From Rich on 3/9/17:

Heilmeier's Catechism[[edit](https://en.wikipedia.org/w/index.php?title=George_H._Heilmeier&action=edit&section=2)]

A set of questions credited to Heilmeier that anyone proposing a research project or product development effort should be able to answer.[[5]](https://en.wikipedia.org/wiki/George_H._Heilmeier#cite_note-5)

* What are you trying to do? Articulate your objectives using absolutely no jargon.
* How is it done today, and what are the limits of current practice?
* What's new in your approach and why do you think it will be successful?
* Who cares? If you're successful, what difference will it make? What are the risks and the payoffs?
* How much will it cost? How long will it take? What are the midterm and final "exams" to check for success?

For the following, keep in mind there are new settings Model\_Mode can be set to for the integrated software:

* “MODSIM\_GSFLOW”
* “MODSIM\_PRMS”
* “MODSIM\_MODFLOW”

GSFLOW, PRMS, MODFLOW:

* Pass in model mode and enable the iteration time-step loop without MODSIM
* Compile PRMS as DLL (Steve, Eric, Enrique)
* Clean-up Fortran, specifically GSFLOW and various PRMS modules (Steve)
* Get PRMS-only Sagehen model running (Steve & Eric)
  + PRMS: Get lakes with multiple HRU’s running
* Add diversions to soil zone:
  + modsim2prms()
  + prms2modsim()
  + prms2mf()
  + sfr2prms()
  + wel2prms()
* Move budget/output routines inside GSFLOW\_PRMS() to proceed the RUN() for the next time step.
* 1st iteration of the 1st time step, all shared vectors are initialized to zero (except for the network mapping arrays)
* With Fortran MMF:
  + check that the RunTimePlots GUI for PRMS,
  + check to make sure the parameter tool works,
  + verify that RESTART() functionality still works
  + Long term goals for Fortran MMF:
    - Flexible dimensions
    - Multiple parameter files
    - NetCDF output support
  + Long term goal of allowing different time step lengths among the integrated codes.
* Add IUPSEG and IOUTSEG-like parameters to PRMS5
* Water-use option files need to be constructed so PRMS-only has them available. Use the PRMS format for this
* Does MODSIM still need to pass KPER and KSTP into the RUN() routines of GSFLOW (it looks like the various MODFLOW packages are passed this argument)? If not, consider removing from MODSIM communications with GSFLOW, just let the GSFLOW communications handle this.

MODSIM:

* Pass MODSIM diversions and releases to MODFLOW
* Pass MODFLOW accretion/depletion to MODSIM
* Pass in an Advance File Reader (AFR) flag
* Pass in “mapping” variables which are arrays of integers
* Pass back from GSFLOW the simulation\_start\_time and overwrite in MODSIM (this is different than data\_start\_time). MODSIM will need to ensure simulation\_start\_time and data\_start\_time make sense.
* Chunk-up to GSFLOW\_PRMS() into an INIT(), RUN(), & CLEAN() accommodate calls from MODSIM

Misc:

* Documentation
* Compare results from the original version of Sagehen (i.e., GSFLOW-only, no lakes) using the published GSFLOW code and the Fortran version of MMF GSFLOW code. This is to make sure that

**Accomplishments at 1st code sprint (11/2/16-11/4/2016, Portland, OR)**

* Review Deschutes GSFLOW model
  + Provided technical support
  + NHM parameters transferred to his model (GIS overlay)
* Shared GSFLOW-MODSIM developments with:
  + OR WSC center director
  + State representative
* Jen and Bob presented the Deschutes MODSIM model (via webex):
* Got repos set-up
  + Source codes organized
  + Started merging development version of MF and GSFLOW into working version
  + Started merging PRMS5 into GSFLOW
* Got started on working on Sagehen 3 Layer model

**Accomplishments at 2nd code sprint (3/7/2017-3/10/2017, Aptos, CA):**

* *3/7/2017:*
  + Got started with repos on Monday
  + Merged various codes and made sure they compiled
  + Enrique and Eric started learning MMF routines
    - Started mimicking MMF/GSFLOW (C/Fortran) interfaces in simple examples
* *3/8/2017:*
  + Started working out mixed language communications with simple VB/C example projects in Vis Studio
  + Plan A: Attempted to wrap a C++ wrapper around the GSFLOW static library (\*.lib) file and the Fortran exe
  + 10pm quit time after failing to sort out Visual Studio’s use of underscores in naming convention of \*.dll and \*.lib files (Name of the project that we were experimenting with is located the GSFLOW\_examples repository and is called
  + More work on merging branches in repos (All)
  + Steve worked on compiling
    - PRMS5
    - Fortran version of MMF
    - Convert multiple parameter files into 1 (Sagehen example)
    - Got 128 HRU and “3 Layer” PRMS models going

* *3/9/2017*
  + Made this document
  + Worked on file that maps MODSIM link information to MODFLOW/PRMS isegs
  + Steve continued developing Fortran MMF
  + Group discussion on code architecture
    - Which terms should remain in GSFLOW control file and which should be added to MODSIM custom code
    - Rich developed new GSFLOW functions for preparing aggregated flux for transfer between MODSIM and GSFLOW
  + Discussed the development of stream network mapping in PRMS. Needs to be something similar SFR approach
  + Discussed order of calls in various GSFLOW functions, and between MODSIM/GSFLOW

**Accomplishments at 3rd code sprint (5/26/2017, Fort Collins, CO):**

* *5/26/2017*
  + 3rd code sprint held in Fort Collins, Research Triangle Institute offices on Harmony Road
  + Spent some time on getting the GSFLOW code to recompile. There were a few issues that crept up owing to SourceTree conflicts. Got these resolved so that code once again compiles (this is referring to the MODSIM branch of GSFLOW).
  + Also, Steve was able to get the MODSIM branch of the GSFLOW code to run the 3 layer sagehen model again.
  + About lunch time we got the latest C# custom code to ‘step into’ the GSFLOW fortran code. That is, the custom code was able to call GSFLOW\_PRMS().
  + To do this, we set up the solution found here:
    - \gsflow.git\msds\gsflow\_develop\_f\DLL\_GSFLOW\GSFLOW\_MODSIM\

Which contains both the C# custom code project and the GSFLOW fortran project within the solution. These files have been pushed to the repo, I believe.

* + After getting the code to run the model a number of small issues kept arising that mostly required Steve’s expertise with the GSFLOW code for us to work through. Steve successfully fixed most such issues that caused the GSFLOW-side of the GSFLOW-MODSIM integration to crash.
  + Near the end of the day, we were running the Sagehen model by calling pieces of GSFLOW from within MODSIM’s custom code (specifically the parts of the code relating to Initialize() and ReadStress() routines). However, at this point we are not sending in diversions or reservoir release amounts.
  + One thing that Steve wanted to try at the end of the day, but we didn’t quite have enough time, was to run the Sagehen 3 layer model in PRMS-only mode (no MODSIM). This might one thing we can work on for next time.
* 6/15/2017
  + Reviewed status of Sagehen MODSIM input file with Enrique for about 45 minutes. Left a few notes in the annotations of the .xy file. Seems it is ready to go.

**Accomplishments during 4th code sprint (7/5/2017, Webex, 12:15-5:05 PM Pacific Time Zone):**

* *7/05/2017*
  + Things to check on in the code:
    - Make sure that the model start time in the GSFLOW control file is returned to MODSIM and that MODSIM sets model start time for its own internal use correctly.
  + Discussed ‘shared variable handling.’ This is the idea that C# might gain access to a fortran module so that shared values no longer need to passed back and forth. Instead, each code would read/write as needed to a variable and when the program flow in another part of the program flow needs to access that part of the code, it could do it without variable passing. (Eric will look into this).
  + Also, from Steve, model mode variable can now have the following values:
    - 0 = GSFLOW
    - 1 = PRMS-only
    - 2 = MODFLOW-only
    - 4,6-10 = special PRMS-only modes
    - 11 = MODSIM-GSFLOW
    - 12 = MODSIM-PRMS
    - 13 = MODSIM-MODFLOW
    - 14 = MODSIM
  + Something that was brought up during the sprint by Steve is that he doesn’t need to keep track of Lake values in PRMS-MODSIM mode. I am (Eric) confused by this, and am noting it here in case we need to revisit this notion. As we left it, when running GSFLOW-MODSIM or MODFLOW-MODSIM modes, MODFLOW will return lake volumes to MODSIM and MODSIM will update accordingly. It seems that when running in PRMS-MODSIM mode, MODSIM will be responsible for maintaining lake volumes, which is fine, but we will need to make sure the code reflects this distinction.
  + Enrique proposed a gsflow\_prmsSettings() call to the GSFLOW control file at the outset of the simulation to get the name of the MODSIM input file (.xy file extension).
  + Enrique will work on reading the mapping file containing the mapping information between the MODSIM links and SFR segments, we didn’t discuss if we also need mapping information for PRMS and will need to follow up on this.
  + Rich is going to tweak SFR2MODSIM and LAK2MODSIM functions for returning information needed by MODSIM
  + Steve is going to work on summary functions for returning values to MODSIM
  + Eric is going to work with Steve on passing arrays back-and-forth between the codes after exploring whether or not C# can access a fortran module.
  + Eric is going to work on scheduling the next code sprint.

**Accomplishments during 5th code sprint (7/21/2017, Webex, 8:00-5:15 PM Pacific Time Zone):**

* *7/21/2017*
  + Today’s code sprint largely centered on nuts & bolts programming to pass variables back-and-forth between the two codes.
  + Schedule start time was 8:15 AM pacific, but as Enrique had another meeting scheduled at 9:30 mtn (8:30 pacific), he and I got started a little early so that he could wrap up pre-sprint adjustments to the C# that include processing of the mapping file and xy file now being passed back from the GSFLOW control file per Steve’s pre-meeting adjustments.
  + Throughout most of the meeting, we were mostly focused on getting GSFLOW-MODSIM to run, leaving the testing of other modes to post-sprint.
  + Much of the meeting was focused on dynamic setting of array sizes to be passed back-and-forth between the two codes. This is a more robust approach that hard-coding arbitrarily-large arrays.
  + By the end of the sprint, variables were being passed between the two codes.
  + Before checking-in the modified code (all on Eric’s machine), ensured that the code compiled OK.
  + Important to emphasize that while this may not sound like a lot of advancement for an entire day’s worth of collaboration, many of the coding issues that were overcome during the day’s efforts will hopefully mean we don’t need to redo things ‘the correct way’ in the future. That is, we tried to take as robust approach the first time around as we could.
  + Next code sprint scheduled for 8/11/2017 in the AM

**Accomplishments during 6th code sprint (8/11/2017, Webex, 8:00-11:30 PM Pacific Time Zone):**

* *8/11/2017*
  + *Initially started by getting code to compile on everyone’s machine.*
  + *As Enrique is unable to compile the Fortran source code, and was noticing that he wasn’t getting flows (GW-SW exchanges) passed back from GSFLOW DLL, we worked on getting that problem fixed on our machines and once that was done, we pushed the updated DLL to the repo for Enrique to work with.*
  + *We spent an hour or adjusting the source code to account for steady state stress periods that are appended to the front end of a GSFLOW model when run in MODSIM-GSFLOW mode. The issue is that GSFLOW will run the SS model to get initial heads for the transient run without having run the MODSIM solver. We coded it in such a way that the MODSIM solver won’t attempt to run until after the SS stress period. We checked this for MODSIM-GSFLOW mode, but need to check that this order of operations is followed in MODSIM-MODFLOW mode as well.*
  + *Got rid of the IDIVERT Variable (or need to get rid of it if we didn’t finish this)*
  + *Discussed whether lake stage or lake volume was needed in MODSIM to check for convergence.*
  + *After Enrique got off the phone (about 12 pm mtn time), conversation continued with Rich, Steve, and Eric:*
    - *Rich showed us the ‘Ag-Options’ module he is working on in MF-NWT for eventual incorporation into GSFLOW. The idea is that SFR and WEL water would be applied to the canopy layer (tank) in GSFLOW. Steve described how this water would then get passed down into the 3 soil-zone tanks (1. capillary tank, 2. gravity tank, and 3. preferential flow tank)*
    - *Eric mentioned an upcoming meeting with BOR folks in Boulder City, NV, regarding their simulation of the Upper Colorado using RiverWare. Dan Bright, deputy director of the NVWSC set the meeting up to discover how we (USGS) might better serve their needs by finding out what they are doing and also to share with them what we are already doing with MODSIM-GSFLOW. That meeting is on the 17th.*
    - *Eric solicited Rich and Steve for ideas on how we might amend (or better construct) Table 1 is the review paper planned for debuting MODSIM-GSFLOW. Should summarize the suggested columns in an email to Enrique and solicit for his feedback as well.*
  + *Ended the meeting with Eric needing to follow-up with Enrique on (1) determination of model convergence, (2) use of commensurate units in GSFLOW and MODSIM, and (3) use of lake volume not stage in MODSIM*
* **Accomplishments during 1st mini-sprint code sprint (8/25/2017, Webex, 8:00-11:o0 PM Pacific Time Zone):**
  + *Initially, the idea for this code sprint was for Enrique and Eric to more appropriately “size” the MODSIM network’s attributes (reservoir size, link capacities, diversion amounts). However, upon trying to compile the GSFLOW project, it wouldn’t compile. Eric asked Steve and Rich to join us on the call (and webex) and we spent most of this sprint instead focused on getting the latest code to compile.*
  + *Eric also worked in the new MF-NWT 1.1.3 source files into his local solution.*
* **Accomplishments during 2nd mini-sprint code sprint (8/31/2017, Webex, 8:00-10:00 PM Pacific Time Zone):**
  + *Enrique and Eric focused on sizing the link capacities and diversion amounts in the MODSIM .xy input file such that periods of stress and abundance were experienced evenly across the network. The next major issue is to address units between the two models.*