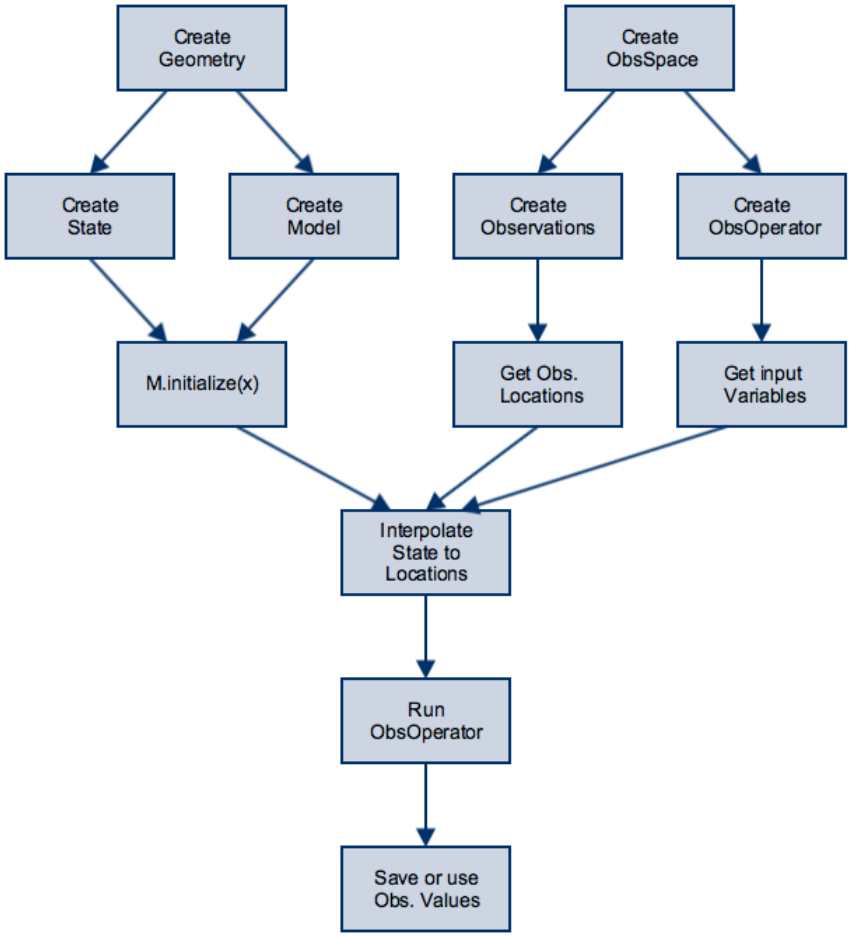
Abstract UFO

Data Flow



UFO Abstract Code

UFO

```
  int execute(const eckit::Configuration & fullConfig) const {

//  Setup observation window
    const eckit::LocalConfiguration windowConf(fullConfig, "assimilation_window");
    const util::Duration winlen(windowConf.getString("window_length"));
    const util::DateTime winbgn(windowConf.getString("window_begin"));
    const util::DateTime winend(winbgn + winlen);
    Log::info() << "Observation window is:" << windowConf << std::endl;

//  Setup resolution
    const eckit::LocalConfiguration resolConfig(fullConfig, "resolution");
    const Geometry_ resol(resolConfig);

//  Setup Model
    const eckit::LocalConfiguration modelConfig(fullConfig, "model");
    const Model_ model(resol, modelConfig);

//  Setup State and Model
    const eckit::LocalConfiguration bgConf(fullConfig, "state");
    Log::info() << "Input state configuration is:" << bgConf << std::endl;
    State_ xx(resol, bgConf);
    model.initialize(xx);
    Log::test() << "Input state: " << xx.norm() << std::endl;

//  Setup observations and obs operator
    const eckit::LocalConfiguration obsConf(fullConfig, "Observation");
    Log::info() << "Observation configuration is:" << obsConf << std::endl;

    ObsSpace_ obsdb(obsConf, winbgn, winend);
    Observations_ yobs(obsdb);
    ObsOperator_ hop(obsdb, obsConf);

//  Setup obs bias
    eckit::LocalConfiguration biasConf;
    fullConfig.get("ObsBias", biasConf);
    ObsAuxCtrl_ ybias(biasConf);

//  Create data structure for interpolated model fields.
    ModelAtLocations_ xxAtLocs(obsdb, hop.variables(), winbgn, winend, resol);

//  Get locations info for interpolator
    Locations_ locs(obsdb, winbgn, winend);

//  Interpolate state variables to obs locations
    xx.interpolate(locs, xxAtLocs);

//  Compute Obs Equivalent
    hop.obsEquiv(xxAtLocs, yobs.obsvalues(), ybias);

//  Save H(x)
    Log::test() << "UFO: H(x): " << yobs << std::endl;
    yobs.save("UFO");

    return 0;
  }
```

UFO Trace

Example output from log file:

Log

```
OOPS_TRACE Geometry<MODEL>::Geometry starting
```

```
OOPS_TRACE Geometry<MODEL>::Geometry done

OOPS_TRACE Model<MODEL>::Model starting
OOPS_TRACE QgModel::QgModel
OOPS_TRACE QgModel created
OOPS_TRACE Model<MODEL>::Model done

OOPS_TRACE State<MODEL>::State read starting
OOPS_TRACE QgState::QgState created and read in.
OOPS_TRACE State<MODEL>::State read done

OOPS_TRACE Model<MODEL>::initialize starting
OOPS_TRACE QgState activated for Model
OOPS_TRACE Model<MODEL>::initialize done

OOPS_TRACE State<MODEL>::norm starting
OOPS_TRACE State<MODEL>::norm done

OOPS_TRACE ObservationSpace<MODEL>::ObservationSpace starting
OOPS_TRACE ObsSpaceQG: Obs files are: Data/qg.truth3d.obt and Data/qg.ufo2.obt
OOPS_TRACE ObsSpaceQG::getHelper: Opening Data/qg.truth3d.obtData/qg.ufo2.obt
OOPS_TRACE ObsHelpQG constructed
OOPS_TRACE ObsSpaceQG created, count=1
OOPS_TRACE ObservationSpace<MODEL>::ObservationSpace done

OOPS_TRACE ObsVector<MODEL>::ObsVector starting
OOPS_TRACE ObsVector<MODEL>::ObsVector done
OOPS_TRACE Observations created

OOPS_TRACE ObsOperator<MODEL>::ObsOperator starting
OOPS_TRACE QG ObsFactory ObsType =Stream
OOPS_TRACE ObsStreamQG created Stream
OOPS_TRACE ObsOperator<MODEL>::ObsOperator done

OOPS_TRACE ObsAuxControl<MODEL>::ObsAuxControl starting
OOPS_TRACE ObsAuxControl<MODEL>::ObsAuxControl done

OOPS_TRACE ObsOperator<MODEL>::variables starting
OOPS_TRACE Variables<MODEL>::Variables shared_ptr done
OOPS_TRACE ObsOperator<MODEL>::variables done

OOPS_TRACE ModelAtLocations<MODEL>::ModelAtLocations starting
OOPS_TRACE ModelAtLocations<MODEL>::ModelAtLocations done

OOPS_TRACE Variables<MODEL>::~Variables starting
OOPS_TRACE Variables<MODEL>::~Variables done

OOPS_TRACE Locations<MODEL>::Locations starting
OOPS_TRACE Locations<MODEL>::Locations done

OOPS_TRACE State<MODEL>::interpolate starting
OOPS_TRACE State<MODEL>::interpolate done

OOPS_TRACE ObsOperator<MODEL>::obsEquiv starting
OOPS_TRACE ObsOperator<MODEL>::obsEquiv done

OOPS_TRACE ObsVector<MODEL>::print starting
OOPS_TRACE ObsVector<MODEL>::print done

OOPS_TRACE ObsVector<MODEL>::save startingObsHelpQG:putdb obsname = Stream, col = UFO
OOPS_TRACE ObsVector<MODEL>::save done

OOPS_TRACE Locations<MODEL>::~Locations starting
OOPS_TRACE Locations<MODEL>::~Locations done
OOPS_TRACE ModelAtLocations<MODEL>::~ModelAtLocations starting
OOPS_TRACE ModelAtLocations<MODEL>::~ModelAtLocations done
OOPS_TRACE ObsAuxControl<MODEL>::~ObsAuxControl starting
OOPS_TRACE ObsAuxControl<MODEL>::~ObsAuxControl done
OOPS_TRACE ObsOperator<MODEL>::~ObsOperator starting
OOPS_TRACE ObsOperator<MODEL>::~ObsOperator done
OOPS_TRACE Observations destructed
```

```
OOPS_TRACE ObsVector<MODEL>::~ObsVector starting
OOPS_TRACE ObsVector<MODEL>::~ObsVector done
OOPS_TRACE ObservationSpace<MODEL>::~ObservationSpace
OOPS_TRACE ObsSpaceQG cleared, count=0
OOPS_TRACE ObsHelpQG destructed
OOPS_TRACE State<MODEL>::~State starting
OOPS_TRACE QgState::QgState destructed.
OOPS_TRACE State<MODEL>::~State done
OOPS_TRACE Model<MODEL>::~Model starting
OOPS_TRACE QgModel destructed
OOPS_TRACE Model<MODEL>::~Model done
OOPS_TRACE Geometry<MODEL>::~Geometry starting
OOPS_TRACE Geometry<MODEL>::~Geometry done
```

The sequence of Fortran calls generated by the UFO is the following:

Fortran sequence

```
! Declarations should go here

! Create state-related objects
call geom%create(resol_conf)
call model%create(geom, model_conf)
call xx%create(geom, bg_conf)
call model%initialize(xx)

! Create observations-related objects
call obspace%create(obs_conf, winbgn, winend)
call obs%create(obspace)
call hop%create(obspace, obs_conf)
call obias%create(bias_conf)

! Create "x<->y intermediate" objects
vars = hop%input_variables()
call x_at_obs%create(obspace, winbgn, winend, geom, vars)
call vars%delete()
call locs%create(obspace, winbgn, winend)

! UFO itself
call xx%interpolate(locs, x_at_obs)
call hop%h_oper(x_at_obs, obs, obias)
call obs%write("UFO")

! Clean-up
call locs%delete()
call x_at_obs%delete()
call obias%delete()
call hop%delete()
call obs%delete()
call obspace%delete()
call xx%delete()
call model%delete()
call geom%delete()
```