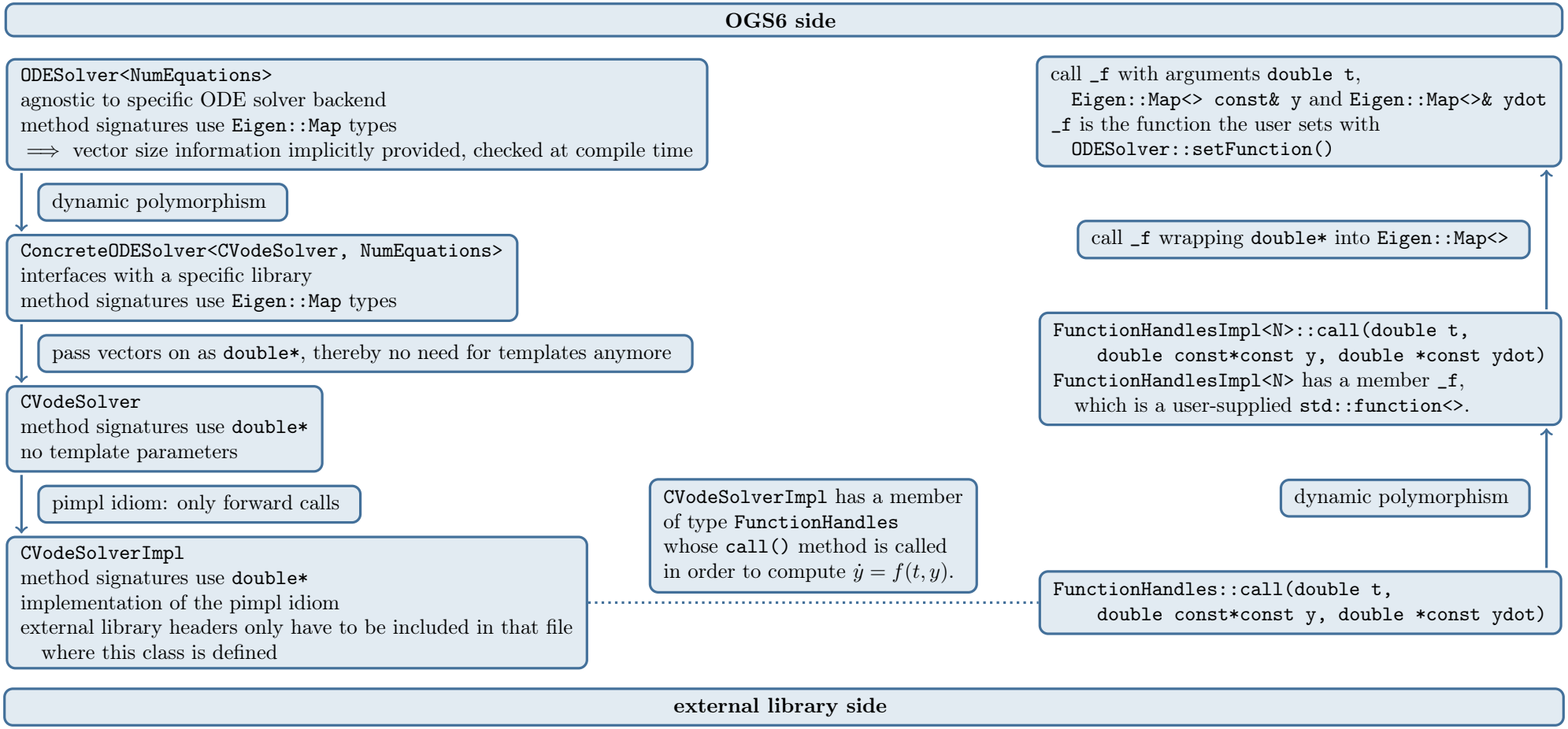
OGS6 side
ODESolver<NumEquations>
agnostic to specific ODE solver backend
method signatures use Eigen::Map types
⟹ vector size information implicitly provided, checked at compile time
dynamic polymorphism
ConcreteODESolver<CVodeSolver, NumEquations>
interfaces with a specific library
method signatures use Eigen::Map types
pass vectors on as double*, thereby no need for templates anymore
CVodeSolver
method signatures use double*
no template parameters
pimpl idiom: only forward calls
CVodeSolverImpl
method signatures use double*
implementation of the pimpl idiom
external library headers only have to be included in that file where this class is defined
CVodeSolverImpl has a member of type FunctionHandles whose call() method is called in order to compute ẏ = f(t, y).
call _f with arguments double t, Eigen::Map<> const& y and Eigen::Map<>& ydot
_f is the function the user sets with ODESolver::setFunction()
call _f wrapping double* into Eigen::Map<>
FunctionHandlesImpl<N>::call(double t, double const*const y, double *const ydot)
FunctionHandlesImpl<N> has a member _f, which is a user-supplied std::function<>.
dynamic polymorphism
FunctionHandles::call(double t, double const*const y, double *const ydot)
external library side