ODTUG Kscope12
SAN ANTONIO, TEXAS JUNE 24-28

Taming of the Shrew – Documenting an APEX Application

Dietmar Aust
Opal-Consulting, Köln
www.opal-consulting.de

OPAL Consulting

Agenda

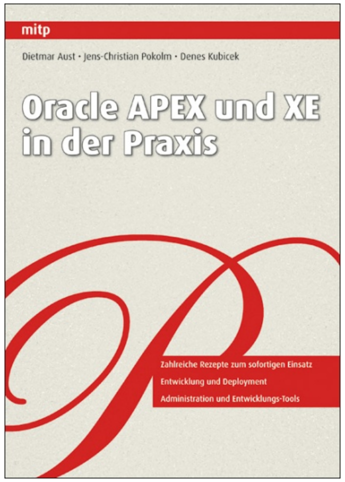
- The Background
- Templates and Checklists – there is a place for everything
- How to Manage a Delta Release
- WordWiki

Introducing Opal Consulting

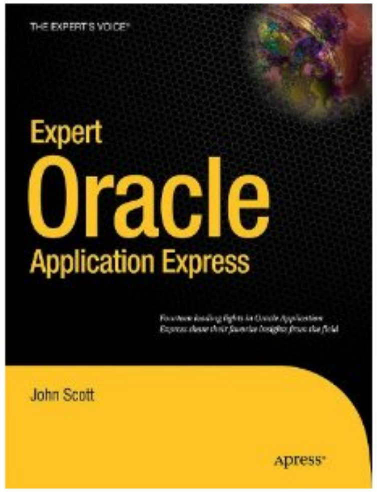
- Building Oracle based Web Applications since 1997
  - Portal, Forms, Reports, OWA Toolkit, now APEX!
- Dipl.-Inform. Dietmar Aust, Freelance Consultant
  - Master's Degree in Computer Science (MSCS)
- 1997-2000: Consultant at Oracle Germany
- Since 09/2000: Freelance Consultant, Since 2006 – APEX only!
- Blog: http://daust.blogspot.com/
- Regular presenter at Oracle conferences (ODTUG, DOAG, OOW)
- Author of the JasperReportsIntegration toolkit
  - Cost free alternative for generating print ready reports in APEX.
  - http://www.opal-consulting.de/tools

Introducing Opal Consulting

- Giving APEX trainings regularly in Germany together with Denes Kubicek


mitp
Dietmar Aust · Jens-Christian Pokolm · Denes Kubicek
Oracle APEX und XE in der Praxis
Zahlreiche Rezepte zum sofortigen Einsatz
Entwicklung und Deployment
Administration und Entwicklungs-Tools

- Co-author of „Oracle APEX und XE in der Praxis“
  - Published 21.12.2009 in German


THE EXPERT'S VOICE
Expert Oracle Application Express
John Scott
apress

- Co-author of „Expert Oracle Application Express“
  - Published 25.05.2011
  - Charity project in memory of Carl Backstrom and Scott Spadafore (previous members of the APEX Team)

The Background ::Challenges

- Too little documentation or … too much. Finding the relevant granularity is difficult
  - Automatic documentation / generation tools will give you too much information
- Documentation is not maintained in a timely manner
  - The existing documentation is outdated most of the time
  - Documentation is not oriented at the source code
- Delta vs. full documentation
  - During the lifetime of an application only the changes are documented (for each release). There is no comprehensive documentation of the current state of the application.
- Documentation Deliverables must be in MS Word format
  - Corporate standards, static text, data model diagrams, embedded MS Office documents

The Background ::Challenges

- Redundancy
  - We must avoid redundancy
    - Extremely hard to maintain
    - Quickly confusing and the complete documentation will be perceived as unreliable

The Background
::Goals

- Writing the **smallest amount** of documentation which is **still meaningful**. Therefore we have to identify **the relevant parts** of an application to document. Write and maintain the documentation with **the least amount of effort.**

- **Don‘t make me think!**
  - Project pressure builds up => People will stop documenting!
  - Implement a process for documenting an application / project which I will only have to follow, based on templates and checklists using a few simple and clear rules to be followed, almost automatically

- **Delta vs. full documentation**
  - Implement a process to make sure the overall full documentation of the application is updated once the delta release is shipped

- **Deliverables in MS Word format**

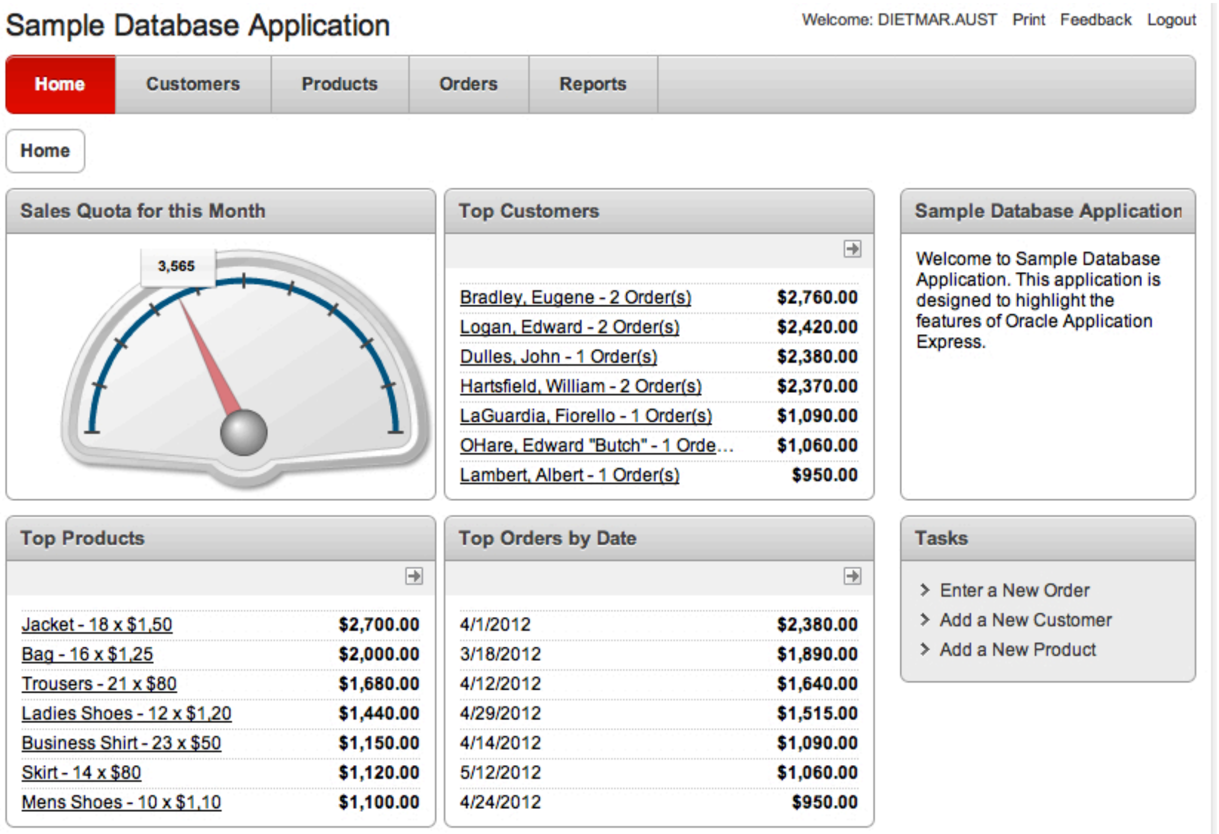
The Background ::What Should We Document?

- Target audiences?
  - Customer / Business User
  - Developer
  - Test-Team
  - IT-Operations
  - 3rd Level Support
  - End-User of the application

- For all of them the most important information is the **behaviour** and the **business rules** of the application
  - **The why and what is more important than the how**
  - As a developer you can always figure out the technical details when looking at the code. But we have to know what is „happening“ on a screen and what the expected result of a user activity is.

Templates and Checklists
::Demo using the Sample Application

- Demonstrating the concept with an example using the „Sample Database Application“ (version 4.x) which is available in every APEX workspace.



Welcome to Sample Database Application. This application is designed to highlight the features of Oracle Application Express.

Top Customers

Bradley, Eugene - 2 Order(s)	$2,760.00
Logan, Edward - 2 Order(s)	$2,420.00
Dulles, John - 1 Order(s)	$2,380.00
Hartsfield, William - 2 Order(s)	$2,370.00
LaGuardia, Fiorello - 1 Order(s)	$1,090.00
OHare, Edward "Butch" - 1 Orde...	$1,060.00
Lambert, Albert - 1 Order(s)	$950.00

Top Products

Jacket - 18 x $1,50	$2,700.00
Bag - 16 x $1,25	$2,000.00
Trousers - 21 x $80	$1,680.00
Ladies Shoes - 12 x $1,20	$1,440.00
Business Shirt - 23 x $50	$1,150.00
Skirt - 14 x $80	$1,120.00
Mens Shoes - 10 x $1,10	$1,100.00

Top Orders by Date

4/1/2012	$2,380.00
3/18/2012	$1,890.00
4/12/2012	$1,640.00
4/29/2012	$1,515.00
4/14/2012	$1,090.00
5/12/2012	$1,060.00
4/24/2012	$950.00

Templates and Checklists ::Documents and Structure

Scope Document
- **Overview for Management and Customer**
- Problems, Goals, main User Groups
- Availability and SLAs, Overview of Interfaces and the data flow
- Overview of Requirements

Requirements-Detail Document
- Explain the Requirements in detail
- Different approaches possible, e.g. Use Cases or functional Requirements
- Just assign a number and describe it in the most appropriate way

System Design Document
- GUI: APEX application(s)
- Modules (business rules / logic)
- Physical data model
- Jobs and Interfaces

Physical Implementation
- GUI: APEX application(s)
- Modules (business rules / logic)
- Physical data model
- Jobs and Interfaces

Closely linked

Source code, APEX Applications

Templates and Checklists ::Documents and Structure

- The system design and physical implementation should be as similar as possible
  - So that we can reverse engineer the design document from the existing source code
  - We can start with an E/R diagram. Once implemented we should only document the physical data model.
  - Remember the goal: "Write and maintain the documentation with **the least amount of effort."**
  - Automated extraction from the source code is the key!

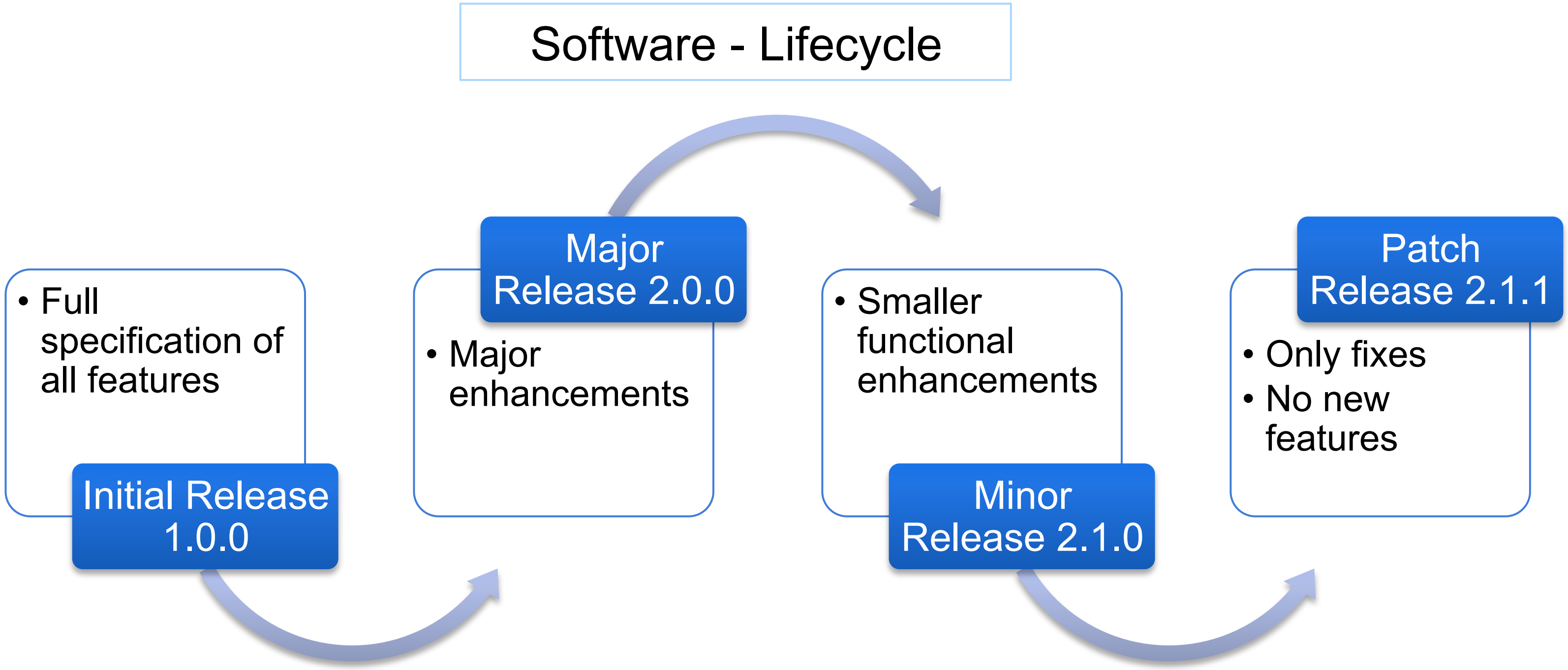
- Define a fixed document structure for all document types
  - **Using a template like a checklist**, use the sections that you need in your project, skip the others

Templates and Checklists
::Demo using the Sample Application

- Review the existing documentation for release 1.0
    - Scope document
    - Requirements-Detail document
    - System Design document

How to Manage a Delta Release ::Initial Release vs. Delta Release

- **Initial Release** := Full specification of all implemented features
- **Delta Release** := Specification of all **modifications** to an **existing** software (major, minor or patch)



How to Manage a Delta Release ::Different Types of Requirements

- Requirement Hierarchy: all requirements can have children to further refine the requirement

- System Requirement
  - Maps directly to a functionality in the system like „Manage customer (create, update, delete)“, „Import CSV-customer list“ or „Enter new order“
  - Level of granularity: system requirements are **almost atomic**
  - The complete list of system requirements **describe each behaviour** of the application (either by a user or by the system)
  - The system requirements are often the basis for the test team (**testable**)

- Change Requirement
  - Describes the **change of a SINGLE existing system requirement**
  - E.g. the requirement „Add column SAP_no to customer“ will change the existing system requirements:
    - Manage customer (create, update, delete), import customer list, report x, report y
  - Impact on the application? Test team can retest the relevant parts

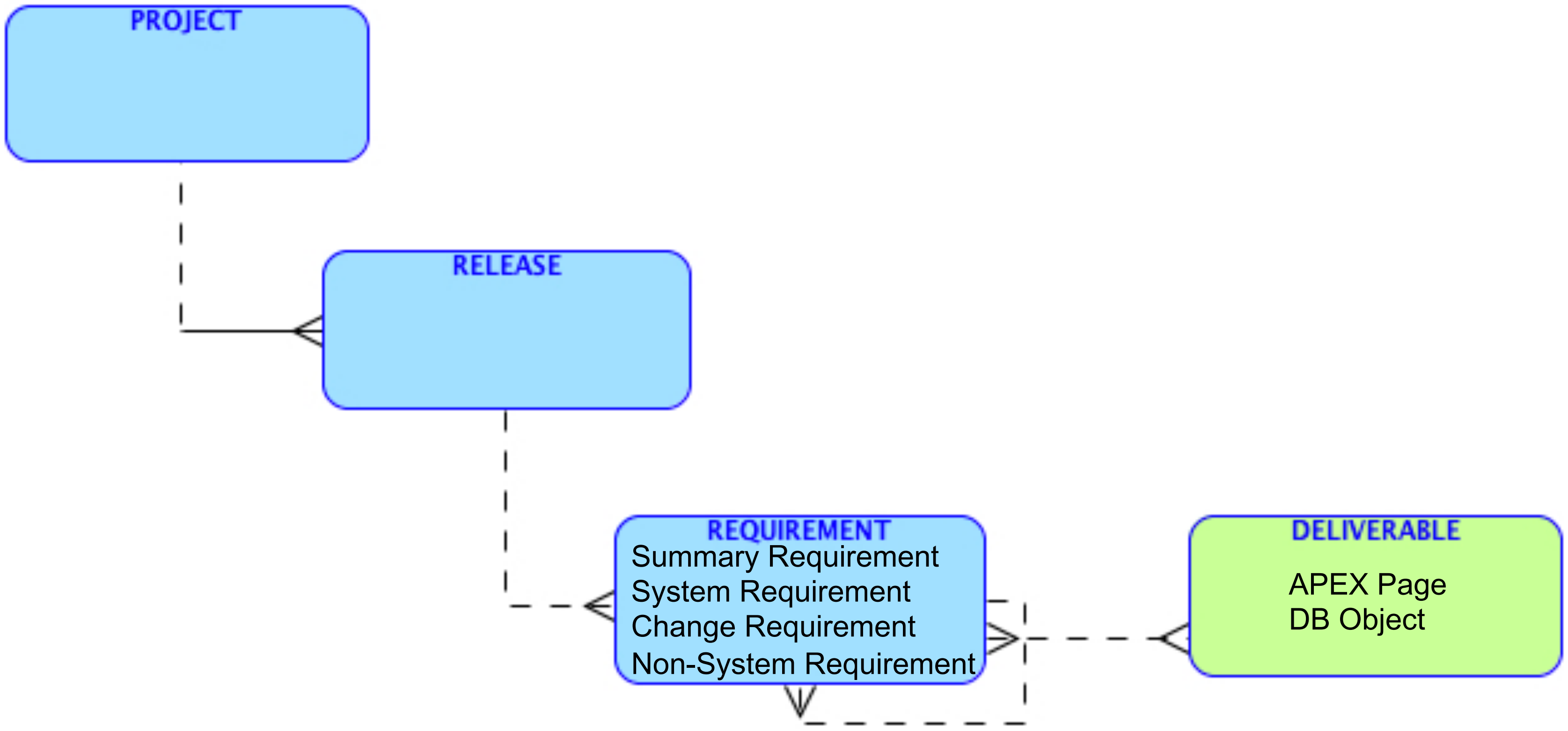
How to Manage a Delta Release ::Different Types of Requirements

- Non-System Requirement
  - Very similar to the system requirement, it just doesn‘t change the algorithms of the application, e.g. a one time import of seed data
  - Often used in delta releases
  - Needs to be tested
  - **Not required for a complete system description**

- Summary Requirement
  - **Higher level** requirement**, a grouping** of requirements, must be refined
  - We need to interpret the summary requirement and
    - Break it down and create change, system or non-system requirements as child requirements
    - Could even be converted into one of the other requirements after refinement, no childs needed then

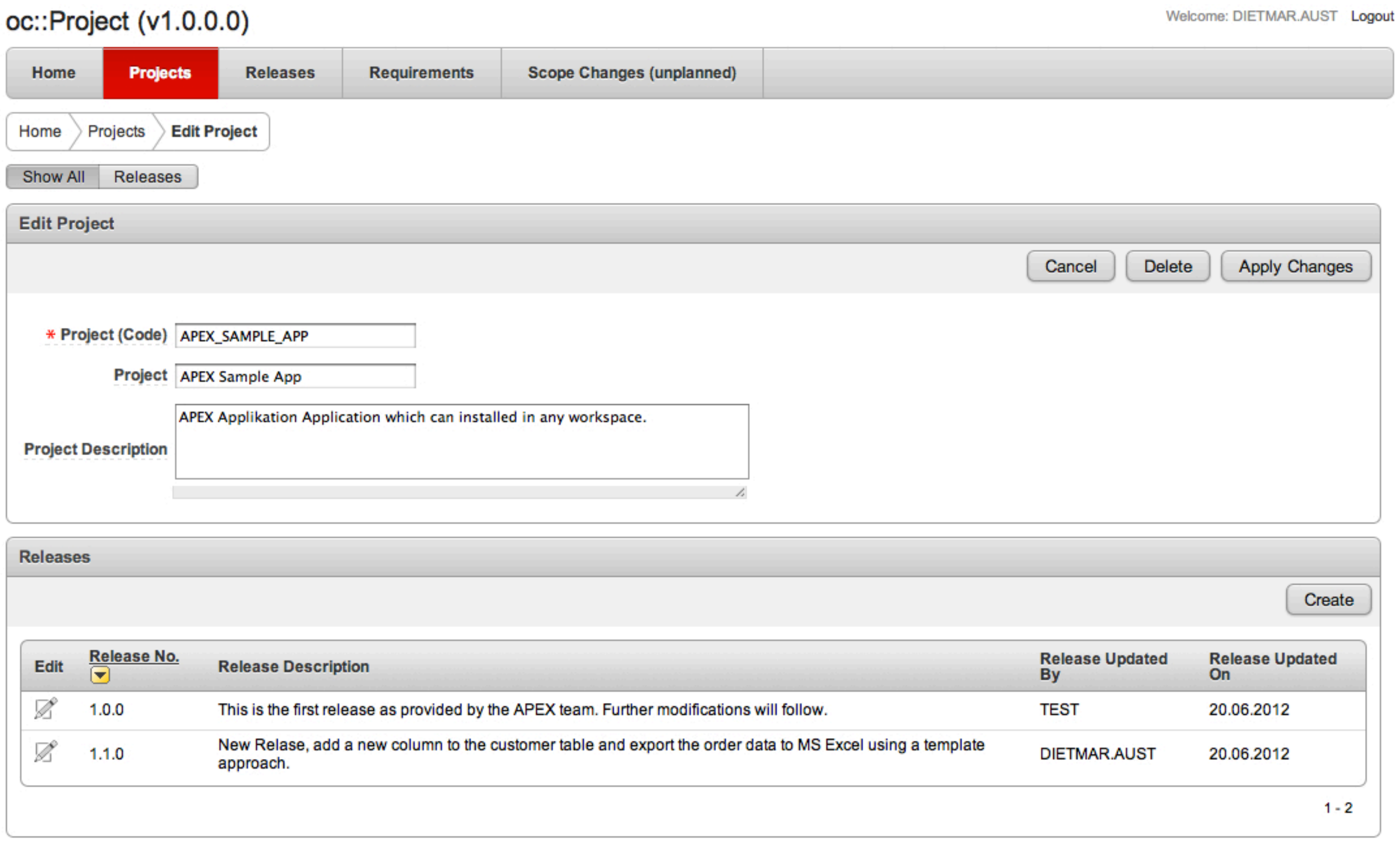
ODTUGKscope12

How to Manage a Delta Release ::Project, Release and Requirement



How to Manage a Delta Release ::Demonstration of a Delta Release

- A simple Project Management Tool



oc::Project (v1.0.0.0)

Welcome: DIETMAR.AUST Logout

Edit Project

* Project (Code)	APEX_SAMPLE_APP
Project	APEX Sample App
Project Description	APEX Applikation Application which can installed in any workspace.

Releases

Edit	Release No.	Release Description	Release Updated By	Release Updated On
✎	1.0.0	This is the first release as provided by the APEX team. Further modifications will follow.	TEST	20.06.2012
✎	1.1.0	New Relase, add a new column to the customer table and export the order data to MS Excel using a template approach.	DIETMAR.AUST	20.06.2012

1 - 2

How to Manage a Delta Release ::Demonstration of a Delta Release

- Release 1.1.0 of the APEX sample application

- Features:
  - **SUMMARY** Requirement: „Add column SAP_NO to customer“
    - **CHANGE** Requirement: Add column SAP_NO to report customer overview
    - **CHANGE** Requirement: Add column SAP_NO to customer form
    - **NON-SYSTEM** Requirement: Import current SAP numbers once into customer table
  - **SUMMARY** Requirement: Orders > Export Orders to Excel
    - Converted into a **SYSTEM** Requirement

How to Manage a Delta Release ::Demonstration of a Delta Release

- Demo:
  - Show Release Definition
  - Show all System Requirements
  - Export Requirements into Delta Release Specification (as HTML)
  - Show MS Word differencing between iterations to communicate to the test team

- At the end of the release we copy/paste all changed system requirements from the delta release specification into the system requirements document => Easy!

- How can we maintain the system design document?

WordWiki
::Our Approach

- Maintaining the system design document?
- Smooth transition
  - Start with a design specification in MS Word
  - Create an application based on the design
  - Replace the static design with placeholders `(:apex.page pageid=1:)`, then reverse engineer the existing application and extract the structure and the comments
- Benefits
  - We can mix static content / diagrams / embedded MS Office documents with extracted source code and object metadata (pages, object definitions, etc.)
  - We decide the point in time when to make the switch

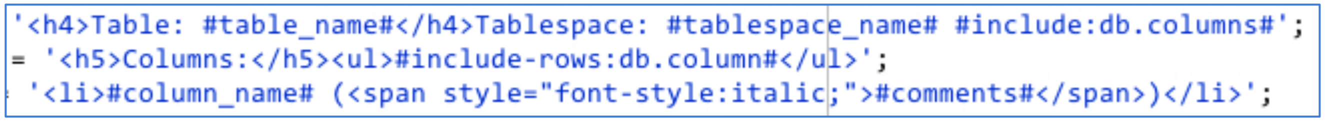
WordWiki ::Architecture

1. The MS Word document contains markup
   - `(:apex.page pageid=1:)`
   - `(:table-view name=DEMO_CUSTOMERS:)`

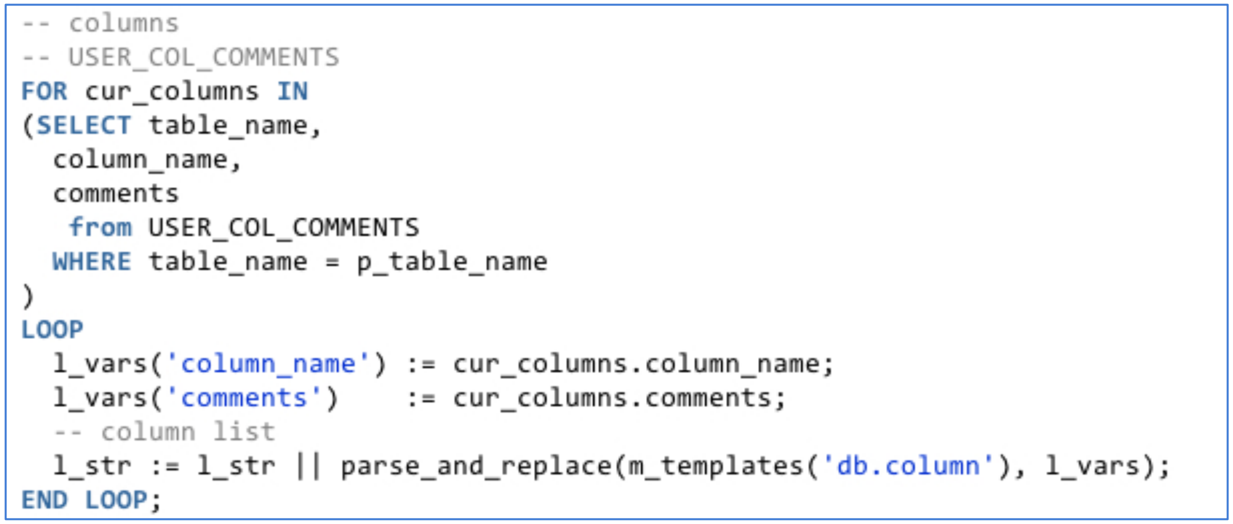
2. A macro in MS Word is executed
   - Copy the document
   - Find markup
     - construct url to call an APEX application
     - http://dev-min.opal-consulting.de:8080/apex/f?p=20120618:2:0::::P2_CALL_INTERFACE,P2_TYPE,P2_APP_ID,P2_PAGE_ID,P2_NAME,P2_THEME,P2_THEME_DETAIL_LEVEL:SHOW_DB_OBJECT,DB.OBJECT,100,,demo_orders,demo_order_items,demo_product_info,,
     - HTML is generated by the APEX application
     - The HTML document is copied into the current document and replaces the markup
     - In MS Word: `<H1>` will be converted into `Heading1` **using the current format template**!

WordWiki ::Architecture

- Characteristics
  - The generated HTML is fully customizable by implementing a template approach:



```
'<h4>Table: #table_name#</h4>Tablespace: #tablespace_name# #include:db.columns#';
= '<h5>Columns:</h5><ul>#include-rows:db.column#</ul>';
 '<li>#column_name# (<span style="font-style:italic;">#comments#</span>)</li>';
```



```
-- columns
-- USER_COL_COMMENTS
FOR cur_columns IN
(SELECT table_name,
  column_name,
  comments
   from USER_COL_COMMENTS
  WHERE table_name = p_table_name
)
LOOP
  l_vars('column_name') := cur_columns.column_name;
  l_vars('comments')    := cur_columns.comments;
  -- column list
  l_str := l_str || parse_and_replace(m_templates('db.column'), l_vars);
END LOOP;
```

WordWiki ::Demo

- Demo:

Summary

- Templates and Checklists – there is a place for everything
  - Making the monkey happy
- How to Manage a Delta Release
  - Describe the modifications in the current release
  - Copy the current state of the system requirements into the overall documentation
  - Show differences between iterations using MS Word „compare documents“ feature
- WordWiki
  - Still working on that
  - Good solution to mix static text and documentation in the source code

Q&A

Contributions to the approach and implementation:
Wolfram Ditzer
Tom Fuhr

Q&A

Dietmar Aust
Opal-Consulting, Köln

www.opal-consulting.de
daust.blogspot.com
dietmar.aust@opal-consulting.de

JDD Spreadsheet Suite

http://jdd-software.com