



2009 Annual Conference  
New Orleans, LA

Programmatic Load of the  
OC Global Library

Steven Rifkin  
Sr. Director of Consulting  
BioPharm Systems



# Topics

- Glib Builder API Goals
- Functions implemented
- Ensuring Consistency between Forms and APIs
- API Structure and Error Handling
- Uses for the APIs
- Caveats

# Glib Builder APIs: Goals

- Build Global Library objects through APIs rather than through Oracle Clinical Forms
- APIs and Oracle Clinical Forms exhibit identical functionality
  - Records in database the same whether object created from the form or from the API
  - Same field and inter-field checks
- Single function handles creation and update of a Glib object
- APIs Designed as Packaged Procedures
  - Error handled via PL/SQL exceptions
- Able to be validated

# Glib Builder APIs: Functions Implemented

- DVGs
  - Create and update
  - Set to Active
- Questions
  - Create and update
  - Set to Active
- Question Groups
  - Create and update
  - Set to Active

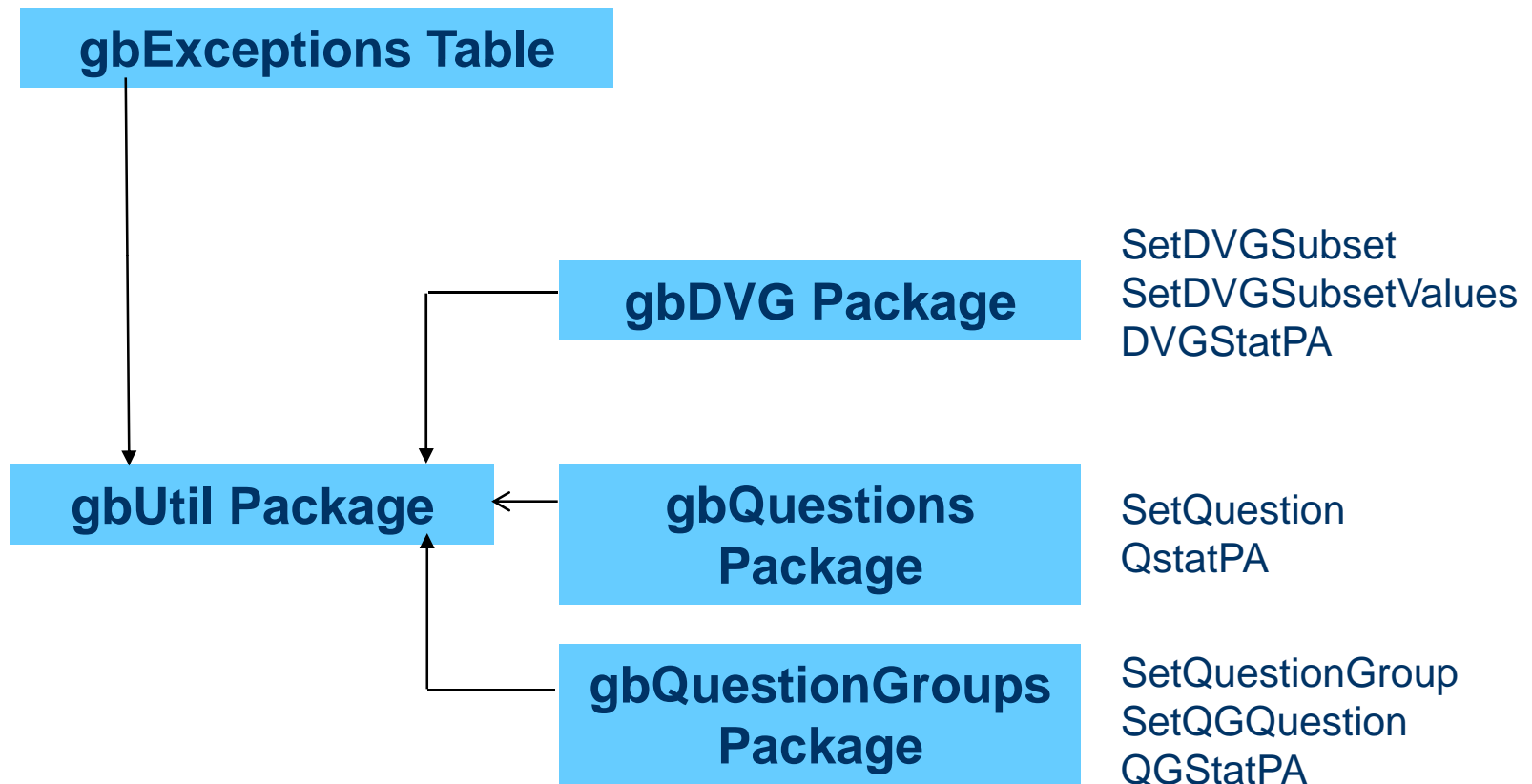
# Glib Builder APIs: Consistency

- Mimic of OC functions with PL/SQL APIs required knowledge of all field and inter-field SQL\*Form validations
  - Mostly experience, trial and error
  - Several tables populated by database triggers as inserts made to underlying tables
    - E.g., extended attributes for a new Question

# Glib Builder APIs: Consistency

- Ensure all mandatory fields are required in the API code
  - May not be a not null column in the base table
- Check for disallowed duplicate records enforced in form
- Test each field in function for allowable characters, field lengths, ranges and case sensitivity for API
- Determine code list used if field has an LOV attached
- Determine which fields/records can be updated or deleted on form
- Allow mixed case entry but force to upper case where required to mimic form behavior

# API Structure and Error Handling



# Function Operation: DVGs

- SetDVGSubset
  - Creates or updates “master” section of the DVG
  - No support for Thesaurus DVGs
- SetDVGSubsetValues
  - Inserts or updates “detail” values of the DVG Subset
- DVGStatPA
  - Activates the specified DVG Subset
  - Activating base subset (subset 0) creates and activates the first subset (subset 1)

# Function Operation: Questions

- SetQuestion
  - Creates or updates a Question
  - No support for Questions Types of UNIT, COMPLEX, QUESTION SET or THES VALIDATED
- QStatPA
  - Activates Question

# Function Operation: Question Groups

- SetQuestionGroup
  - Creates or updates the “master” Question Group
- SetQGQuestion
  - Adds or updates (an existing) “detail” Question in the Question Group
  - Full call or short call for adding a question
    - Short call uses the default attributes from the Glib Question
- QStatPA
  - Activates Question Group

# Error Handling

- If “expected” error is encountered, a common utility routine is called with an error number
- Utility routine will fetch the error message from the gbExceptions Table and RAISE\_APPLICATION\_ERROR with the message and the number
- WHEN OTHERS captures “unexpected” problems if encountered
- APIs designed so calling program should perform commit if no errors, or rollback when any exception is raised
  - Some procedures may have changed records prior to raising the exception

# Example Errors

- 20001 Cannot create a new DVG before Subset 0 has been created
- 20002 Missing value for DVG Name
- 20003 Missing value for DVG Domain
- 20004 Missing Value for DVG Subset number
- 20005 DVG must be specified as INTERNAL or ALPHA
  
- 20050 Question name contains invalid characters
- 20051 Question name too long
- 20052 Domain for Question is unknown
- 20053 Invalid Question Type or is null
- 20054 Invalid Question Data Type or is null
  
- 20125 Question Group name is missing
- 20126 Question Group domain is missing
  
- 20200 Question Type CHAR must have Data Type CHAR
- 20201 Question Type CHAR must have null Date Time Type Code
- 20202 Question Type CHAR must have null Date Time Format Code

# Validation

- APIs were thoroughly tested but no formal system validation has been performed
- APIs were prepared with full User Requirements which defined all cross field checks
- Full Validation Package with Traceability Matrix and Test Cases can be relatively easily developed

# Uses

- Move all (or parts of) Glib from one instance to another if replication not possible
  - New objects will not be created if object already exists in target
  - Could load all objects into a newly defined domain
- Use to rapidly install new standards (e.g. SDTM) into a new Glib Domain (over multiple instances)

# Caveats

- Code not supported or sponsored by Oracle
- As Glib changes, these routines must change
  - Tested against V453

Must be used with caution!

# Summary

- APIs developed to mimic 3 Oracle Clinical Functions
- When functions execute, can create a Glib object from a calling program
- There appears to be no difference between API created objects and those created through the OC form

# Contact Information

Steve Rifkin  
BioPharm Systems  
908 822 0553  
[srifkin@biopharm.com](mailto:srifkin@biopharm.com)



?

Steve Rifkin has almost 14 years of experience working with the OLS Product Suite. As an Functional Consultant, and then as a Practice Manager with Oracle Consulting in the Oracle Clinical Group, Steve has assisted over 60 companies with implementation of Oracle Clinical, RDC and TMS. He has led the implementation process and has provided training and performed custom coding. Since joining Biopharm Systems in 2000, Steve has been responsible for developing Oracle Clinical training courses, and providing training and implementation services for Biopharm clients.