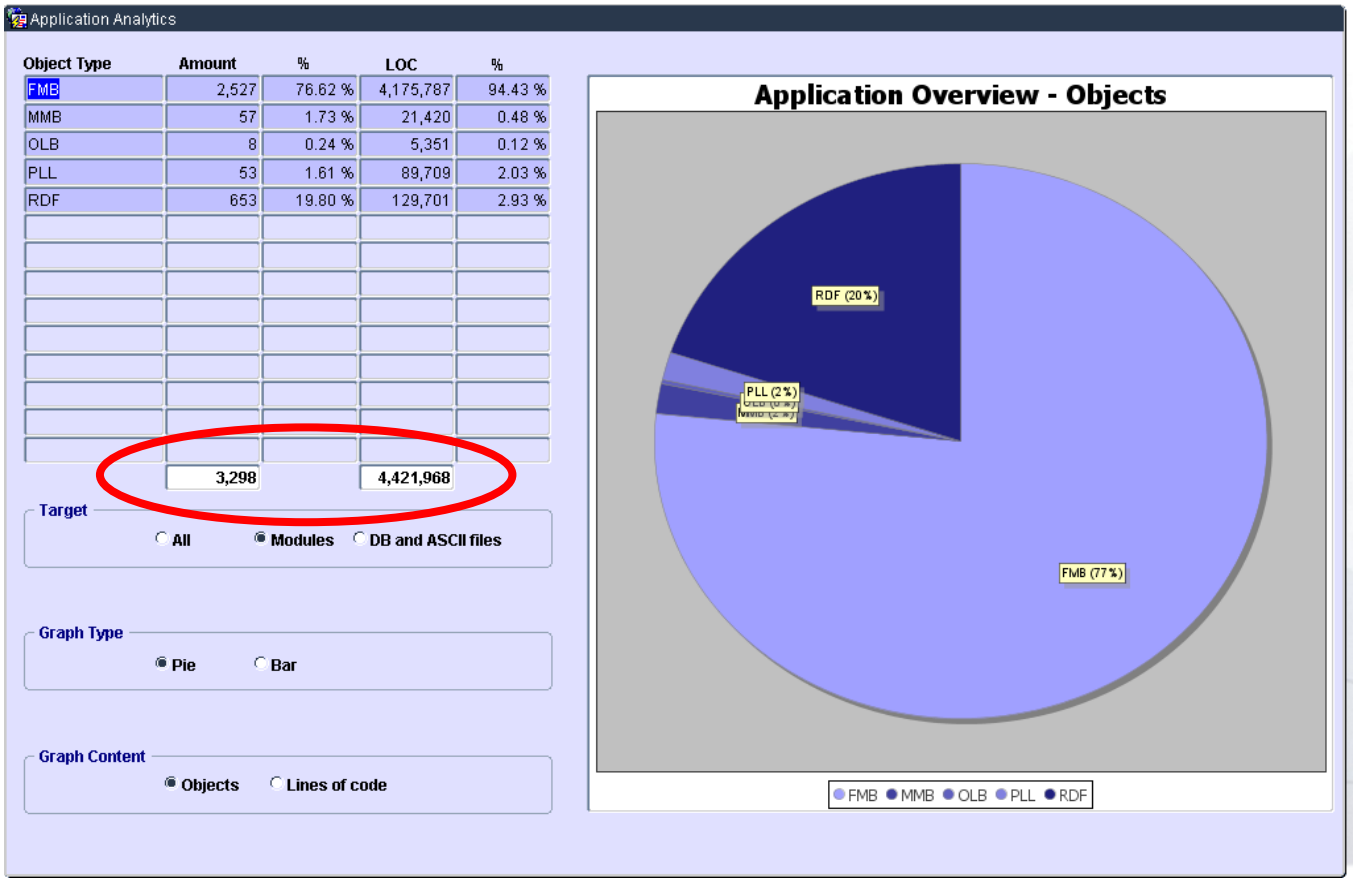
- A small number of tools out there, but PITSS was the only tool we identified which worked at the code level we required
- We ran a proof of Concept activity to evaluate PITSS in late 2012. It worked for Forms, Reports and Database
- COBOL could be loaded in and allowed simple textual search but didn't provide the full impact analysis we required
- With an assurance a COBOL parser could be provided, we purchased a number of PITSS modules in Mid 2013
- Modules purchased – Technology Base, Maintenance/Development, Application Analysis, Application Engineering and Source Code Analytics

ANALYSING THE APPLICATION

PITSS SOURCE CODE ANALYTICS

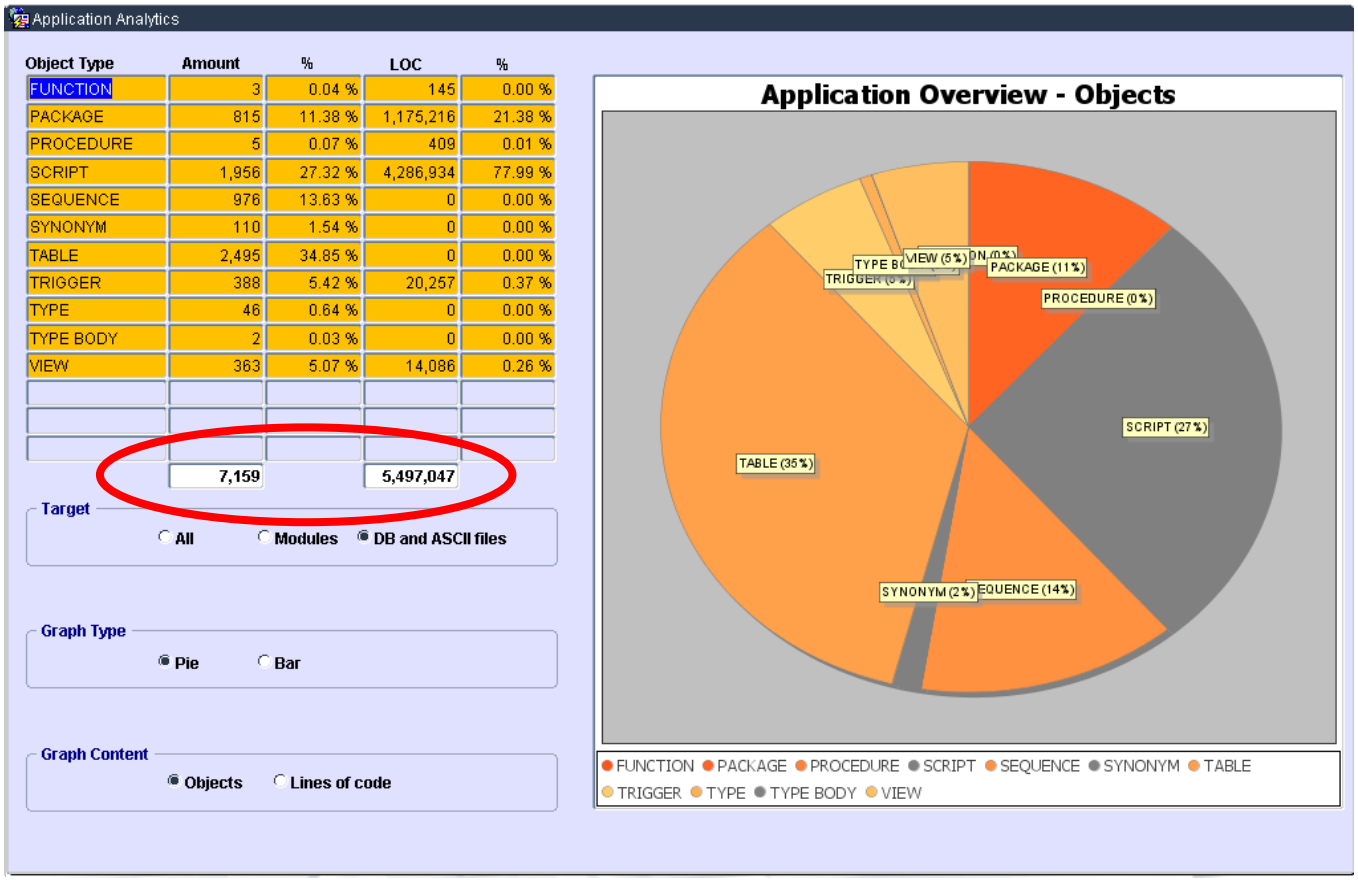
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- We know we have a large application, but now we can quantify it…



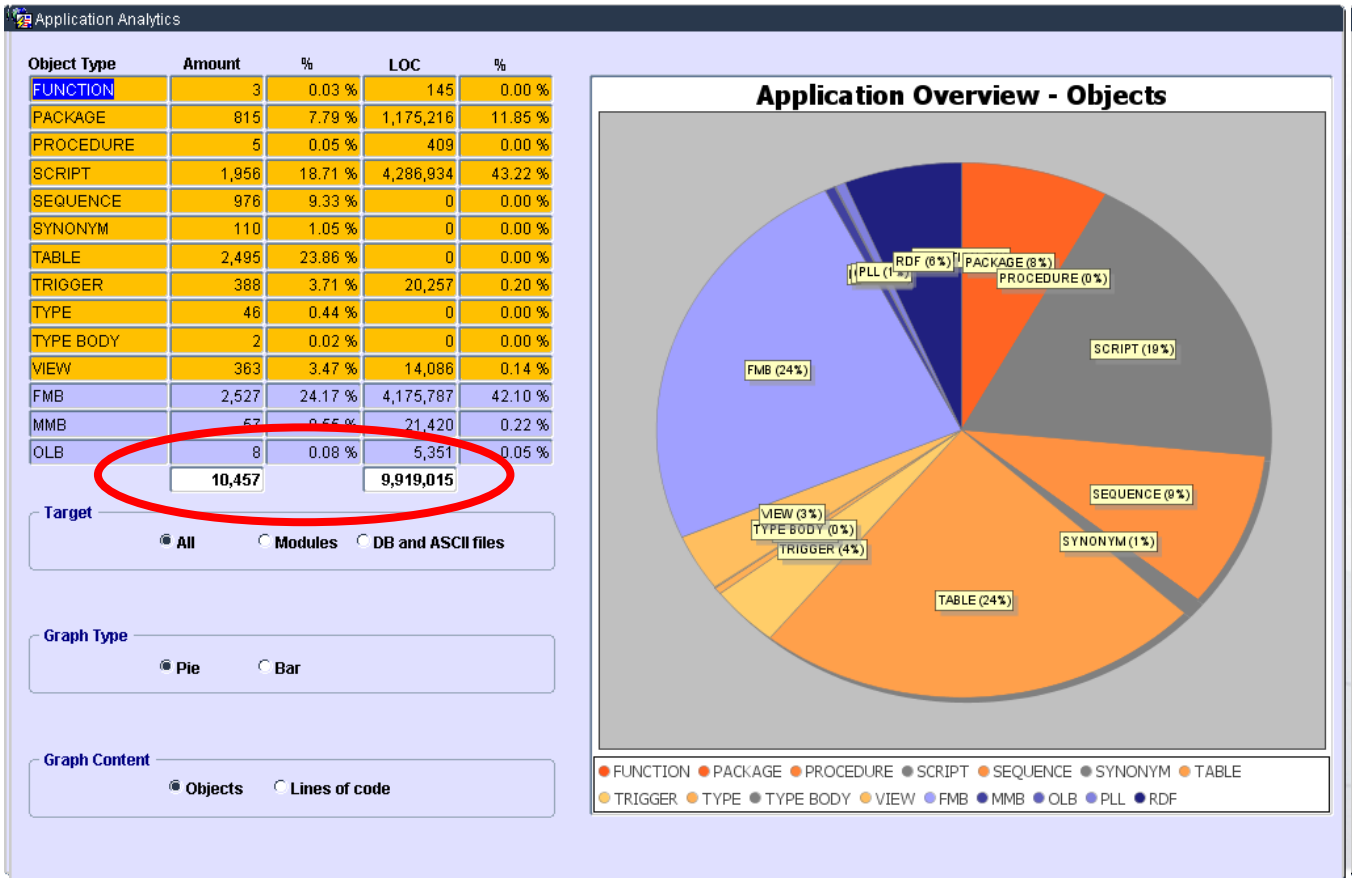
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- We know we have a large application, but now we can quantify it…



WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- We know we have a large application, but now we can quantify it…



WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- We know we have a large application, but now we can quantify it…

Application Analytics

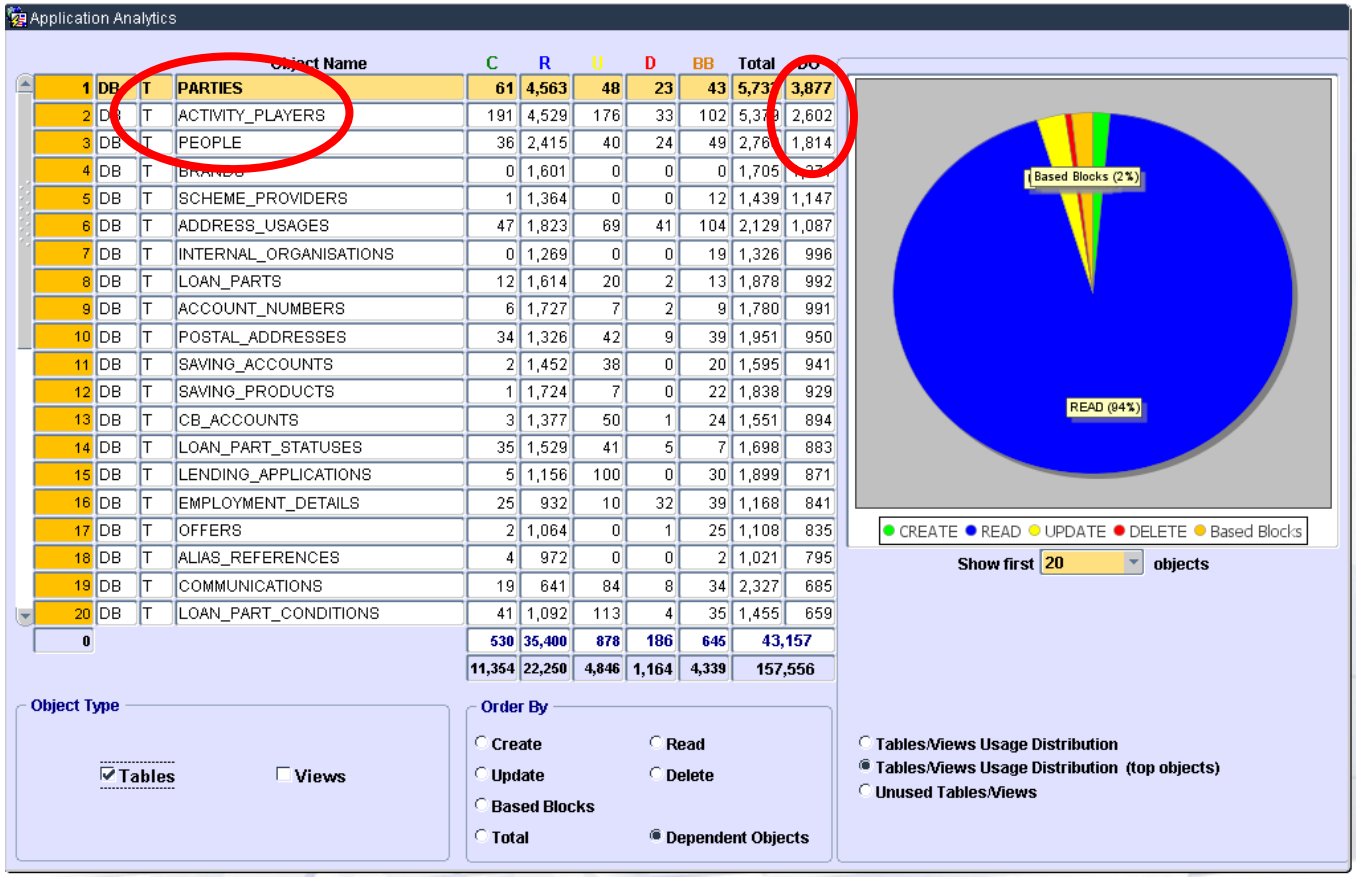
Object Type	LOC	Empty Lines	Empty Lines Ratio	Commented Lines	Comment Ratio	Med.Line Size Source Code (Chars)	(Mb)	PU	Lines/PU	PU ***	Lines/PU ***
FUNCTION	145	29	20.00 %	49	33.79 %	42	0.006	3	48	3	48
PACKAGE	107,553	6,142	5.71 %	21,280	19.79 %	35	3.769	815	132	8,776	12
PACKAGE BODY	1,067,663	105,117	9.85 %	259,479	24.30 %	40	42.765	813	1,313	9,083	118
PROCEDURE	409	12	2.93 %	112	27.38 %	40	0.016	5	82	5	82
SCRIPT	4,286,934	17,425	0.41 %	15,936	0.37 %	39	167.417	1,956	2,192	1,956	2,192
TRIGGER	20,257	2,229	11.00 %	2,961	14.62 %	38	0.771	388	52	388	52
VIEW	14,086	179	1.27 %	1,236	8.77 %	45	0.635	363	39	363	39
FMB	4,175,787	539,936	12.93 %	1,043,575	24.99 %	34	142.170	210,311	20	212,766	20
MMB	21,420	2,721	12.70 %	4,622	21.58 %	33	0.708	4,194	5	4,194	5
OLB	5,351	585	10.93 %	0	0.00 %	35	0.188	449	12	449	12
PLL	89,709	11,248	12.54 %	14,636	16.31 %	38	3.414	831	108	3,696	24
	9,919,015	**703,661**	**7.09 %**	**1,382,344**	**13.94 %**	**34**	**365.754**	**230,365**	**43**	**251,920**	**39**

Target: (•) All ( ) Modules ( ) DB and ASCII files

Graph Content:
(•) Medium Line Size (Chars)
( ) Source Code Size (Mb) - Pie
( ) Source Code Size (Mb) - Bar
( ) PU medium size (lines)
( ) PU medium size (lines) ***

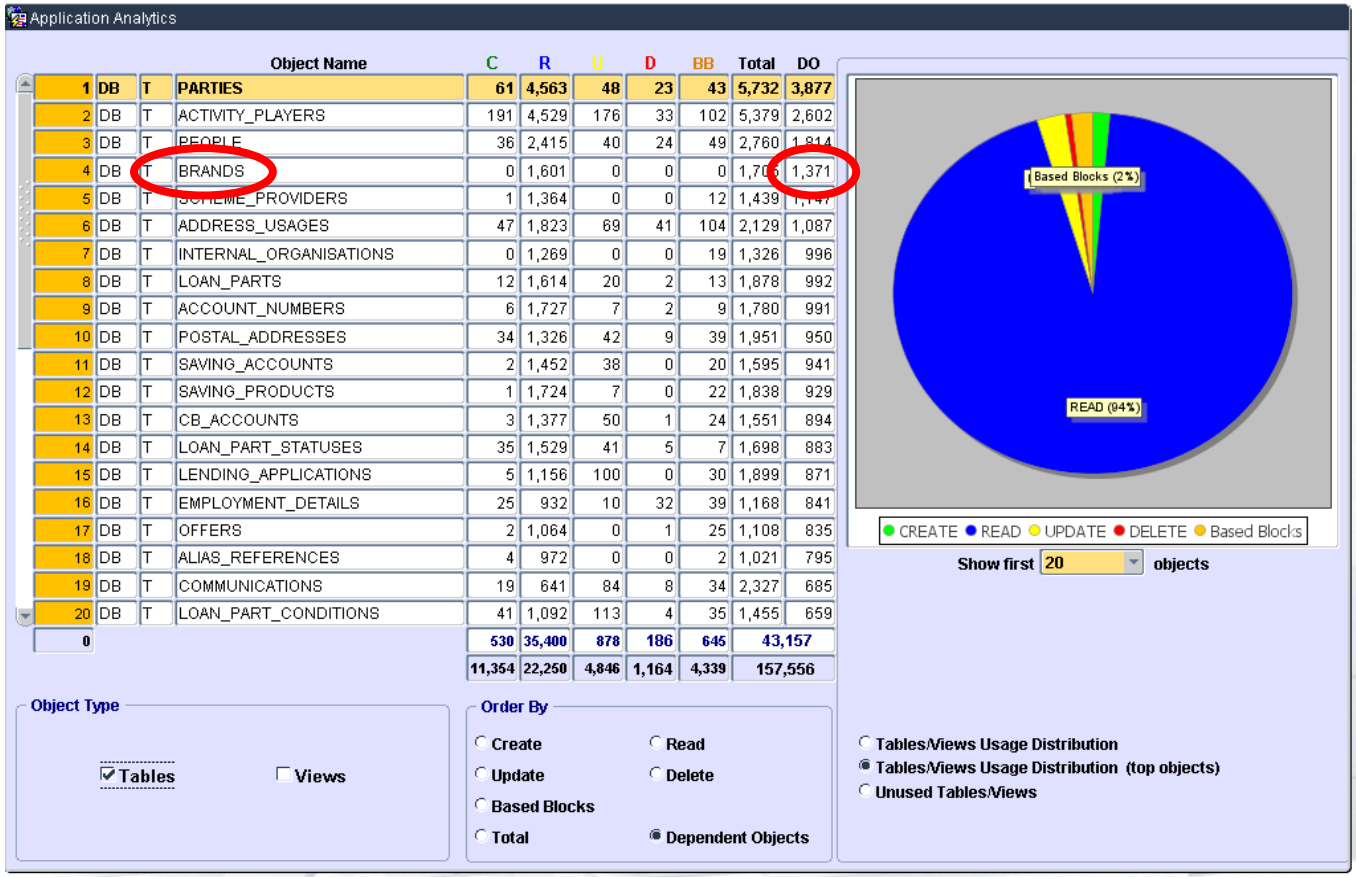
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- So, we have a complex application. But how complex is it?



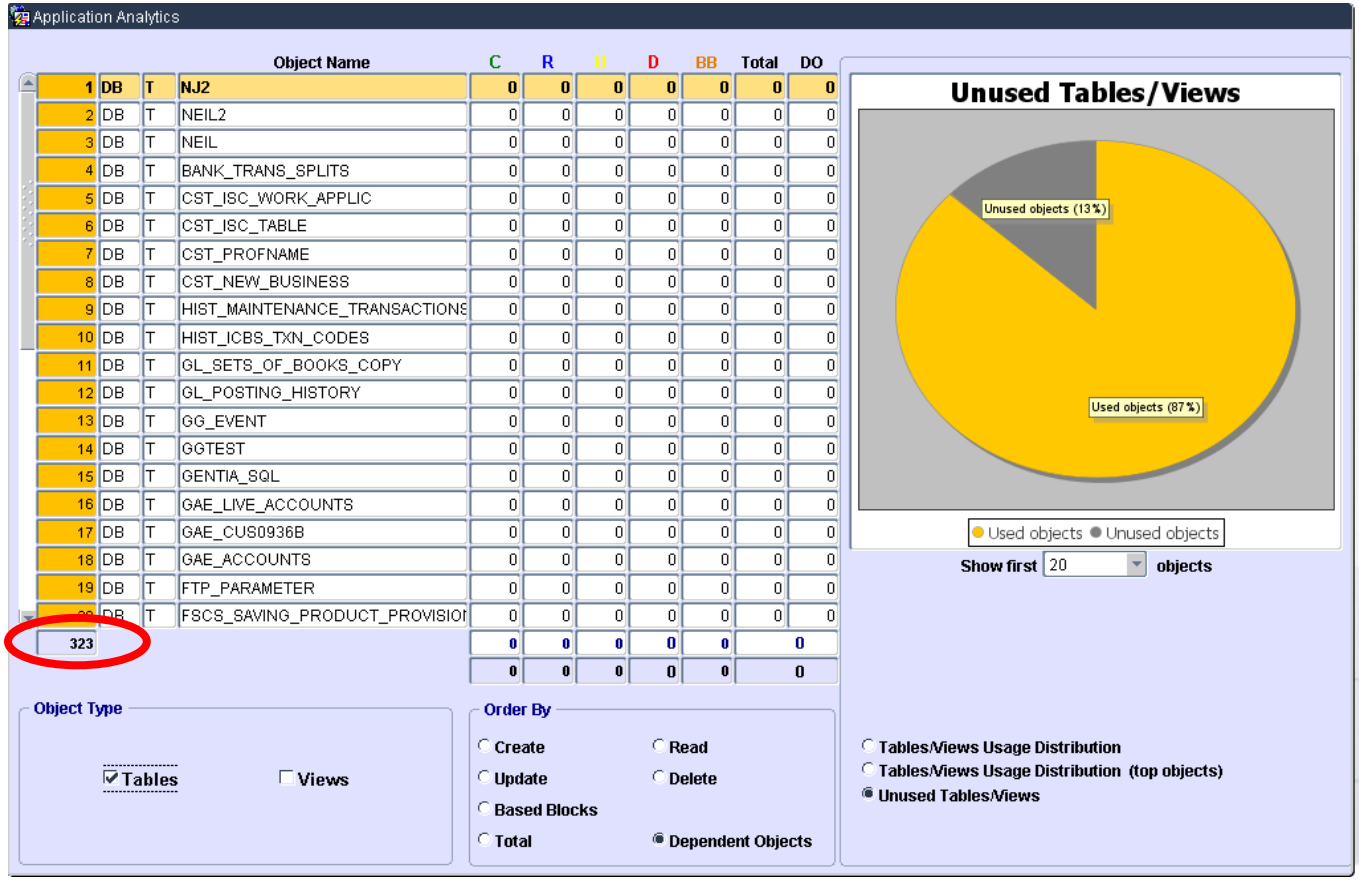
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- We initially thought we would only need to brand a handful of tables!



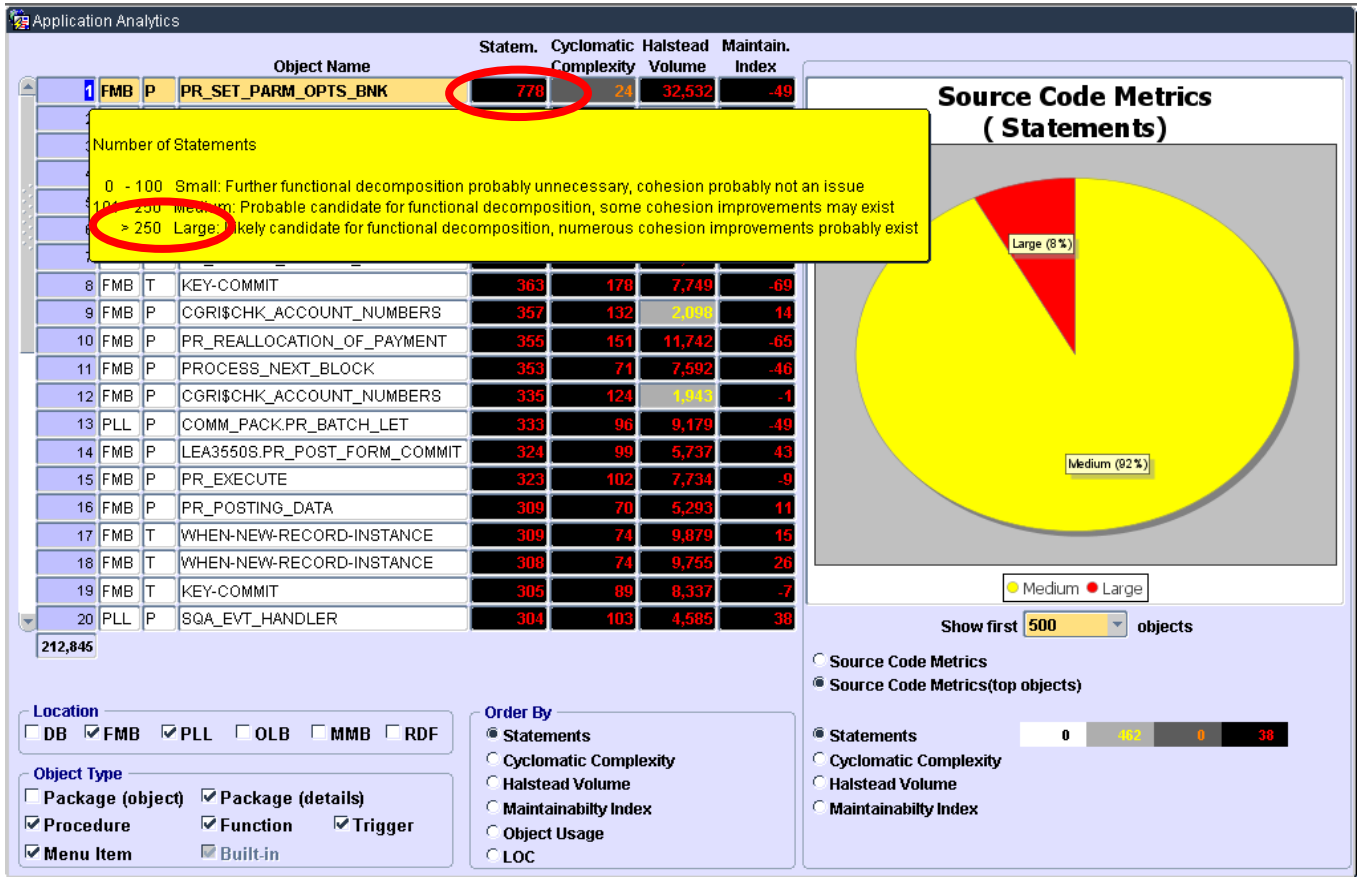
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- We were surprised to find we have a number of unused tables!



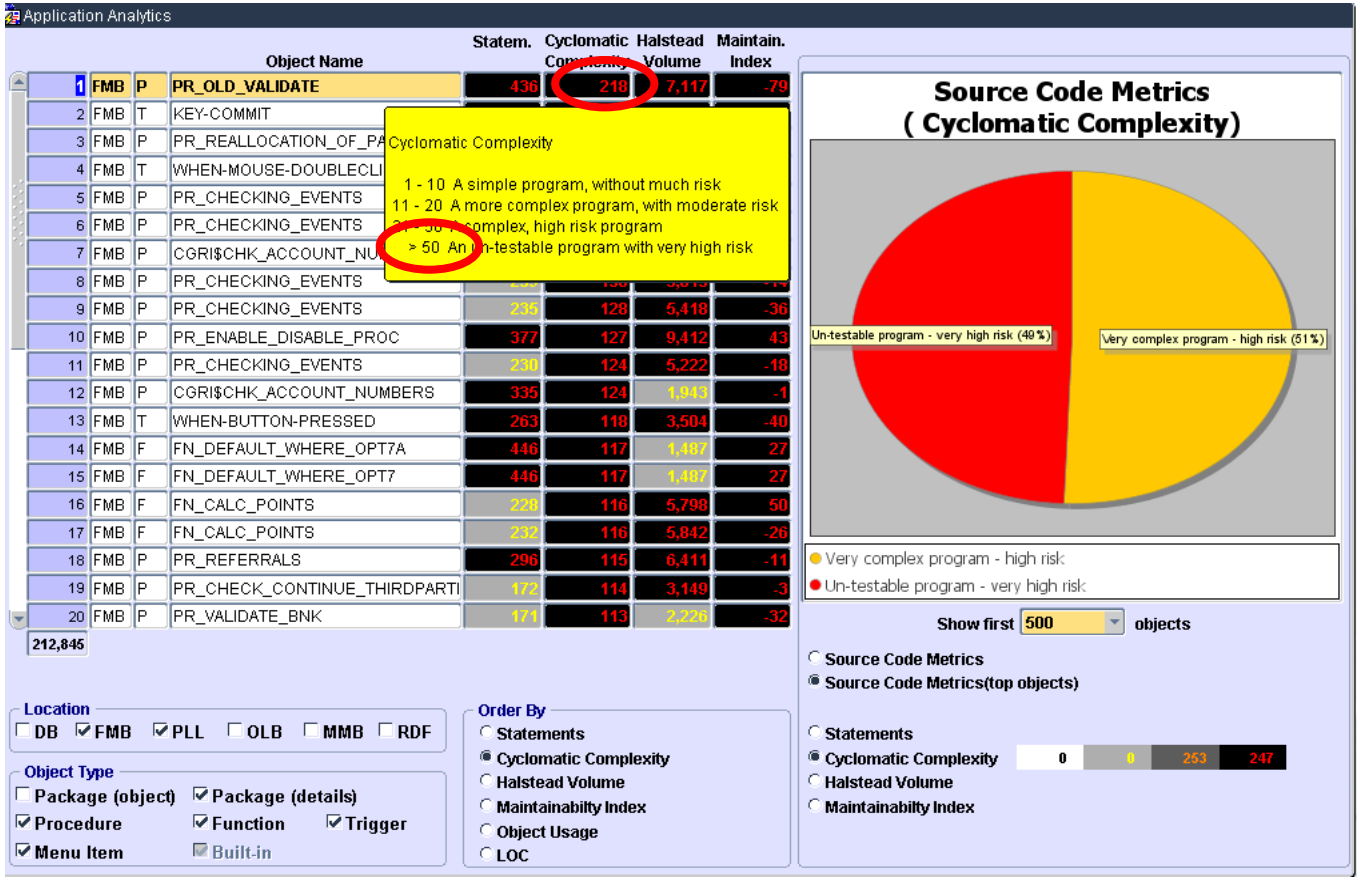
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- Source Code Metrics – the number of Statements within a module



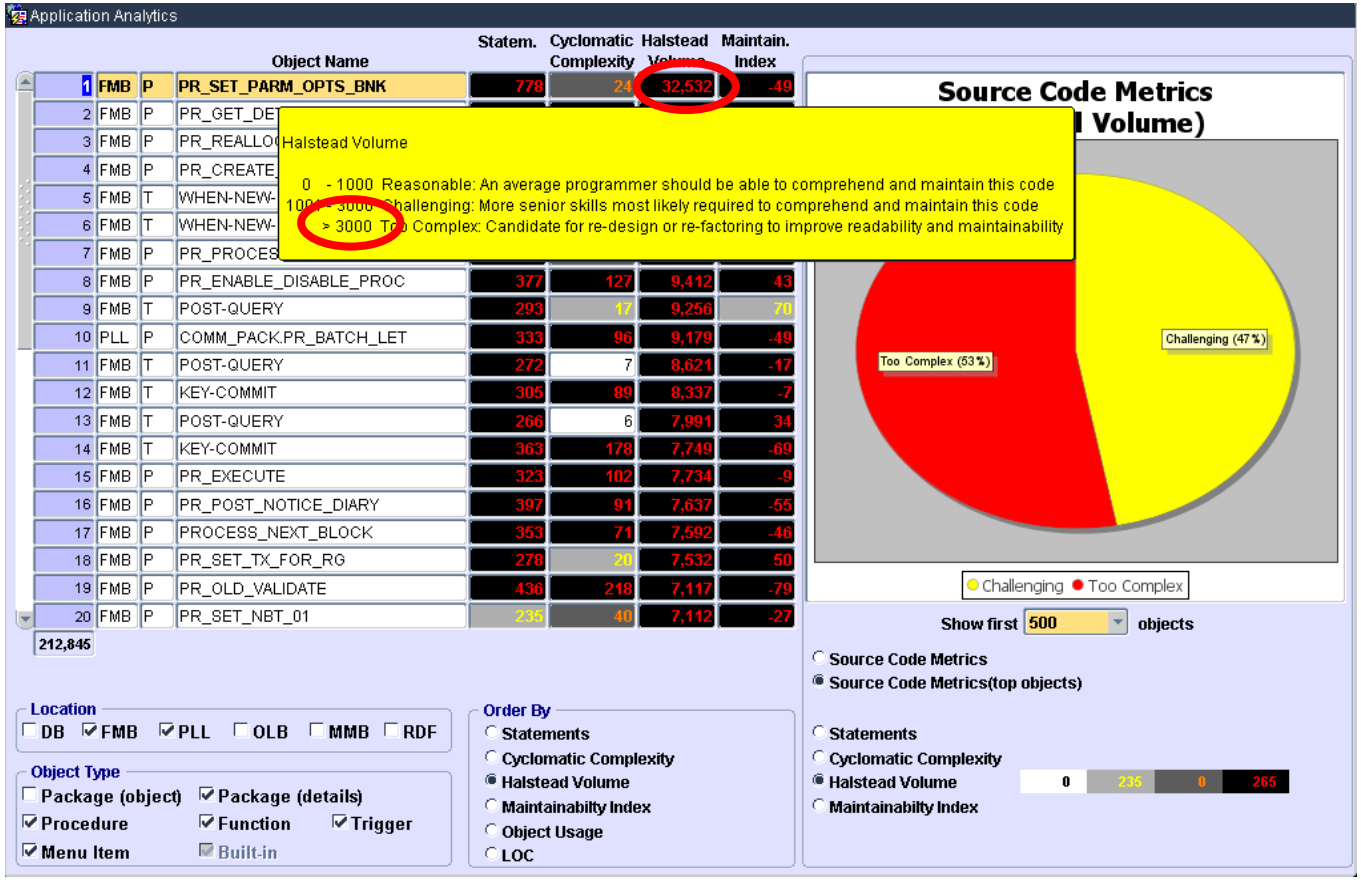
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- Cyclomatic Complexity – How easy is it to test?



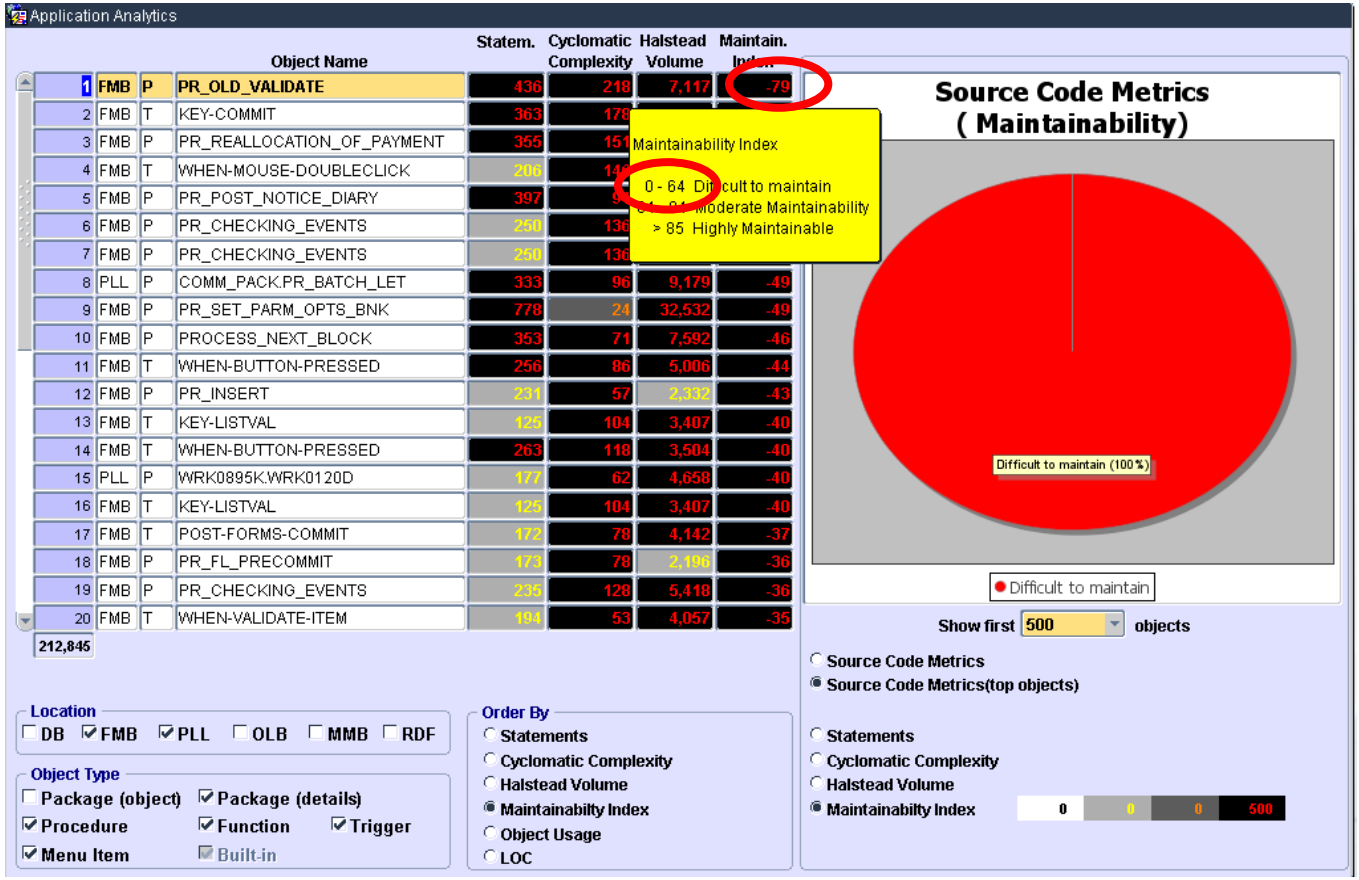
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- Halstead Volumetrics – How easy is the code to comprehend and maintain for developer



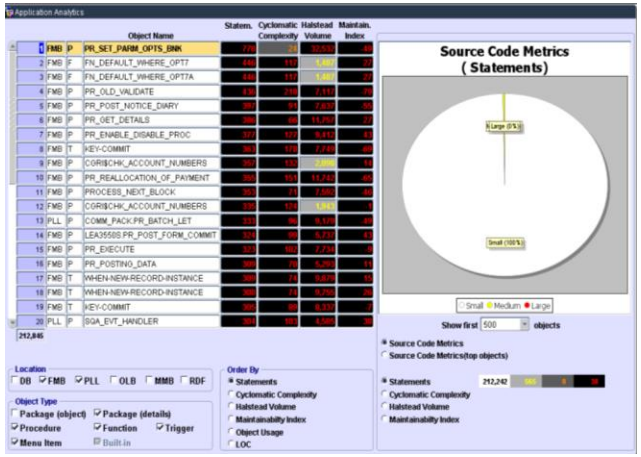
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

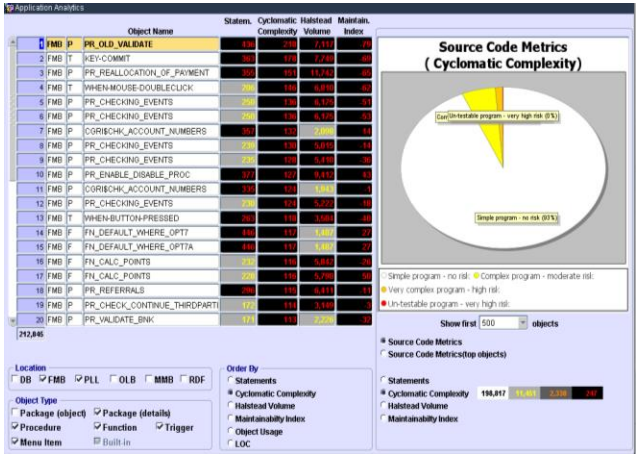
- Maintainability – How easy is the code to maintain?

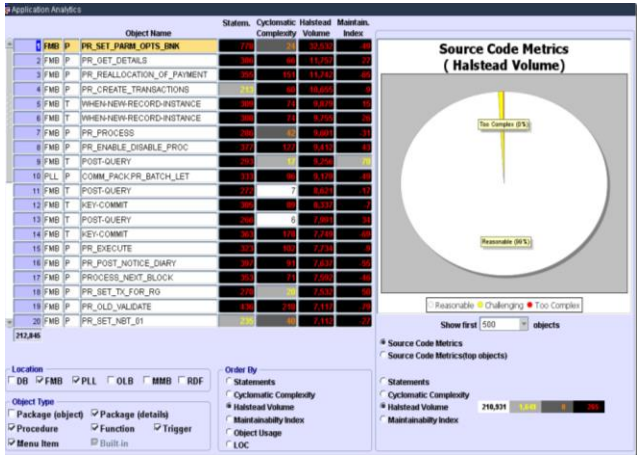


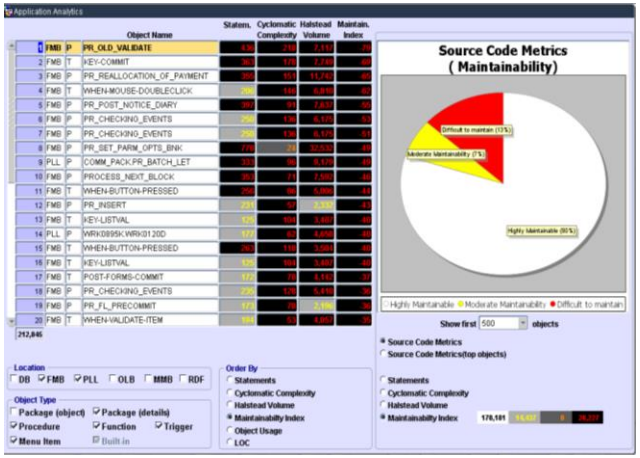
WHAT HAVE WE LEARNED ABOUT OUR APPLICATION?

- So the application really is complex. Right?







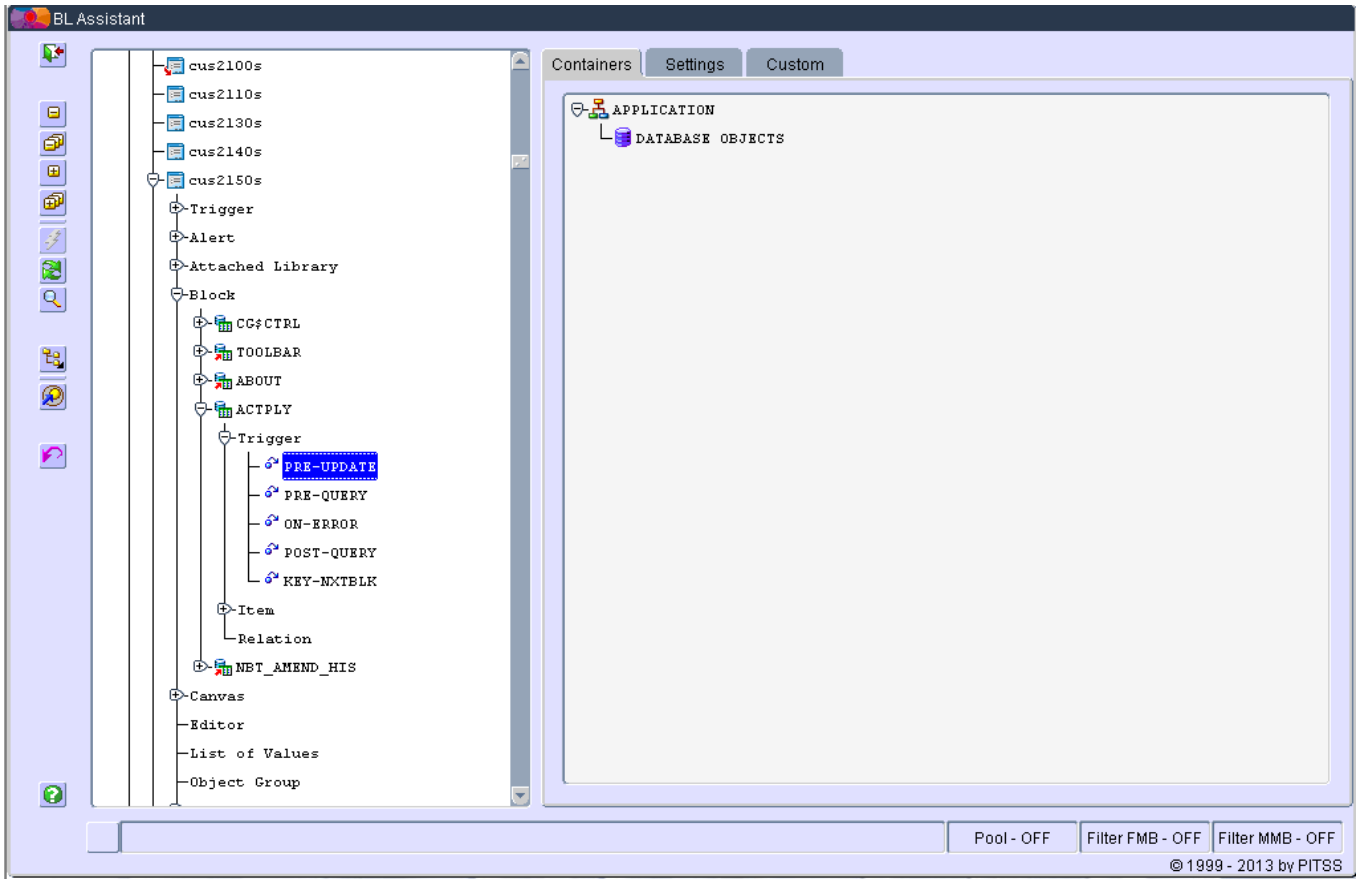


THE JOURNEY TO SOA

PITSS APPLICATION ENGINEERING

WHAT CAN WE START TO REUSE IN A SOA WORLD?

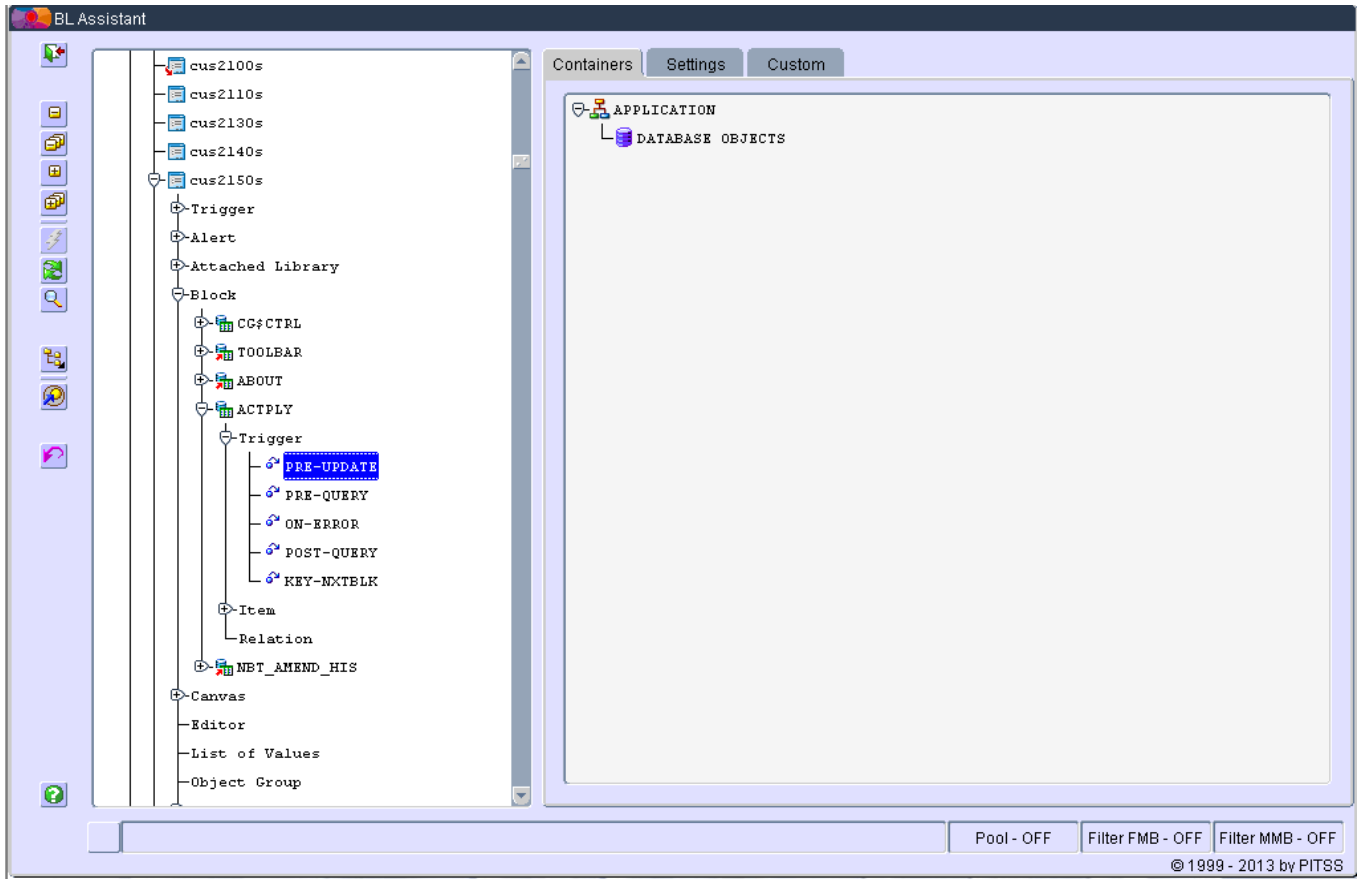
- We have logic within our existing Forms functionality which we may want to reuse rather than having to re-write within a new technology



- Using the Application Engineering functionality within PITSS, where it is appropriate we can look to push this logic from Oracle Forms into the database

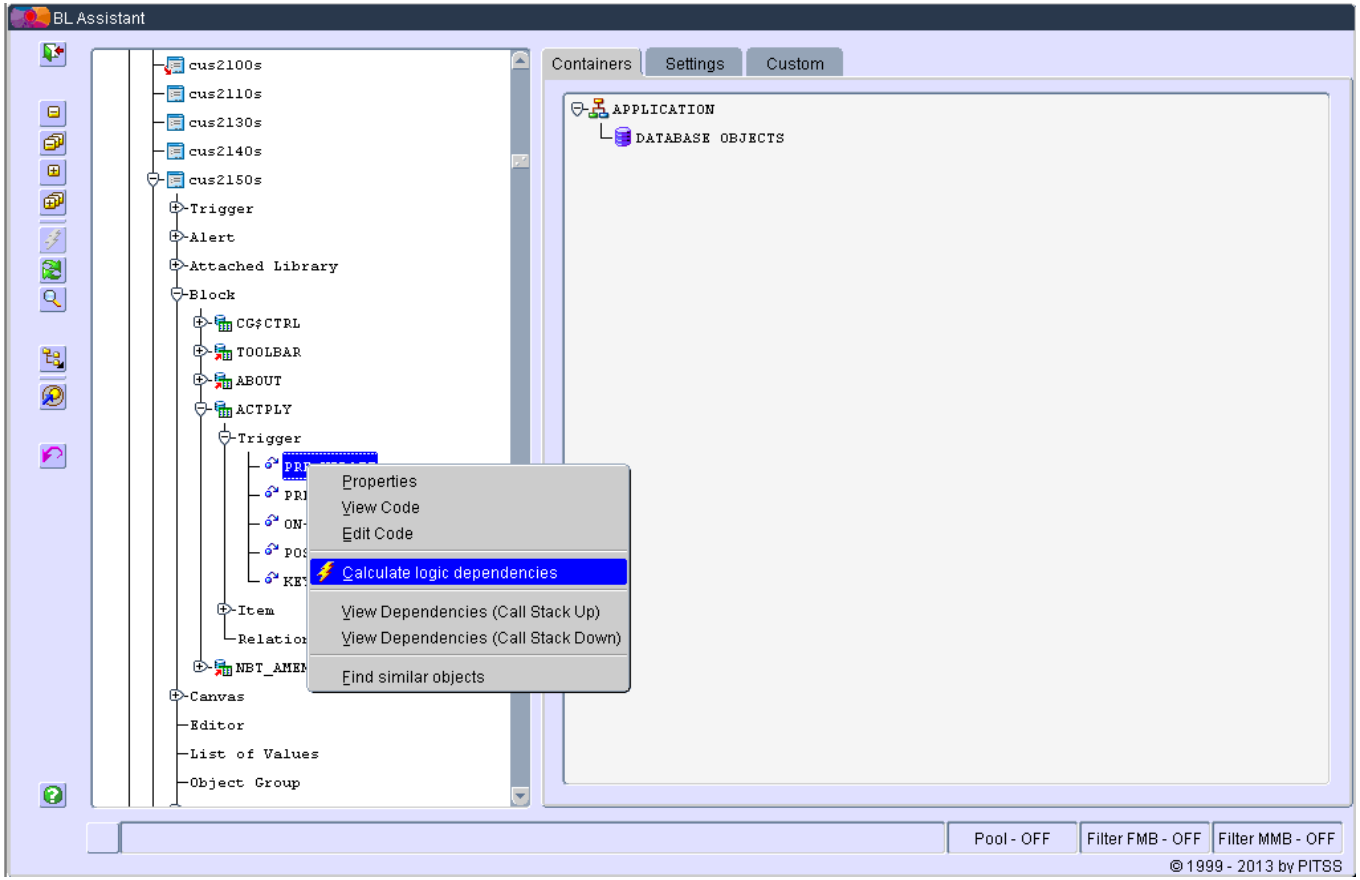
WHAT CAN WE START TO REUSE IN A SOA WORLD?

- The PRE-UPDATE trigger contains logic which allows the business user to link customer records and associated customer related data together



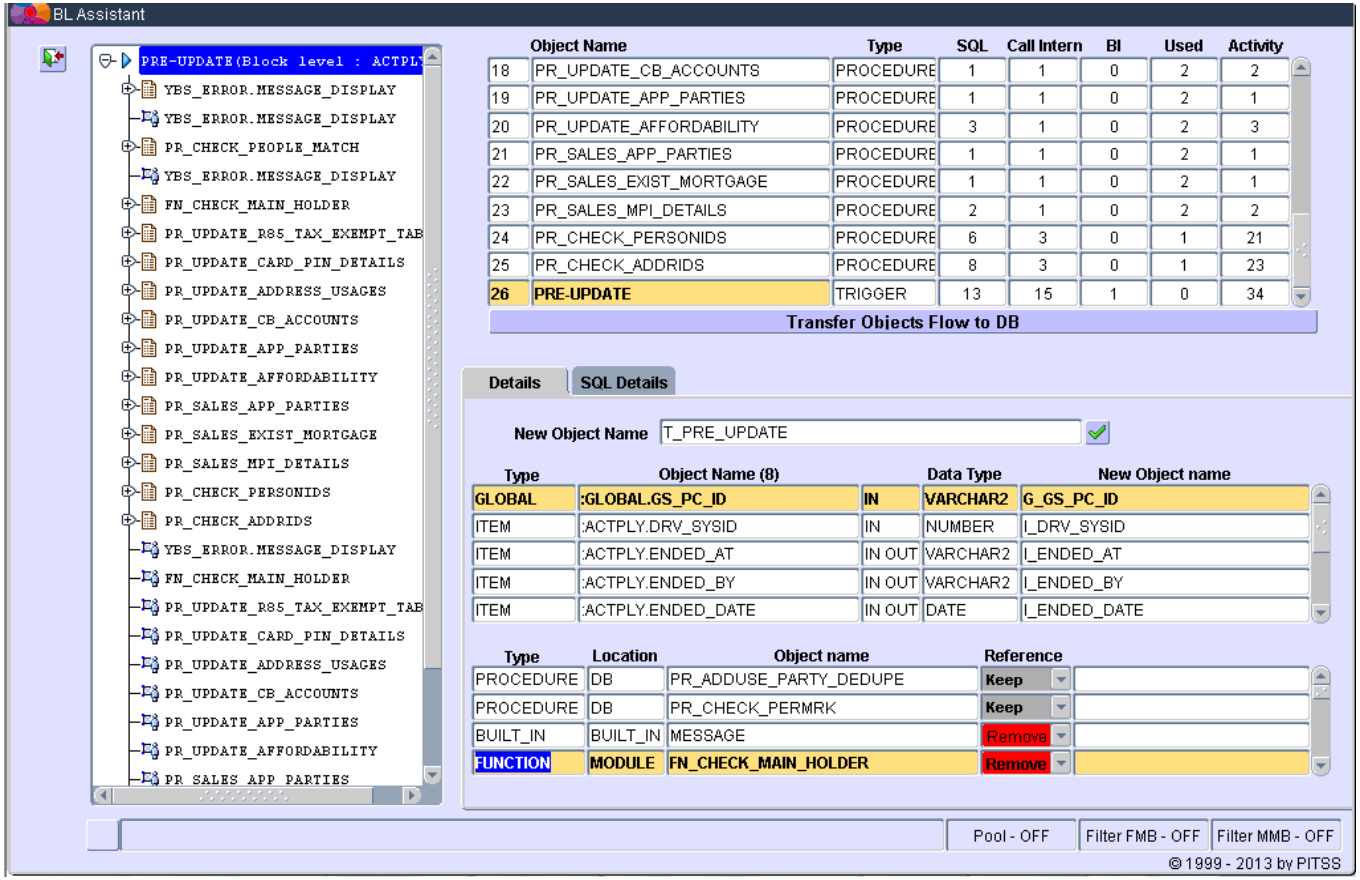
WHAT CAN WE START TO REUSE IN A SOA WORLD?

- PITSS calculates all the dependencies within the trigger



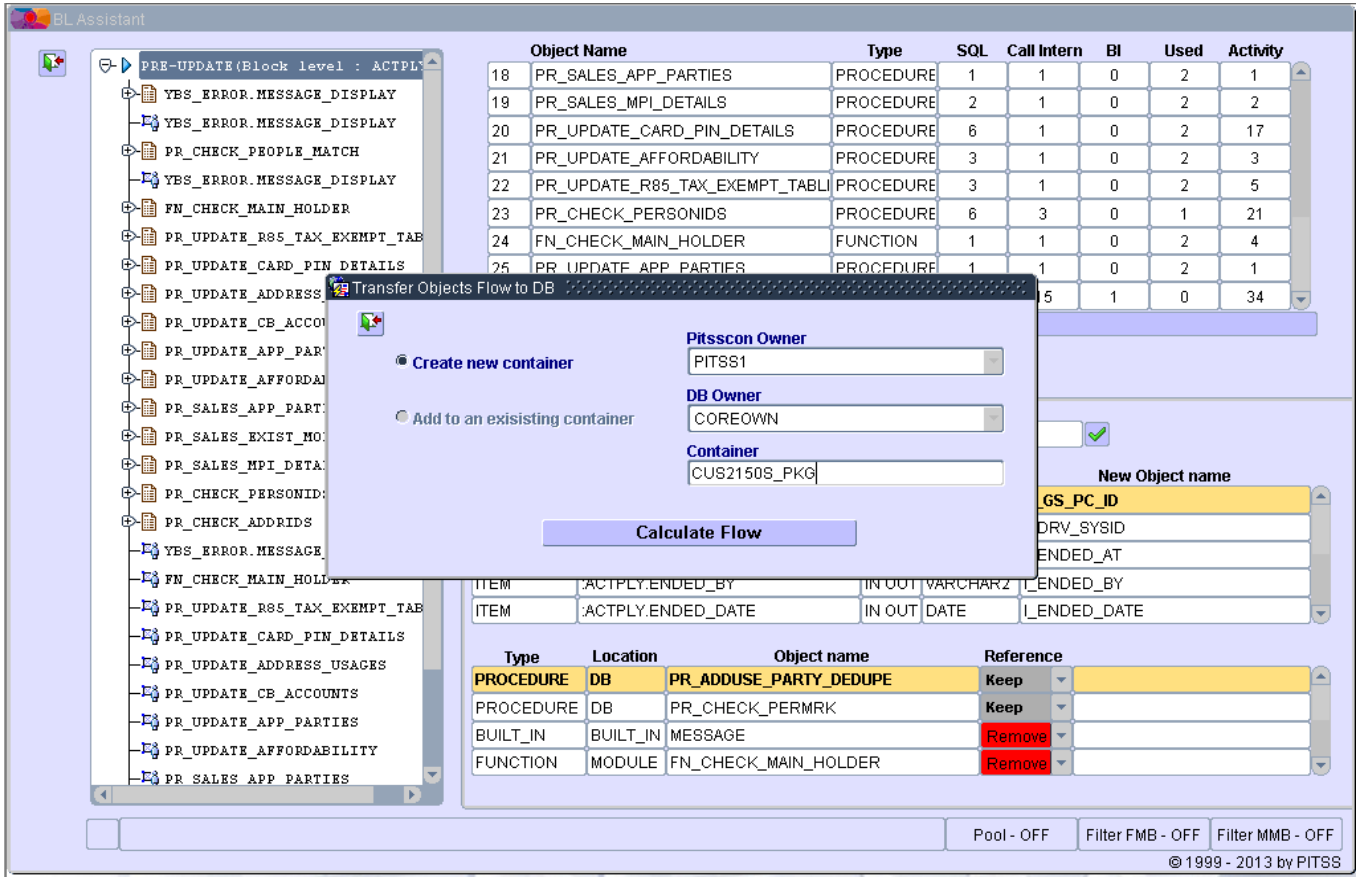
WHAT CAN WE START TO REUSE IN A SOA WORLD?

- And displays them for review…



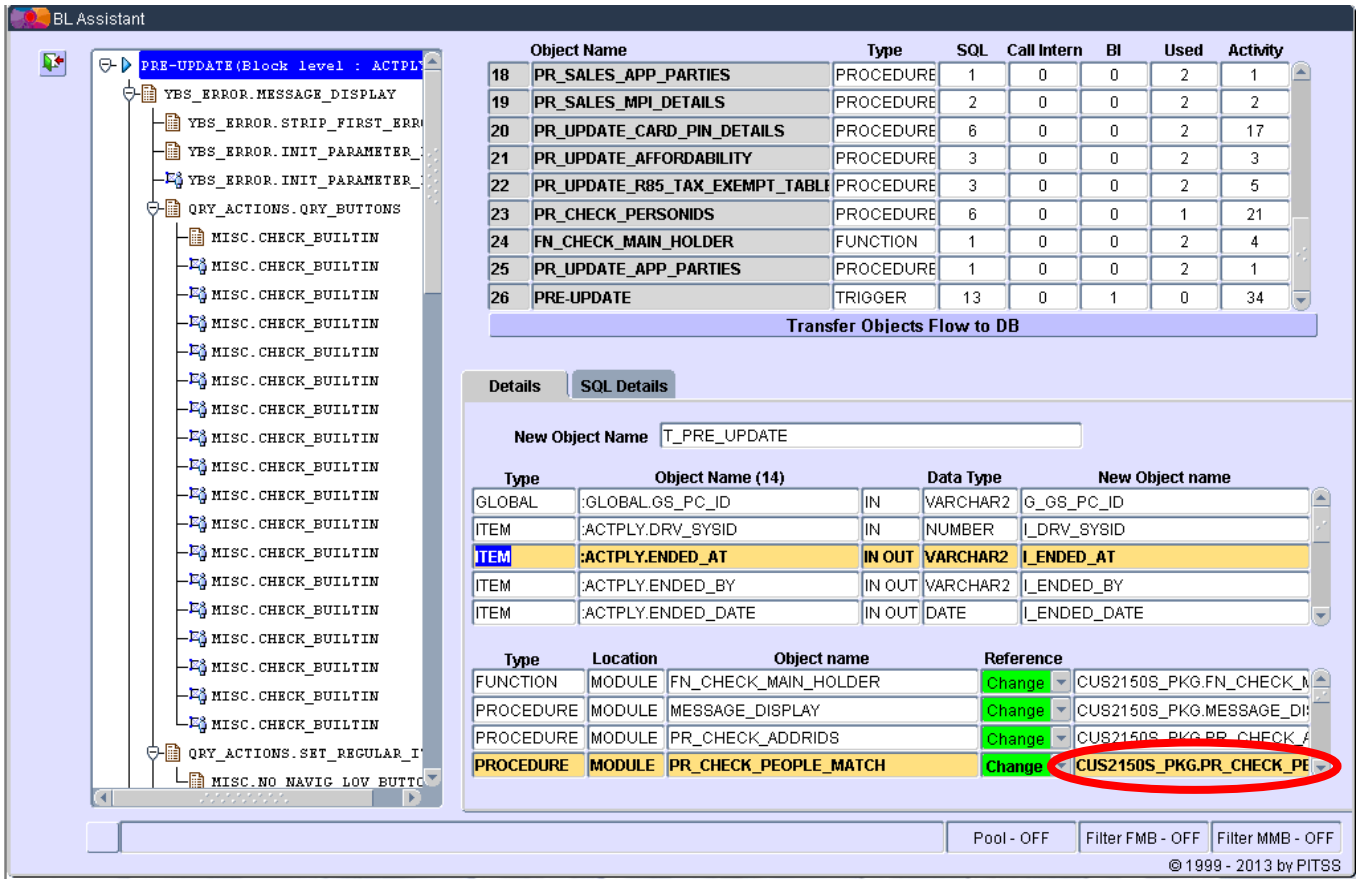
WHAT CAN WE START TO REUSE IN A SOA WORLD?

- The objects can then be transferred to a database package



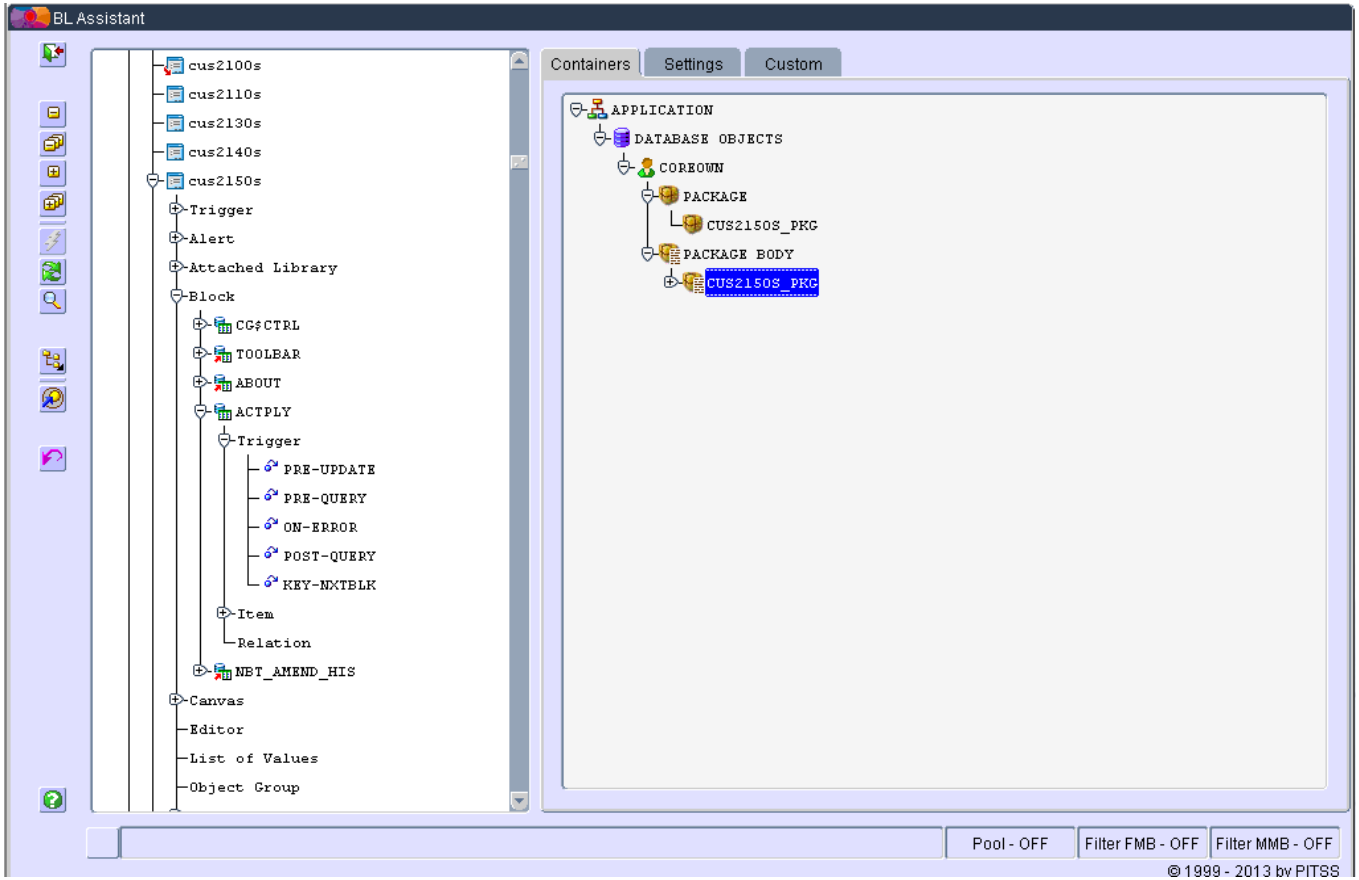
WHAT CAN WE START TO REUSE IN A SOA WORLD?

- Internal logic calls now reference other objects transferred to the package



WHAT CAN WE START TO REUSE IN A SOA WORLD?

- The newly created package can be inspected and edited within PITSS before being applied to the database



- This example was estimated by development to have reduced the amount of developer effort required by between 60-75% over doing it manually

NEXT STEPS FOR YBS

NEXT STEPS

- Strategic Journey continues – currently mapped out to 2019
- SOA & BPM has initially proved to be perhaps more difficult than we originally envisaged
- Importance of our investment in Oracle Forms is back on the agenda (It is running our Business today!). We have an upgrade project in progress - although in reality, it was the constraints of IE8 which has pushed the upgrade to the fore
- Identify further opportunities to exploit using PITSS. We would like to take the opportunity as part of the Forms upgrade to remove some of the considerable quantities of 'dead code' which exists within our application today
  - Would demonstrate we are shrinking our 'legacy' code base as we increase deployment of newer technologies and code, but also
  - Make this legacy code 'easier and simple' to maintain in future – but….
  - Requires a change of mind-set, both Management and Developer - currently no time is allocated to improving or refactoring our code when checked out by projects
- Continue to push the benefits of PITSS to our development community and harden code refresh processes
- Look to exploit new features of PITSS – specifically:
  - Development effort in Source Code Analytics – more accurate estimating?
  - User Journey's – ability to track execution activities within the application which are not directly visible to our end-user

Any regrets?

- That we didn't have the tool 10 or more years ago!

QUESTIONS?

Thank you for listening