Multi-User Development

Tony Goldstein
Systems Developer
Information Packaging
September 23, 1997

Single Developer Vs. Multiple Developers

The Single Developer	The Development Team
• Fewer issues • Limited scope of projects	• Requires coordination • Able to complete larger projects

Single User Development

The Single Developer
- Is all knowing
- Controls the application file
- Makes changes without affecting anyone

Multi-User Development

In contrast to the Single Developer, the Team must also consider:

- Only one person can work on an Application at a time
- Partial project knowledge
- One persons changes may affect many people

Factors in Team Development

- DLL Management
- Template Management
- Version Control
- Fault tracking
- Communication

DLL Management, Part I

Why DLLs?

- Each DLL is a separate APP file
- Multiple programmers on a single project
- Procedures called from separate executables have only one place to update

DLL Management, Part II

How to organize DLLs

- Function
- Responsibility
- Dependency

Organizing DLLs by Function

Group similar functions and procedures together

- Function location becomes intuitive
- Work becomes easier

Organizing DLL's by Responsibility

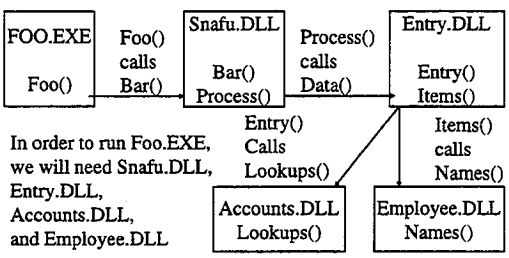
Group functions and procedures by developer

- Avoid multiple developers working in the same APP file
- Easier to locate programmers for problem resolution

Organizing DLLs to minimize Function Dependency

- Executables must load all DLLs referenced
- DLLs also must load all DLLs referenced
- Lack of organization leads to unnecessary dependencies

Function Dependency Illustrated



Some Useful Template Extensions

- Save Calls Generator
- Module Include Generator
- Global Data Definitions Generator

The Save Calls File

- Created every time the source is generated
- Lists all functions in the application
- Contains Prototype, Description, Created Date and Changed Date
- Shows Export Flag

Save Calls File: An Example

E SQLLongValue (String), Long
Description: Pass in a SQL statement which returns a Long.
Created: 10/30/96, 10:15 A.M.
Changed: 6/06/97, 3:30 P.M.

The Module Include File

- Created every time the source is generated
- Contains Function names and prototypes for all exported functions
- Eliminates the need to prototype functions in every app
- Automatically updated when prototypes change

Module Include File: An Example

```
Module('UTILITY.LIB')
  SQLLongValue(String), Long
  CheckSQLError(File),byte,proc
  MyCheckOpen(File, *Long)
  MyCheckClose(File, *Long)
  SQLStringValue(String),String
END
```

The Global Data Definition File

- Created every time the source is generated
- Include file contains Global Variable names and types, defined as External
- Eliminates the need to define Global Data in every application
- Automatically updated when data changes
- Prevents data corruption

Global Data Definition File: An Example

```
Glo:User    Byte,External,DLL(dll_mode)
Glo:Debug   String(1) ,External,DLL(dll_mode)
Glo:StrQ    Queue,Pre(),External,DLL(dll_mode)
StringQString   String(20)
    END
Glo:Zone    String(40) ,External,DLL(dll_mode)
```

Version Control

- Check Out / Check In
- Regularly archive and checkpoint the project
- Document and date changes

Template Management

- Central repository for templates
- Checkpoint templates along with the Project
- Changes to TopSpeed and Third Party templates must be well documented
- Limit template change privileges.

Fault Tracking

The Three stages of Fault Tracking

- Where's the problem?
- What's the problem?
- Keeping track of it all

A Fault Tracking Database

- Who reported the fault
- Notes on the fault
- Module and procedure of the fault
- Who has worked on the fault
- Additional developer notes on fault resolution

The Database Administrator

- Database changes affect multiple people
- Developer's knowledge of database use is limited
- DBA oversees all database changes