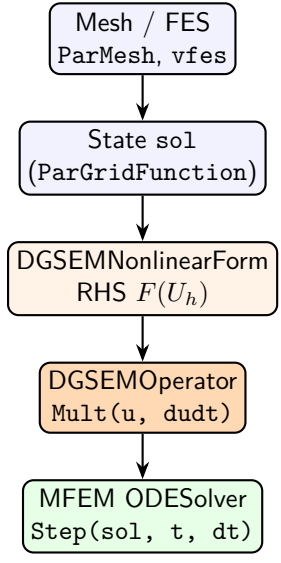
Prandtl: Current Architecture (High-Level)

Major constructs

- `ParMesh`, `ParFiniteElementSpace`
  mesh + DG space (`vfes`, `fes0`)
- `ParGridFunction sol`
  conserved state U_h on the mesh
- `DGSEMNonlinearForm`
  DG RHS $F(U_h)$
- `DGSEMOperator`
  wraps `DGSEMNonlinearForm` as a `TimeDependentOperator` (`Mult(u, dudt)`)
- MFEM `ODESolver`
  advances `sol`: `Step(sol, t, dt)`



Prandtl Development Plan: LTE on GPU

- **EOS→LTE Capability**
  - Inject meaningful ideal gas tests, then
  - Centralize thermo in a new Gas Model/EOS interface (contract)
  - Implement first: ideal-gas EOS behind that interface
  - Later: plan LTE modifications/ interface update
- **Device readiness**
  - Harden DGSEM operator stack: `DGSEMOperator` (ODE wrapper) vs. `DGSEMNonlinearForm` (DGSEM discretization)
  - Refactor `DGSEMNonlinearForm` into MFEM integrators (volume, faces, viscous, boundary)
  - Introduce device/PA-capable variants of each integrator
- **Coordination / Gotchas**
  - Freeze state layout; avoid hard-coded indices
  - No EOS logic (or Mutation++) calls inside DGSEM kernels
  - Virtual dispatch only at top level (not in inner loops)
  - Integration testing to avoid divergence between GPU and EOS/LTE

Clean DGSEM layers: leadup work

- **DGSEMOperator**
  - Thin `MFEM::TimeDependentOperator`: axisym reweighting, FV blending, entropy/gradients (!fluxes)
  - *Does not* own low-level physics (no element/face loops).
- **DGSEMNonlinearForm**
  - Owns all volume and face contributions.
  - Is the single home for the PDE spatial discretization (no physics hiding elsewhere)

- Split `DGSEMNonlinearForm::Mult` into:
  - volume inviscid part,
  - viscous part,
  - interior face terms,
  - boundary face terms,
  - later: LTE-related source terms.

- Ensure `AddBdrFaceIntegrator` (and friends) cleanly forward into `DGSEMNonlinearForm`.
- Avoid smuggling EOS logic into BC classes: they should only use the gas model interface.

GPU-enabling - 1/2

- Move heavy DGSEM work into device-friendly kernels
- Lead-up work already done:
  - Clear split: `DGSEMOperator` vs `DGSEMNonlinearForm`
  - Integrators own all physics/system-specific volume / face ops
- Follow MFEM's GPU patterns:
  - Per-element tensor-product kernels for DGSEM volume terms
  - Per-face kernels for numerical fluxes (interior + boundary)

GPU-enabling - 2/2

- Device santize loops & kernels
  - No `new/delete` or STL in element / face loops
  - No virtual calls in quadrature loops (incl 3rd party)
  - No system calls (3rd party, `std::cout`, etc.)
- Add PA / device variants
  - For each key construct in `DGSEMNonlinearForm`:
    - Add MFEM PA-style assembly (`AssemblePA`, `AddMultPA`).
    - Retain original/CPU version for debugging and reference.
  - Use MFEM restrictions / element-batch layouts instead of ad-hoc loops.