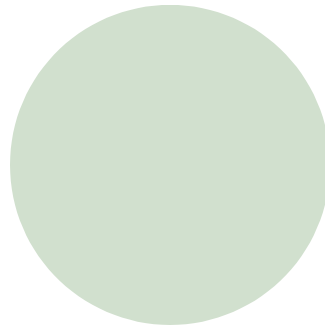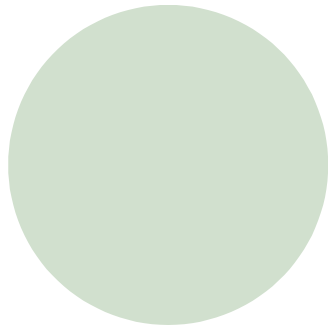# Genetic Resources Information Management System

Genetic Resources Center, IRRI

RPrantilla , MHabito, and
TMetz

ICIS Workshop 2006

Outline

- Part 1: GRIMS Overview
- Part 2: IRGCIS Integration with IRIS
- Part 3: GRIMS GUI

Part 1: GRIMS Overview

I. Introduction
   - a. Background
   - b. Objectives
   - c. Modules

II. Application Development
   - a. Tools
   - b. Testing
   - c. Documentation
   - d. Source Code Management
   - e. Graphical User Interface (GUI) Design

III. Database Management
   - a. Database Functions and Procedures
   - b. Database Triggers
   - c. Aides to Parallel Implementation
   - d. Schema
   - e. Migration to PostgreSQL

Terminology

- Incoming sample – a germplasm sample that is received by the genebank through a donation, a collecting activity, or some other activity and that is worthwhile preserving

http://cropwiki.irri.org/icis/index.php/GRIMS

Background

- There is an existing system called the International Rice Genebank Information System (IRGCIS)
  - Developed using Oracle Developer 2000
  - Oracle Database Server 8.0.5
- GRIMS is a re-implementation of IRGCIS that uses ICIS Schema (i.e. GMS, IMS, and part of DMS) and some sub-systems.

Objectives

1. To upgrade the existing information system by providing efficient quality control over all genebank operations, including the acquisition of germplasm, its safe storage and maintenance, its characterization, and its distribution.

3. To integrate IRGC data into International Rice Information System (IRIS)

5. To improve efficiencies and cost-effectiveness in conserving and making available genetic resources as Global Public Goods

Main Modules

- Acquisition
  - Donor information registration and passport registration
  - Incoming sample initial seed increase, viability monitoring, and inclusion into the IRGC collection
- Multiplication
  - Replanting of IRGC accession to replenish stock, and for special request (i.e. Genebank restoration)
  - Dependent on seed stock and seed viability

Cont…
Main Modules

- Seed Management
  - Inventory monitoring
  - Viability monitoring
  - Seed distribution
  - Frequently requested accessions monitoring
  - NCGRP backup
  - Seed monitoring in the drying room
- Characterization
  - Morphological and agronomic traits
  - Reactions to biotic and abiotic stresses

Application Development : Tools

- **Borland Delphi Professional Version 6.0 / 2006**
  - **recently acquired Borland Developer Studio which includes Delphi 2006**
- **Fast Reports Reporting Tool**
- **Barcode Font**

Application Development :

Programming Methods

- Re-using of ICIS Delphi Libraries that interfaces to the ICIS DLL Function
- Common library used by GMSSearch, SetGen, and InTrack)

  - GMS, GMSLib, DMS, DMSLib
  - i.e. creating seedlist, retrieving valid values for the passport descriptor, etc.
  - Example of calling ICIS DLL inside delphi code:

    ```
    iFind:= DMS_getScaleDis(rSDis,FIND_FIRST);
    while iFind = DMS_SUCCESS do begin
            …..
            iFind:= DMS_getScaleDis(rSDis,FIND_NEXT);
    end;
    ```

Application Development :
Programming Methods

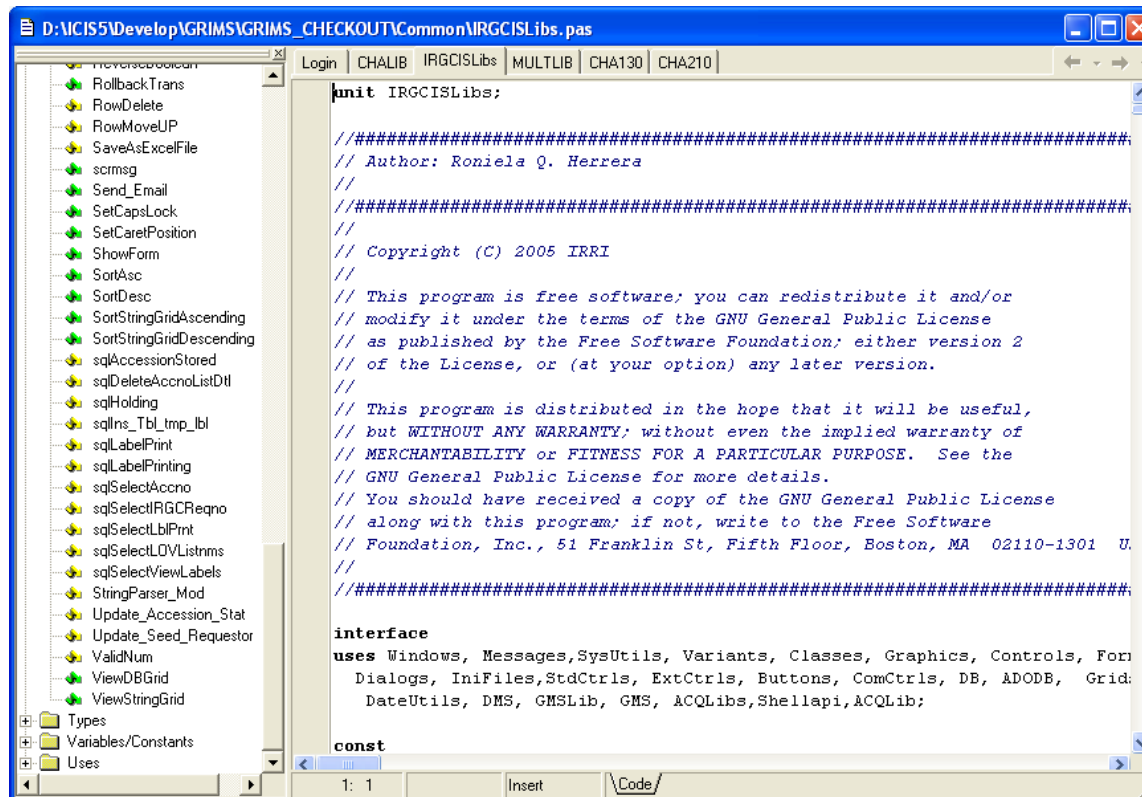- **Module-specific library: e.g. acquisition -> ACQLIB.pas, characterization-> CHALIB.pas**

Application Development :
Programming Methods

- **Common library for all modules: IRGCISLibs.pas**

Application Development :
Programming Methods

- **Naming convention of forms**

  ○ **All** .dfm **files are named using the use-case code**

  ○ **The** Name **property of a form is a string with prefix "frm" followed by the use-case code.**

  e.g. For CHA110.dfm: **'frmCHA110'.**

Application Development :
Programming Methods

- **Naming convention of form objects**

  - **Prefixed with a short string containing a few letters from the name of the component type**

  - **Followed by a word that describes the information/event related to the component**

  e.g.   btnSubmit is of type TButton
  → "Submit" button

Application Development :

Programming Methods

- **Naming convention of variables**

  ○ Prefixed with first letter of variable's data type followed by a word that describes the value being stored by the variable.

  e.g.   sSpecies is of type String

  stores the species code

Application Development :
Programming Methods

1. **Naming of functions and procedures**

   ○ **Name is descriptive of what the function/procedure does.**

e.g.

   ○ procedure Create_Prooflist;
   ○ function Insert_Generation(iGid,iAccno:longint; sCropyear, sSourceCropyear:string):boolean;

Application Development :
Programming Methods

8. **Re-writing of IRGCIS source code**

- Algorithms in the GRIMS modules are based on existing algorithms in the IRGCIS.

- GRIMS mirror the fundamental functionalities of IRGCIS.
  - However, a number of algorithms from IRGCIS were revisited and modified to meet the changing requirements of system users.

- Introduced new technologies (i.e. ICIS and bar-coding)

Application Development :
Documentation

- **GRIMS Development is documented in the form of Wiki articles on the Internet.**
- **ICISWiki: http://cropwiki.irri.org/icis/index.**
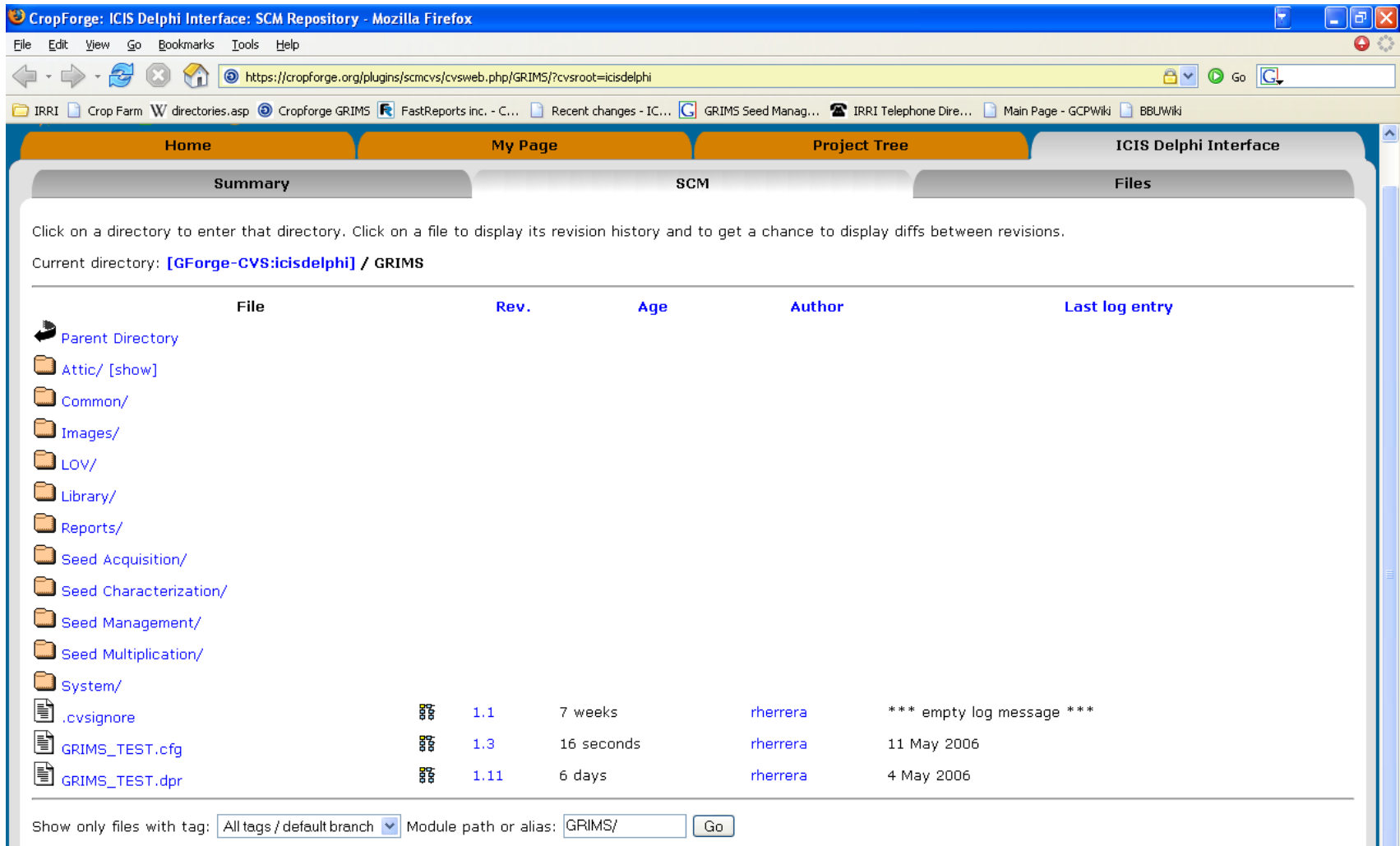
Application Development :
Source Code Management

CropForge: **https://cropforge.org/**

Application Development :
Graphical User Interface (GUI) Design

- **Most GRIMS UIs are patterned after existing IRGCIS UIs.**

- **Usability Rule-Of-Thumb: Minimize typing/keyboard input.**

IRGCIS

GRIMS

Application Development :
Graphical User Interface (GUI) Design

New UIs developed for GRIMS to address areas of improvement in IRGCIS
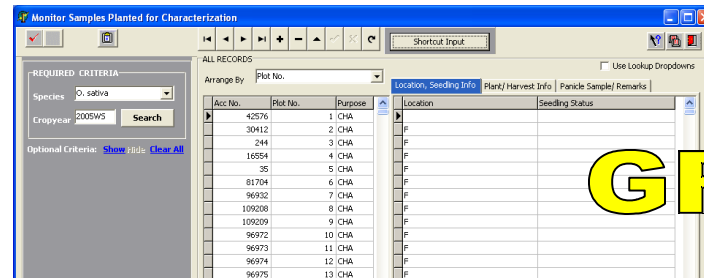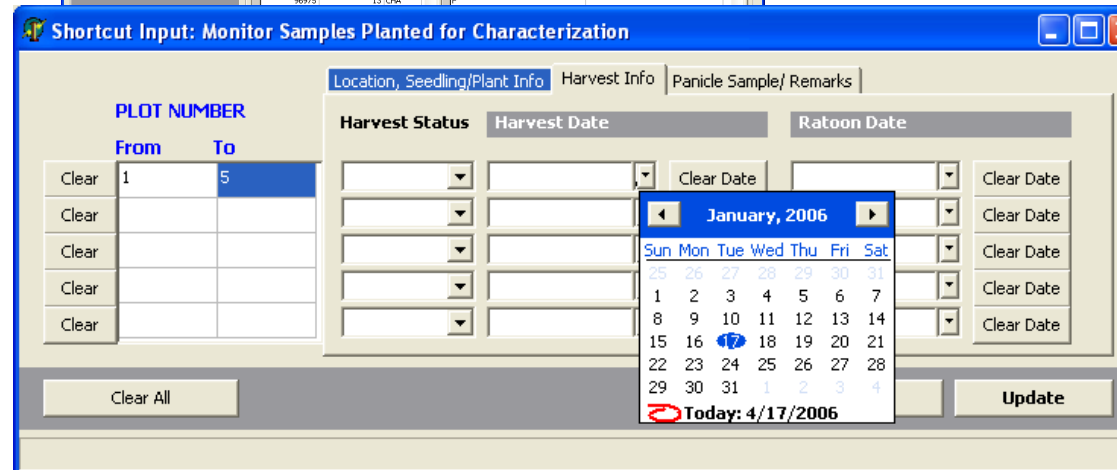
Application Development :
Graphical User Interface (GUI) Design

- **Printing of reports/ proof list in more re-usable format**
  - **MS Excel**
  - **Notepad**

- **A number of IRGCIS reports/ prooflist are created in Oracle Developer Reports builder which doesn't allow copying and pasting of the result to a new file format**

Part 1: GRIMS Overview

I. Introduction
   a. Background
   b. Objectives
   c. Modules

II. Application Development
   a. Tools
   b. Testing
   c. Documentation
   d. Source Code Management
   e. Graphical User Interface (GUI) Design

III. Database Management
   a. Database Functions and Procedures
   b. Database Triggers
   c. Aides to Parallel Implementation
   d. Schema
   e. Migration to PostgreSQL

Database Management:
Stored Procedures and Functions

- ## Database Server Specific
- ## Stored Procedure
  - Executes on the database with possible input & output parameters
  - Does data manipulation operation (i.e. insert, delete, update)

- ## Function
  - Does data manipulation operation as well, but returns a value
  - Can be called inside a SQL query
    - Example:
      ```
      SQL>  SELECT Get_Accno(gid) accno, gid, nval FROM NAMES
        2   WHERE gid =1 and ntype=6;

      ACCNO          GID        NVAL
      ----------     --------- ----------------------------------------
            747         1      T 12-42
      ```

Database Management:
Stored Procedures and Functions

```sql
CREATE OR REPLACE FUNCTION Get_Imsapack(xAccno NUMBER, xCropyear
VARCHAR2:='') RETURN NUMBER IS

xVal                    NUMBER;
CURSOR c1 IS
        SELECT NVL(SUM(trnqty),0)
        FROM IMS_LOT a, IMS_TRANSACTION b
        WHERE a.lotid=b.lotid AND trnstat=1
                AND scaleid=-1000
                AND a.eid= Get_Gidcropyear(xAccno,xCropyear);
BEGIN
        OPEN c1;
        FETCH c1 INTO xVal;
        IF c1%NOTFOUND THEN
                xVal := 0;
        END IF;
        RETURN xVal;
        CLOSE c1;

END;
```

> Cursors are pointers used to fetch rows from a result set.
> One can think of a cursor as a data structure that describes
> the results returned from a SQL SELECT statement.
> One of the variables in this structure is a pointer to the
> next record to be fetched from the query results.
>
> www.orafaq.com/glossary/faqglosc.htm

Database Management:
Calling a stored procedure inside Delphi code using ADO

```
function Mark_Insufficient_Distribution(strIrgcreqno:string):boolean;
//---------------------------------------
// author: rqh
// description : marks the seed as insufficient for distribution
//---------------------------------------
var storedProc:TAdoStoredProc;
begin
 try
   try
       Result:=True;
       if strIrgcreqno='' then exit;
       storedProc:=TAdoStoredProc.Create(nil);
       storedProc.Connection:=adoCnn;
       with storedProc do begin
             ProcedureName := 'Mark_Insufficient_Distribution';
             Parameters.Refresh;
             parameters.ParamByName(Parameters.Items[0].Name).Value := strIrgcreqno;
             ExecProc;
       end;
       GRC_Information('Done checking list for insufficient stock!');
   finally
       storedProc.Free;
   end;
 except on e:exception do
   begin
     Result:=false;
     messagedlg('Error encountered.',mtInformation,[mbOK],0);
   end;
 end;
end;
```

Database Management:

Aides to Parallel Implementation

- ## Use of Database Trigger
  - ○ A program in a database that gets called each time a row in a table is INSERTED, UPDATED, or DELETED.
  - ○ Allows checking that any changes are correct, or filling in missing information before the data is committed in the database

Database Management:

Aides to Parallel Implementation

```sql
CREATE OR REPLACE TRIGGER I_LOCAL.tbl_acs_tgr
            AFTER UPDATE OF TRAYNO, cropyear,amt_bulk, num_apack, amt_base ON accession_stored
FOR EACH ROW
DECLARE
            xScaleid INTEGER;

BEGIN
--Description: This trigger is fired whenever an update/ delete is made to ACCESSION_STORED table COLUMN
-----trayno, cropyear,amt_bulk, num_apack, amt_base.

    IF (:OLD.TRAYNO<>:NEW.TRAYNO)  THEN
        Update_Lot_Locid(:NEW.gid,:NEW.trayno,:NEW.stotype);
    END IF;

    IF (:OLD.amt_bulk<>:NEW.amt_bulk) AND (:OLD.amt_bulk=NULL AND :NEW.amt_bulk IS NOT NULL)
        AND (:NEW.STOTYPE='A') THEN
        Append_Trigger_Imsbulk(:NEW.gid,:NEW.accno,:NEW.cropyear,:NEW.amt_bulk);

    END IF;

    IF ((:OLD.amt_base<>:NEW.amt_base) OR(:OLD.amt_base IS NULL AND :NEW.amt_base IS NOT NULL) )
        AND (:NEW.STOTYPE='B')  THEN
        Append_Trigger_Imsbase(:NEW.gid,:NEW.accno,:NEW.cropyear,:NEW.amt_base);
    END IF;

    IF (:OLD.num_apack<>:NEW.num_apack)   AND (:NEW.STOTYPE='A') THEN
        Append_Trigger_Imsapack(:NEW.gid,:NEW.accno,:NEW.cropyear,:NEW.num_apack);
    END IF;
End;
/
```

Database Management:

Aides to Parallel Implementation

Append data to IMS tables

```
xDate:=TO_NUMBER(TO_CHAR(SYSDATE,'YYYYMMDD'));
xuserid:=Get_Icisuserid(USER);
xScaleid:=-1000;


xLotID:=Get_Lotid(xeid,xscaleid);
xCurrTrnqty:=Get_Imsapack(xAccno,xCropyear);
xTrnid:=Get_Nexttrnid(xeid,xscaleid);

  IF (xAmt>0) AND (xLotid=0) THEN

          SELECT LOT_SEQ.NEXTVAL
          INTO xLotid
          FROM dual;

          INSERT INTO IMS_LOT…
          INSERT INTO IMS_TRANSACTION …
          xTrnid:=xTrnid+1;

END IF;

          xTrnQty:=xAmt- xCurrTrnQty;

          IF xTrnqty<>0 THEN
                    INSERT INTO IMS_TRANSACTION …
          END IF;

END;
/
```

Database Management:
Oracle to PostgreSQL Database Migration

- Under way
- Slight differences on the following:
    1. Data type declaration
        - Varchar2
        - Varchar
    2. Function call
        - Select <function> from dual
        - Select <function?
    3. Creation of stored procedures and functions
        - Create or replace <function > Return varchar2 **IS**
        - Create or replace <function> Returns VARCHAR **AS** **$$**

            ...

            LANGUAGE 'plpgsql';

Part 2: Integration with IRIS

Part 2: Integration with IRIS

I. ## IRGCIS-IRIS Integration (GRIMS)

   a. IRGCIS Data in IRIS before World Bank (Phase I) Project

   b. Strategy of Data Migration to IRIS

   c. ICIS Applications in GRIMS

      i. Incoming germplasm registration (SHU Portal)

      ii. Germplasm distribution (seed export)

      iii. Inventory management

      iv. Seed list management

      v. Study

II. ## Design/ Implementation Issues and Solution

IRGCIS Data in IRIS
Before World Bank Phase I Project

- IRGC Accession
  - ~100,000 accessions
- Location information
  - Some data contains inconsistent info
  - donor location vs origin
- Names

- Due to the Data Quality Activity of WB project, the data in IRIS needs to be updated as well

Strategy of Data Migration to IRIS

1. Identification of accession generation information

   ○ Assignment of negative germplasm identifier to all available accession generation

2. Assignment of seed packet identifier (LOT ID)

   ○ Active
       • Bulk
       • Aluminum packet
       • Paper
   ○ Base
       • Aluminum can

Strategy of Data Migration to IRIS

1. Reformatting of passport data into IRIS standard, stored as germplasm ATTRIBUTE

    i. Defined passport descriptors in ICIS user defined field table (UDFLDS)

    ii. Transformed IRGCIS valid values for passport descriptors and for morpho-agronomic traits into ICIS SCALE, SCALEDIS, SCALECON

    iii. Created script that inserts and formats the IRGCIS passport data into ICIS ATTRIBUTE table

    ○ Still outstanding

    • Loading of Morpho-agronomic passport data in the DMS as STUDY

ICIS Applications for Data Management

- Curation of Incoming Data through IRIS
    1. Input incoming data using SetGen in a separate IRIS local database for incoming germplasm
    2. Upload the entire incoming local database to the central database (IRIS Database Administrator)
    3. Import to GRIMS database

- Listdata (trigger on update/ insert/delete to insert data automatically to IRGCIS)

INCOMING

GRC

SHU

IRIS CENTRAL

GRIMS
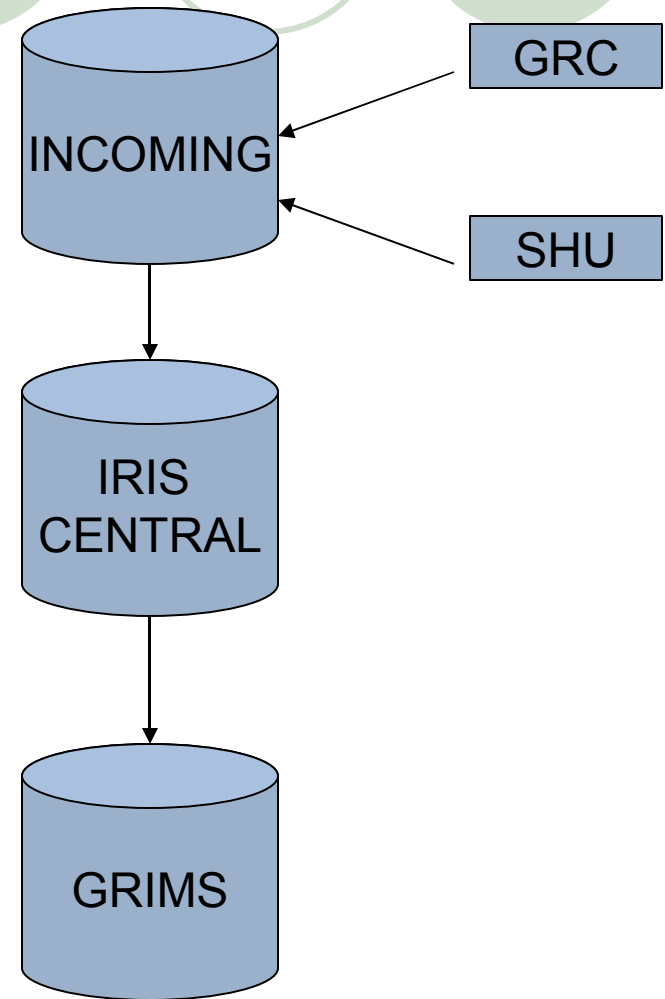
Integration to IRIS :

Germplasm distribution/ seed export

1. The request is made by accession number and the seed source (GID) is initially unknown

2. After finalizing the seed source and updating of the seed stocks, the seedlist is created which can be view

3. A seed export template is filled in by the user which is submitted to the Seed Health Unit.

Integration to IRIS:
Inventory Management

- IMS Lot

- IMS Transaction

Acc ID: **IRGC 28062**

Name: MUSHKAN 7

Produced: **1983DS**

MLS Germplasm

GR-0000227118-0646

Accession number

Name. Hierarchy: 1) Cultivar name, 2) Foreign accession number, and 3) Collector's number

Crop season the seed was produced

Multi-lateral system designation

**Barcode: GR-LOTID where GR-Genetic Resources, Lotid – system generated identifier for seed packet
Location - Tray number**
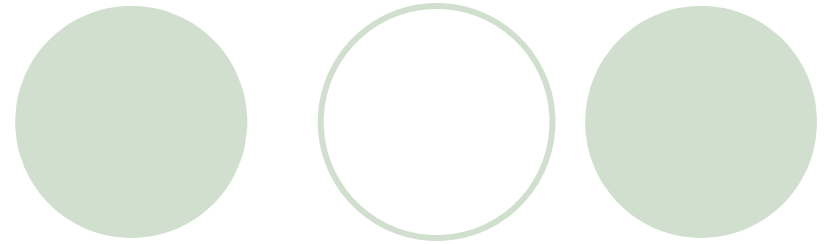
Integration to IRIS:

Seed list management

- The list is grouped according to process which is performed by GRIMS

- i.e.

  ○ Barcode Labels – generic printing of barcode labels

  ○ Initial Seed Increase – seed list used only by incoming samples for initial seed multiplication

  ○ Seed requests/ distribution

    • Already in production database

List Manager

List   List Entry   Edit   GMS   Batch   Help

List Selector   ✛ ⊠ ▬   B+ Bx B-   M

GRC BARCODE LABEL

☐ SetGen Lists
   ⊞ GRC BARCODE LABEL
   ⊞ GRC INCOMING BATCHES
   ⊞ GRC INITIAL SEED INCREASE
   ⊞ GRC SEED CHARACTERIZATION
     GRC SEED MANAGEMENT
     GRC SEED REJUVENATION
   ⊞ GRC SEED REQUESTS
   ⊞ GRC SEED STOCKS
     GRC VIABILITY MONITORING
   ⊞ INGER NOMINATION LIST
   ⊞ INGER NURSERY
   ⊞ SEED HEALTH UNIT
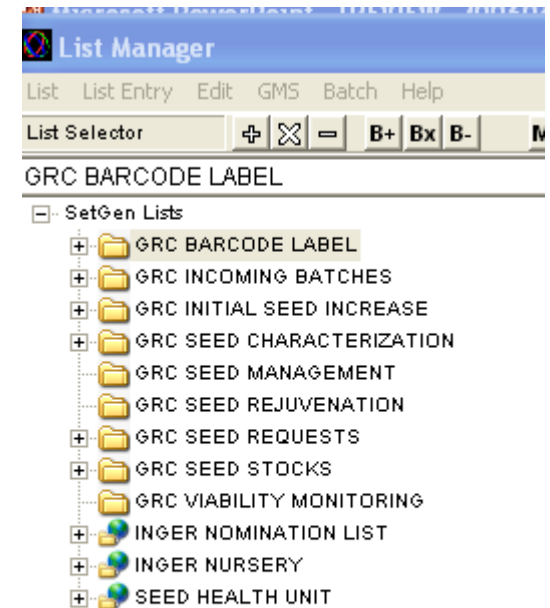
Implementation Issues & Solution

- Viability data is yet to be uploaded to the DMS

- Characterization data is already in DMS but the data is not yet used for selection of planting materials

- Germplasm taxonomy and mission information are not properly represented in ICIS schema.

Design Issues & Solution:
Selection of Planting Material for Characterization

- Importance of null value

- In DMS, null value means no row/ record

- Plant selection criteria for seed characterization is based on the following:

  1. Accessions with no single description

  2. Accessions with incomplete description

     → incomplete stage(s)

     → number of null traits

  3. Accessions for re-characterization/verification

     → range of accessions

     → morpho-agron traits

Design Issues & Solution
Selection of Planting Material for Characterization

- Considerable execution time if flat table is used
- E.g. Select O.sativa accessions with no specified traits (no descriptors)

```
select    a.accno,a.cropyear
from      morpho_agron1 a
where     get_sppcode(get_mgid(accno)) ='S'
          and     get_status_acc(accno) = "AV"
          and     a.pop_code =              'A'
          and    GET_NULL_TRAIT(a.accno,a.pop_code) = 45
```

Design Issues & Solution
Selection of Planting Material for Characterization

```
CREATE OR REPLACE FUNCTION Get_Null_Trait (xAccno
        NUMBER,xPop_Code VARCHAR2) RETURN NUMBER IS
cnumber NUMBER;
sppCode CHAR(1);
CURSOR cSat IS
SELECT (45 - (
                DECODE(hdg_80head,NULL,0,1) +
                DECODE(sdht_code,NULL,0,1) +
                DECODE(mat,NULL,0,1)     +
                DECODE(blpub_veg,NULL,0,1) +
                DECODE(blco_rev_veg,NULL,0,1) +
                DECODE(blsco_rev_veg,NULL,0,1) +
                DECODE(la,NULL,0,1) +
                DECODE(ligsh,NULL,0,1) +
                DECODE(cco_rev_veg,NULL,0,1) +
                DECODE(ligco_rev_veg,NULL,0,1) +
                DECODE(auco_rev_veg,NULL,0,1) +
                DECODE(cuan_repro,NULL,0,1) +
                DECODE(noco_rev,NULL,0,1) +
                DECODE(inco_rev_repro,NULL,0,1) +
                DECODE(cust_repro,NULL,0,1) +
                DECODE(fla_repro,NULL,0,1) +
                DECODE(pty,NULL,0,1) +
                DECODE(second_br_repro,NULL,0,1) +
                DECODE(pex_repro,NULL,0,1) +
                DECODE(pa_repro,NULL,0,1) +
```

- BEGIN
- sppCode:=Get_Spp_Code(xAccno);
- IF sppCode='G' THEN
-    OPEN cGlab;
-    FETCH cGlab INTO cnumber;
-    IF cGlab%NOTFOUND THEN
-       cnumber := 0;
-    END IF;
-    CLOSE cGlab;
- ELSIF sppCode='S' THEN
-       OPEN cSat;
-    FETCH cSat INTO cnumber;
-    IF cSat%NOTFOUND THEN
-       cnumber := 0;
-    END IF;
-    CLOSE cSat;
- ELSIF sppCode='W' THEN
-    cNumber:=0;
-   END IF;
- RETURN cnumber;
- END;
- /

Design Issues and Solutions:
Taxonomy information

- At the moment, GRIMS and IRGCIS uses the taxonomy number provided by IPGRI.

```
TAXNO                     NOT NULL NUMBER(5)
KINGDOM                   NOT NULL VARCHAR2(30)
DIVIPHYLUM                         VARCHAR2(30)
CLASS                              VARCHAR2(30)
ORDR                               VARCHAR2(30)
FAMILY                             VARCHAR2(30)
GENUS                     NOT NULL VARCHAR2(60)
GHYBRID                            VARCHAR2(1)
GAUTHOR                            VARCHAR2(100)
SUBGENUS                           VARCHAR2(30)
SECT                               VARCHAR2(30)
SER                                VARCHAR2(30)
SUBSERIES                          VARCHAR2(30)
SHYBRID                            VARCHAR2(1)
SPECIES                   NOT NULL VARCHAR2(60)
SAUTHOR                            VARCHAR2(100)
SUBSP                              VARCHAR2(30)
SSPAUTHOR                          VARCHAR2(100)
VARHYBRID                          VARCHAR2(1)
BVAR                               VARCHAR2(30)
VARAUTHOR                          VARCHAR2(100)
SVHYBRID                           VARCHAR2(1)
SUBVAR                             VARCHAR2(30)
FORMA                              VARCHAR2(70)
```

Design Issues and Solutions:
Collecting mission

- ## Mission

```
Name             Null?     Type          Comment
---------------  --------  ----          ----------
MISSION_CODE     NOT NULL  NUMBER(5)     Code given to a mission
INSTITID                   NUMBER(10)    Funding institute
START_DATE                 NUMBER(10)    Date when the mission started
END_DATE                   NUMBER(10)    Date when the collecting mission
                                         ended

MISSION_TYPE               VARCHAR2(5)

                                         Rescue collecting
                                         Immediate use
                                         Gap filling
                                         Research purpose
                                         Opportunistic reasons
CNTRYID                    NUMBER(10)    Location or country where the
                                         mission was held
```
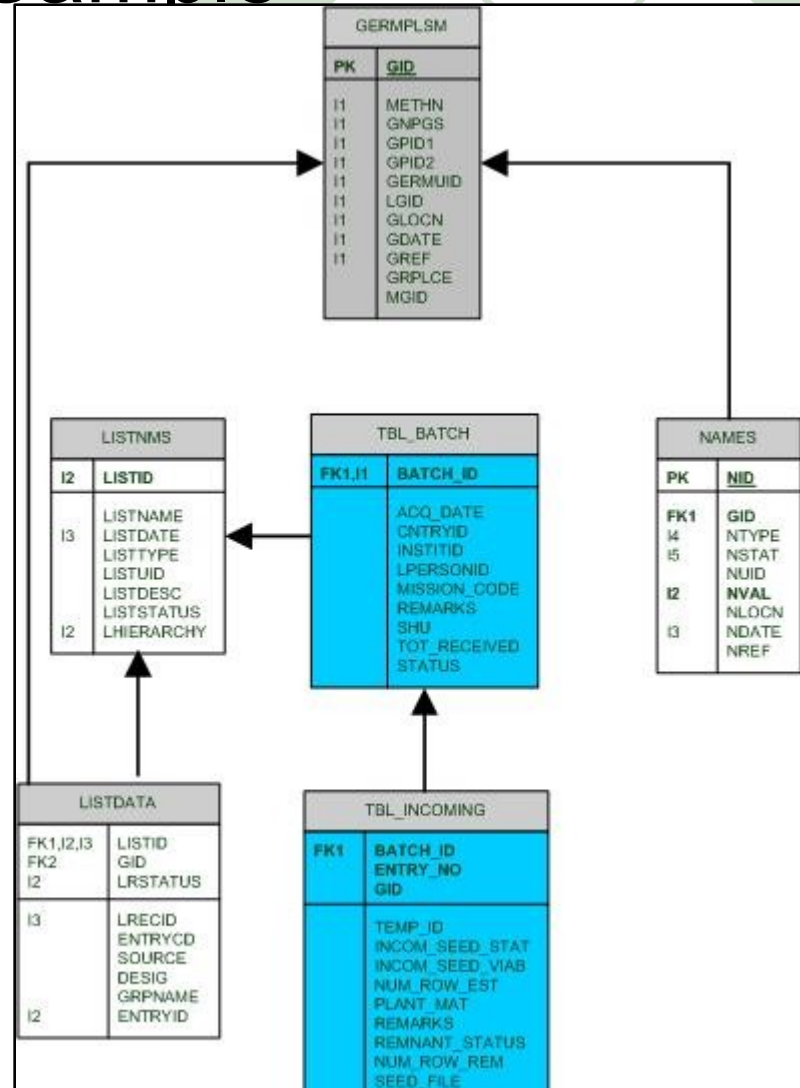
```
┌─────────────────────────┐
│       TBL_MISSION        │
├──────┬──────────────────┤
│ PK,I1│   MISSION_CODE    │
├──────┴──────────────────┤
│      INSTITID            │
│      START_DATE          │
│      END_DATE            │
│      MISSION_TYPE        │
│      CNTRYID             │
└─────────────────────────┘
           ▲
           │
           │
┌─────────────────────────┐
│     TBL_MISSION_COOP     │
├──────┬──────────────────┤
│ FK1  │   MISSION_CODE    │
│      │   LPERSONID       │
├──────┴──────────────────┤
│                          │
└─────────────────────────┘
```

- ## Mission Collectors

```
Name             Null?     Type          Comment
---------------  --------  ----          ----------
MISSION_CODE     NOT NULL  NUMBER(5)     Code given to a mission
PERSONID                   NUMBER(10)    Collector ID
```
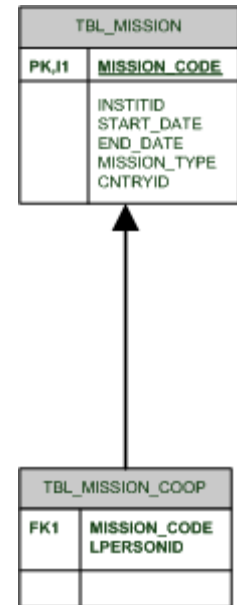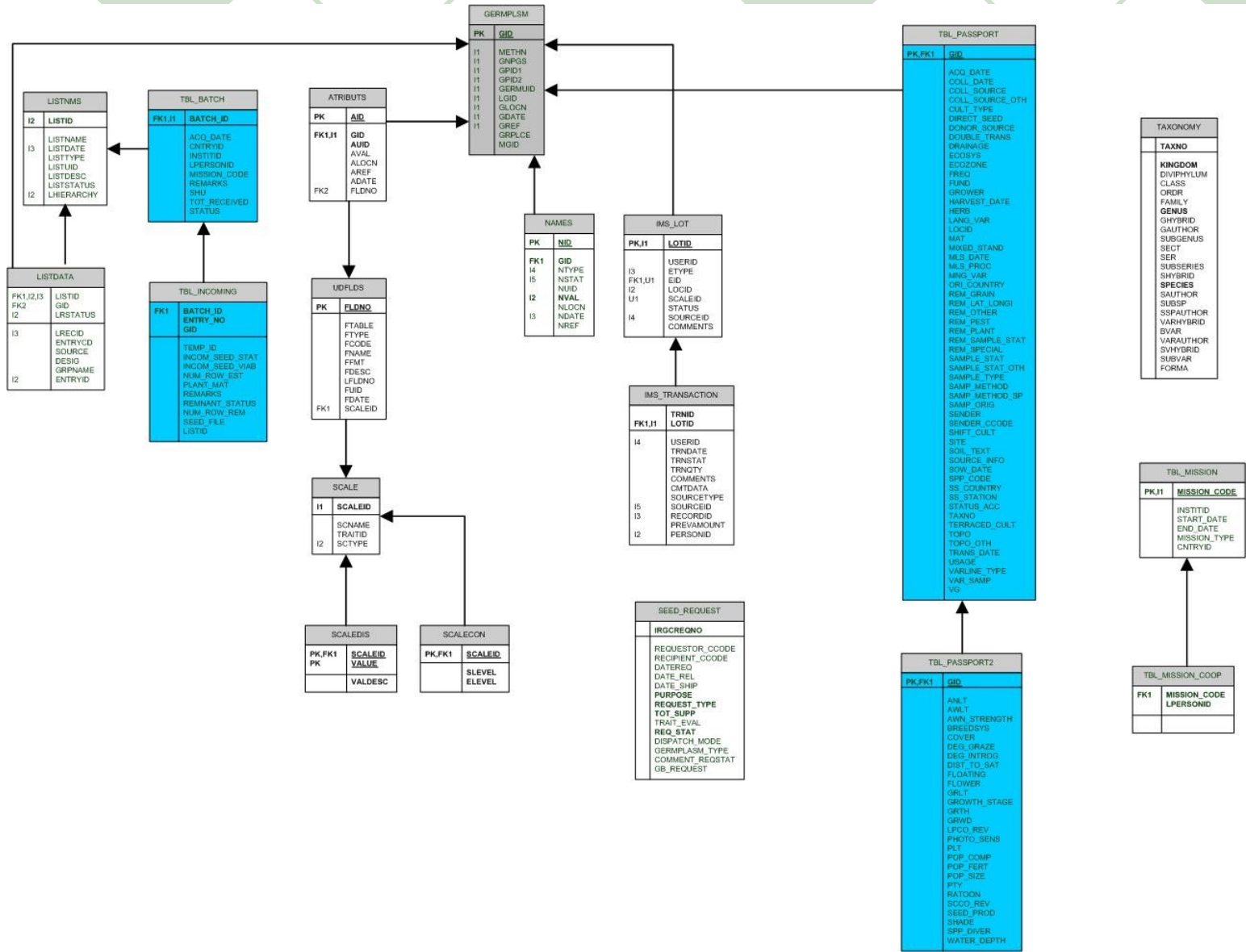
Integration to IRIS:
ERD for Incoming Sample

- Listname=
- BATCH_ID

IRIS/ GRIMS/ IRGCIS Integration

Summary and Conclusion

- Heavily used ICIS modules
  - GMS
    - GERMPLSM, NAMESM, LISTNMS, LISTDATA, UDFLDS
    - LOCATION
  - IMS
    - IMS_TRANSACTION, IMS_LOT
- DMS
  - SCALE, SCALEDIS, SCALECON, SCALETAB
  - For publication of data into IRIS Central database

Acknowledgement

GRC

Flora de Guzman

Adelaida Alcantara

Ato Reaño

Soccie Almazan

BBU/ CRIL

Arllet Portugal

William Eusebio

May Ann Sallan

Lilibeth Sison

Part 3: GRIMS GUI