Layout of the Code

Generating the Schema

(1) From the sqlplus prompt, the schema is recreated by doing:

@pdbschema.sql

Schema: GinoSchema20040819.jpg

(2) From sqlplus, stored procs are replaced / compiled by doing:

@pdbprocedures.sql

Stored Procedures

(3) The perl module that wraps the stored procedures is generated by doing:

smartGen.pl > filename.pm

(I use >ORACLE_SP_DPF.pm)

There are class modules that are used by smartGen.pl:

OraArg, OraProc, OraFunc

Perl worker modules

(4) There are several perl modules that do most of the db management work:

LSFOutput (Navid) parses an LSF log file for the job statistics
PDBConnect (Alex) is the DB entrypoint
DPFProc provides access to a task process instance and it's datasets
PDBManage is the main module for DB updating and does all the heavy lifting
PDBPathManage formats filepaths and performs variable substitution vars are: $(RUN_NAME), $(TASK_NAME), $(TASK_PROCESS_NAME)
PDBMgmtTP manages a task process
PDBMgmtDS manages a dataset
PDBMgmtDSI manages a dataset instance

(5) A run of processing is scheduled into a task by calling CreateRun.pl

(6) DPFScheduler.pl looks for completed task processes and schedules the next for submission to batch.

(7) There are many test scripts that show how to use various modules, generally named testMODULE_NAME.pl

(8) A pipeline process must be wrapped using a clone of taskProcessTemplate.pl It gives access to dataset names and

Summary

Once a task is configured in the web front end, and the processes wrapped with a clone of taskProcessTemplate.pl, a run is scheduled by calling CreateRun.pl. DPFScheduler.pl is then called periodically to invoke each of the taskprocesses in turn, assuming their predecessors succeeded. Currently, I call DPFScheduler.pl manually so I can watch the batch farm and database between schedulings.