



# *Standards and Measures of Quality!*



Maggie Tompkins

Defense Finance and Accounting Service (DFAS)

MAOP - December 2005

# *Why Use Standards?*

- Consistency
- Eases learning curve - Staff turnover
- Maintainability - Impact analysis
- Reduces long term costs
- Promotes reusability
- **Absolutely required for rapid application development and/or iterative development**

# *Consistency*



- Eric Okin, Assistant Deputy Director, Data Architecture, for DFAS
- **When it comes to standards, it is far better to be consistently incorrect than it is to be inconsistently correct!**

# *Where Do Standards Come From?*

- Custom made - Very Expensive !!!
- Adopt industry accepted standards - best practices
- Buy them - Best practices - Modify as necessary

# *Standards Recommendation*



- **Oracle Designer, A Template For Developing An Enterprise Standards Document** by Mark Kramm and Kent Graziano, Prentice Hall
- Inexpensive
- Best practices
- Outstanding start for any organization

# *Customize It - Make It Your Own!*

- Instructions Include: “Copy the file from the CD-ROM to your hard disk, open MS Word, do a global replace on *Company-Name*, save the file, **and you are done!**”
- Actually you should consider each standard for your organization and modify as needed but **IT IS A GREAT START !!!**

# *It's All In The Name*

## *Class Words = Consistency*

- Code (CD) : A combination of one or more numbers, letters, or special characters substituted for a special meaning - category, status, type - Domain is often applied
- Quantity (QY): A non-monetary numeric value - average, balance, count, deviation, factor, index, mean, median, mode

# *More Class Words*

- Amount (AM): A monetary value - average, balance, cost, price
- Rate (RT): A quantitative expression that represents the numeric relationship between two measurable units - acceleration, density, force, frequency, humidity, intensity, magnitude, percent, pressure, resistance, scale, speed, tension, torque, velocity

# *And More Class Words*

- Identifier (ID): Sequence generated surrogate primary key for a table
- Name (NM): A designation of an object expressed in a word or phrase
- Text (TX): An unformatted character string of words used for longer expressions such as notes

# *How To Judge Standards Violations?*

- Standards violations are like crimes - They are not created equal
- Jaywalking is not as serious a crime as capital murder
- **In RAD projects, risk levels are accepted as a trade off to time - minor defects are accepted as a cost of doing business**
- Standards violations are relative - you judge!

# *Severity of Defects in Designer Applications - Meaningful Scale*

- Fatal Flaws
- Critical Concerns
- Preventable Problems
- Minor Defects
- Insignificant Defects

# *Severity of Defects in Designer Applications - Fatal Flaws*

- Fatal Flaws - The system will stop
- This could result in an abort, codeabend, or inconsistent database state
- Users cannot access the system or major portions of it
- Immediate intervention is required to restore the system to a usable state
- Ex: Tables in the wrong tablespace

# *Severity of Defects in Designer Applications - Critical Concerns*

- Critical Concerns - The system will function but not very well
- Poor performance
- Poor database design
- Missing Table API or index
- Inefficient PL/SQL code
- Ex: Unique key component not uppercase

# *Severity of Defects in Applications - Preventable Problems*

- Preventable Problems - Future analysis and development are hindered
- Inadequate or non-existent documentation
- Current production is not effected or only effected slightly
- Ex: Attribute with no length on datatype
- Ex: Column missing link to an attribute

# *Severity of Defects in Designer Applications - Minor Defects*

- Minor defects - When I get extra time - (WIDGET) - Not the same as Oracle Applications Widget Objects!
- Typos and inconsistent definitions
- Annoying discrepancies
- Everyone knows what the correction should be

# *Severity of Defects in Designer Applications - Insignificant Defects*

- Forget It, Never (FIN) - Never gets fixed
- Anything that management determines is not worth the resource investment to fix
- It should never be a fatal flaw or critical concern and seldom a preventable problem

# *Standards, Defects, and Scale*



- Use exponential scale - not all defects are created equal (linear)
- Fatal Flaw = 100 points
- Critical Concern = 10 points
- Preventable Problem = 2 points
- Any other defect = 1 point

# *Total Defect Points*




- Multiply each fatal flaw by 100
- Multiply each critical concern by 10
- Multiply each preventable problem by 2
- Multiply each minor defect by 1
- Total defect points is the sum of all of the above numbers

# *Function Points*



- Measure the size of an application
- International Function Point User Group (IFPUG) standards
- In Oracle Designer some automated function point reports (older versions only)
- Other automated methods - manual methods

# *Defect Density*


$$\text{Defect Density} = \frac{\text{Total Defect Points}}{100 \text{ Function Point Count}}$$

# *Defect Density*



- Used as a metric to compare the quality of applications regardless of their size
- Excellent metric to measure improvement (or lack of) with iterative releases within the same application

# *Measurement and Monitoring*



- Systematic method, generally automated, that allows for the defect density of an application to be determined quickly and often
- If periodic measurements can be taken quickly and easily, then flaws can be identified while there is still time to fix them before going to production

# *Measurement and Monitoring*



- **Continuous measurement is required for RAD projects**
- Continuous measurement is recommended for iterative release development where defects can be fixed in the next release
- Scripts using PL/SQL or SQL
- Accesses metadata in the Oracle Repository

# *Measurement and Monitoring*



- Since SQL scripts are fairly easy to construct against the Designer Repository, a full set of scripts can be developed quickly for analysis and design standards
- **The real value is in the identification of defects quickly so they can be fixed!**

# *Don't Fight Mother Nature*

## *Oracle Applications*

- **Standards must be flexible** when integrating custom code with Financials & other apps
- Change the standard to meet the commercial off-the-shelf (COTS) software if consistency can be achieved
- “bending is better than breaking”, “go with the flow”, “consistency above standard”

# *Oracle Applications Are Special*

- Because Financials uses many Oracle developed forms, packages, and procedures, more keywords need to be excluded from casual use by developers. For example, folder, calendar, and appcore would not be allowed as variable names
- Oracle Apps often make use of special journal tables, snapshots, and replication tables that tend to restrict the length of names

# *Integrate Applications With Custom Software Development*

- Besides length requirements that are more restrictive, naming conventions in Applications are important due to the large number of files and database objects
- Standardizing these naming conventions allows developers to find and manipulate objects more quickly

# *Capability Maturity Model*

## *(CMM-CMMI)*

- Developed by the Software Engineering Institute (SEI) at Carnegie Mellon University
- SEI is a Department of Defense (DOD) funded think-tank formed to insure quality in the development of software by DOD contractors
- 5 levels of maturity for a software development organization

# *CMM Level 1*

- Initial level
- Little documentation - ad hoc - chaotic
- Rely heavily on a few heroes to do work
- More inherent risk in completing a project on time at an acceptable level of quality
- Risk in loosing the hero!

# *CMM Level 2*



- Repeatable level
- Basic project management in place
- Organization can plan and track work while expecting a reasonable degree of quality
- Processes are in place that can be repeated to achieve similar results

# *CMM Level 3*



- Defined level
- Organization looks at the best practices defined by each project and outlines a standard software process for the organization to follow
- Very compatible with function points and defect density

# *CMM Level 4*



- Managed level
- Takes advantage of the measurements that have been put in place at levels 2 and 3
- These measures are used and tracked to make management decisions
- The processes and products are understood so that the affects of any change can be reviewed quantitatively

# *CMM Level 5*



- Optimizing level
- Process and metrics work together to identify the best solutions for organizations
- The insertion of new methods or new technologies can be evaluated quantitatively
- A process drives decisions about making changes to the processes themselves

# *Function Points in Designer*

- Usages, associations, and implementations define the function points in the Oracle Repository and are the essence of any Oracle application
- Function-Entity usages
- Table-Module usages
- Function-Module implementations
- Entity-Table implementations

# *Usages in Oracle Designer*

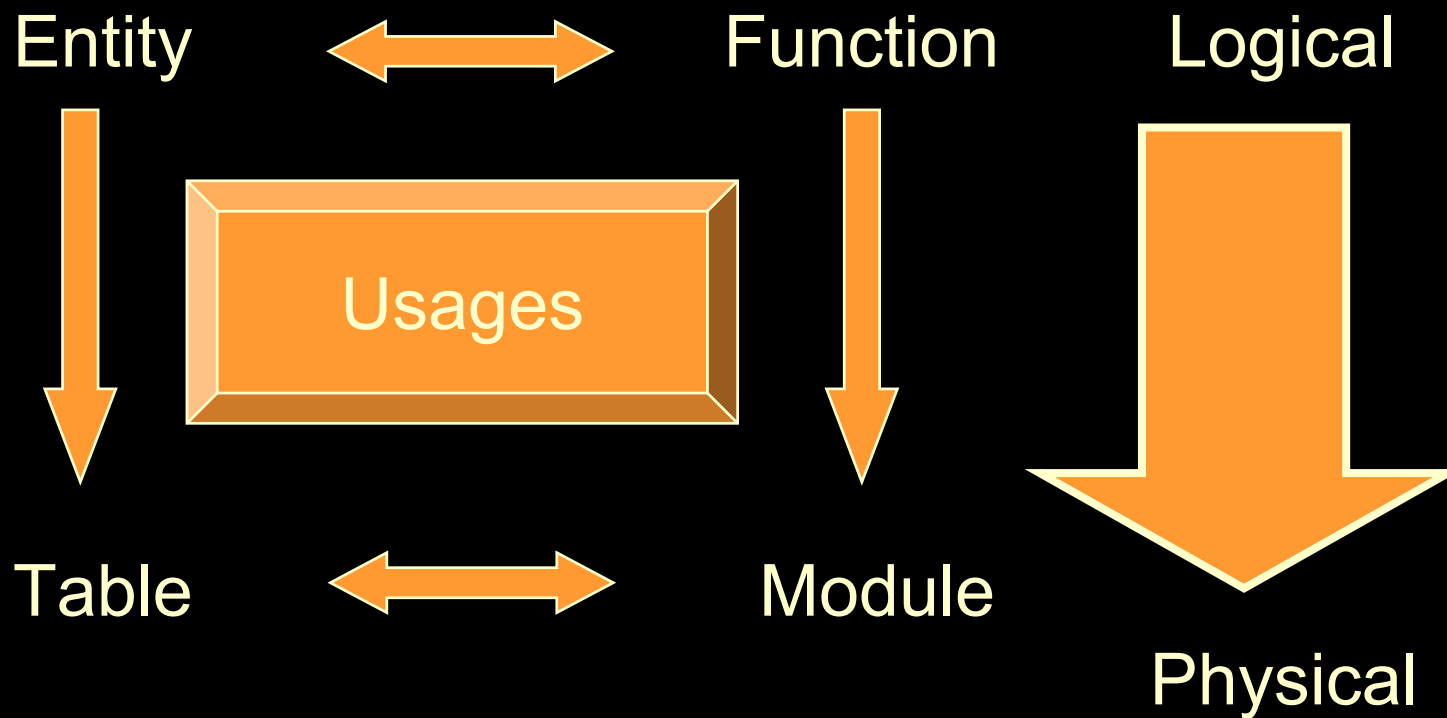
- Each entity is generally used by many functions
- Each function generally uses many entities
- Each entity generally becomes one table
- Each function generally becomes one module
- Each table is generally used by many modules
- Each module generally uses many tables

# *Usages in Designer*



- For each entity usage, there generally are many attribute usages
- For each table usage, there generally are many column usages
- The many types of usages in Designer document the application and provide tracability between requirements and implementation

# *Usages in Designer* *Logical to Physical*



# *Payback Time - Impact Analysis*



- Impact analysis reports in Oracle Designer are very good
- The Dependency Analyzer impact analysis tool in Designer 6i is even more powerful
- If the usages are complete and accurate, then the impact analysis will be complete and accurate, just like the function point counts

# *RAD and Iterative Releases*

- **The ability to do iterative releases rapidly is totally dependent on accurate and complete impact analysis**
- **The ability to do accurate impact analysis is totally dependent on the usages being accurate and complete in the Designer Repository**

# *Cost Effective Development*



- RAD and iterative release development require quality impact analysis quickly
- Define usages accurately and completely and impact analysis becomes easy
- Set standards for analysis and design
- Develop scripts to measure and monitor
- Count function points and defect density



# *Standards and Measures of Quality!*



Maggie Tompkins

Defense Finance and Accounting Service (DFAS)

Margaret.Tompkins@dfas.mil

Thank You MAOP 2005 !