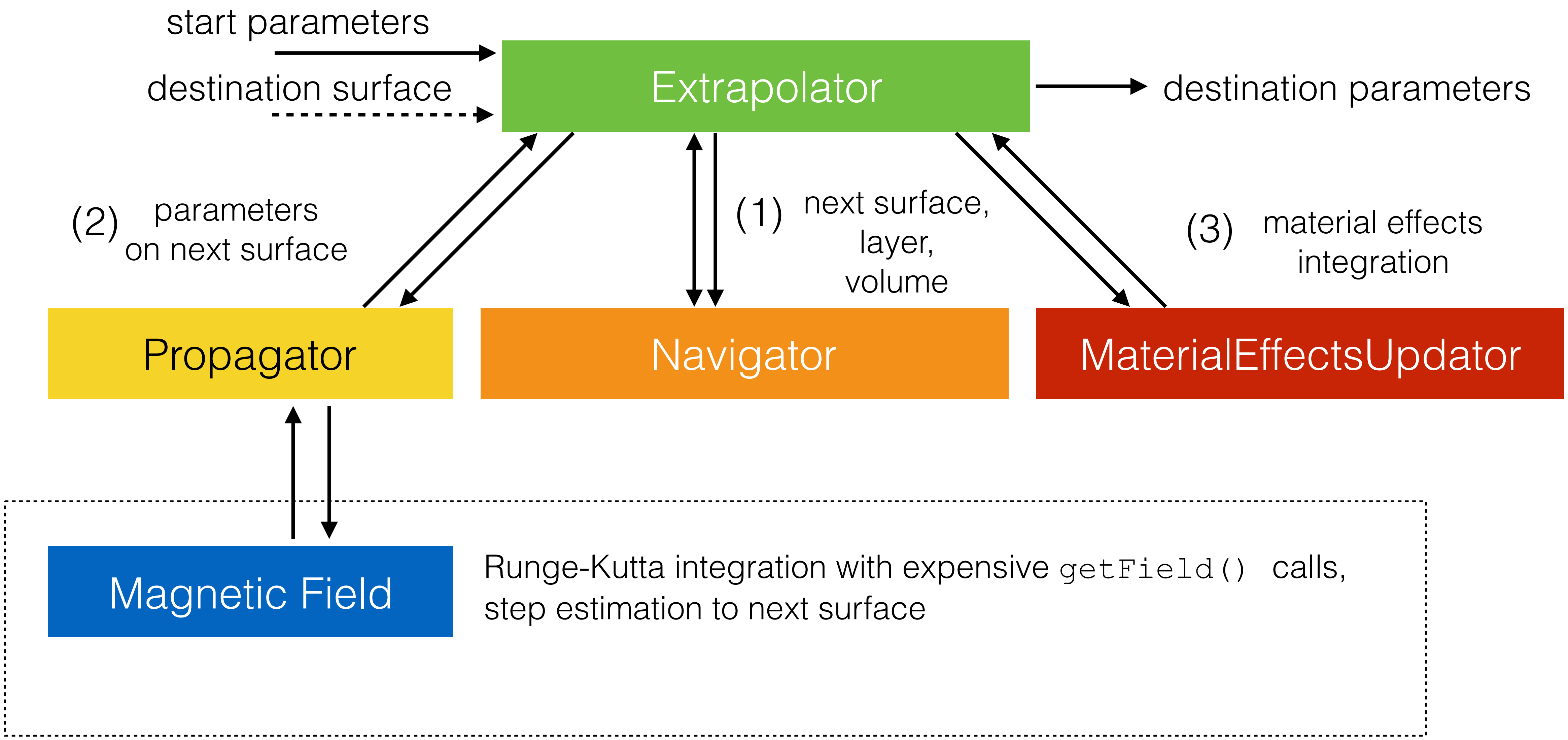
Propagator - Update

A. Salzburger

Reminder: ATLAS Propagator/Extrapolator setup


start parameters

destination surface

Extrapolator

destination parameters

(2) parameters on next surface

(1) next surface, layer, volume

(3) material effects integration

Propagator

Navigator

MaterialEffectsUpdator

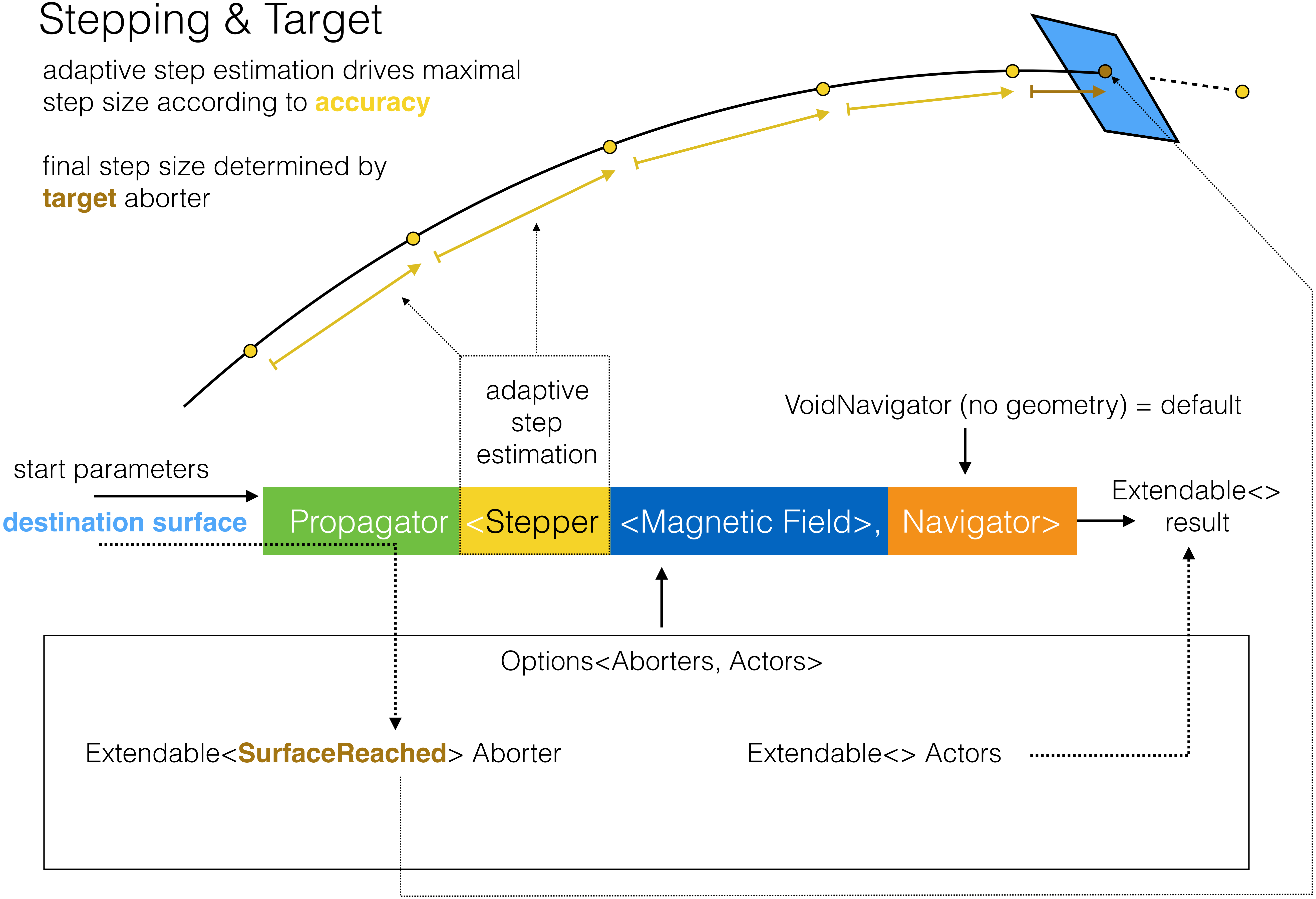
Magnetic Field

Runge-Kutta integration with expensive `getField()` calls, step estimation to next surface

Stepping & Target

adaptive step estimation drives maximal step size according to **accuracy**

final step size determined by **target** aborter



adaptive step estimation

VoidNavigator (no geometry) = default

start parameters

destination surface

Propagator <Stepper <Magnetic Field>, Navigator>

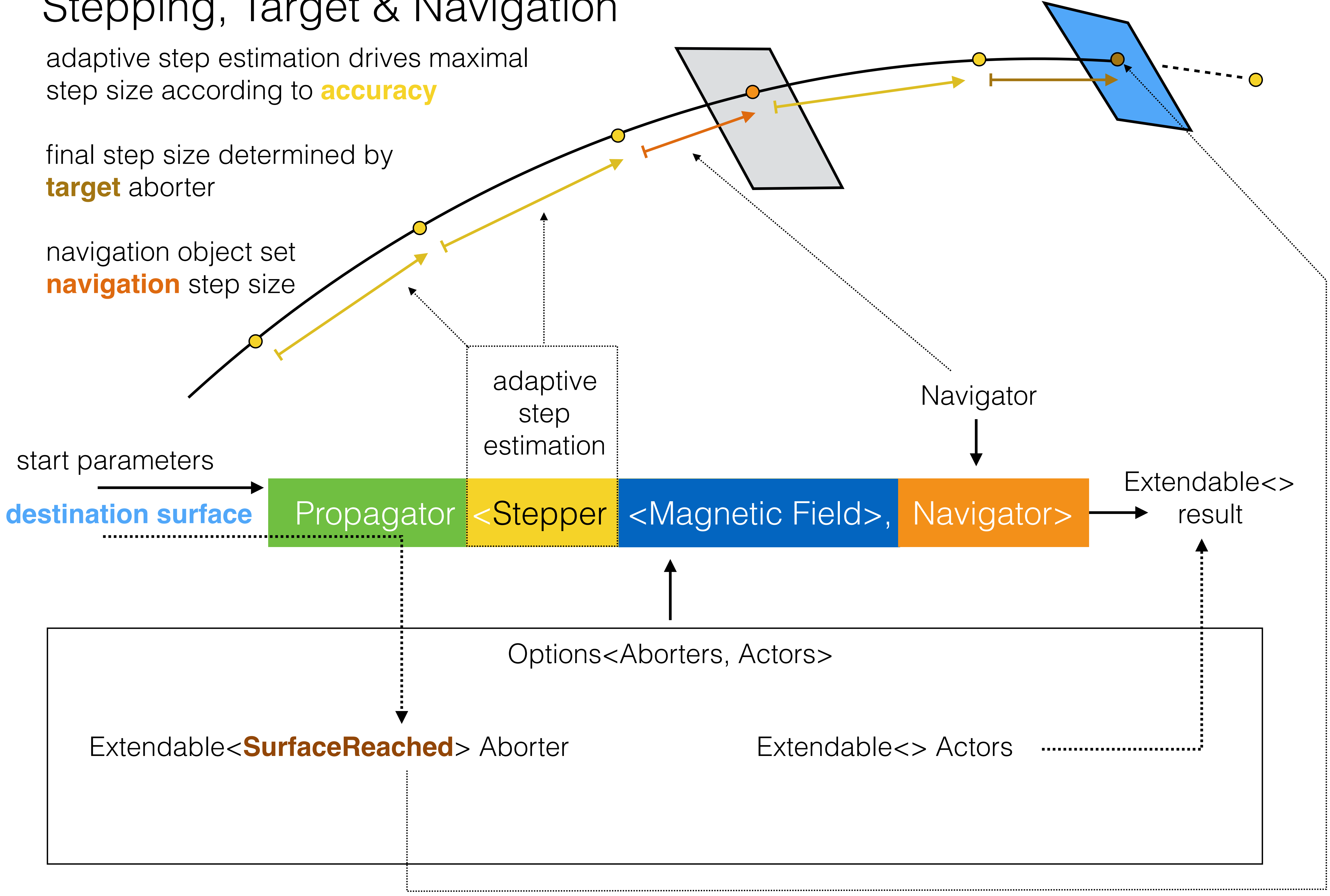
Extendable<> result

Options<Aborters, Actors>

Extendable<**SurfaceReached**> Aborter

Extendable<> Actors

Stepping, Target & Navigation



adaptive step estimation drives maximal step size according to **accuracy**

final step size determined by **target** aborter

navigation object set **navigation** step size

adaptive step estimation

Navigator

start parameters

destination surface

Propagator <Stepper <Magnetic Field>, Navigator>

Extendable<> result

Options<Aborters, Actors>

Extendable<**SurfaceReached**> Aborter

Extendable<> Actors

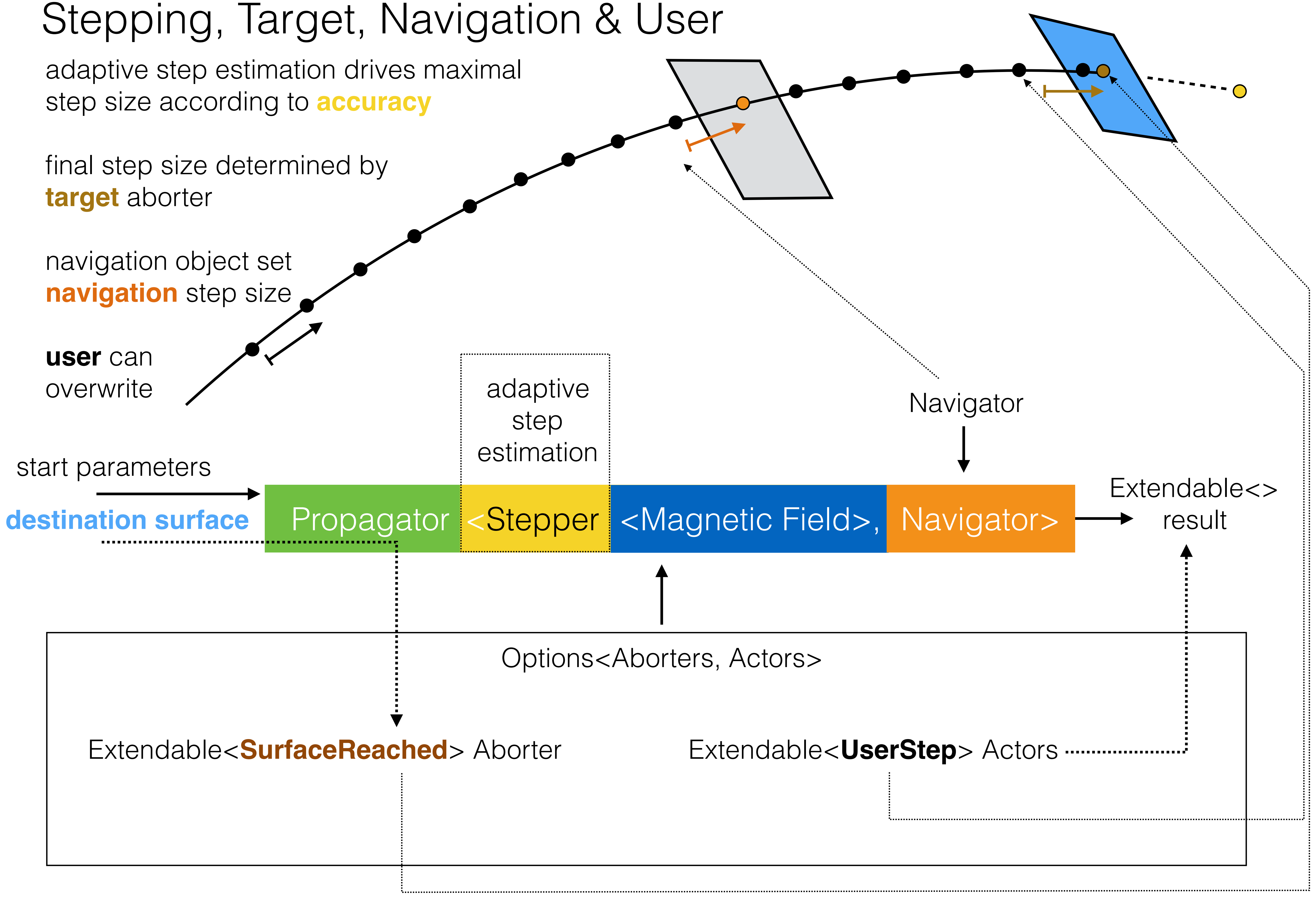
Stepping, Target, Navigation & User

adaptive step estimation drives maximal step size according to **accuracy**

final step size determined by **target** aborter

navigation object set **navigation** step size

user can overwrite



adaptive step estimation

Navigator

start parameters

destination surface

Propagator <Stepper <Magnetic Field>, Navigator>

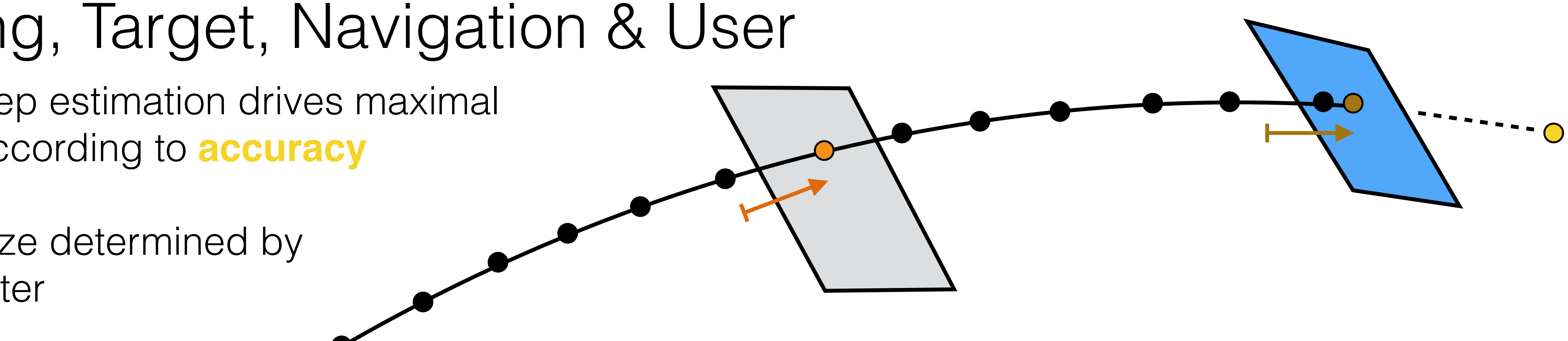
Extendable<> result

Options<Aborters, Actors>

Extendable<**SurfaceReached**> Aborter

Extendable<**UserStep**> Actors

Stepping, Target, Navigation & User



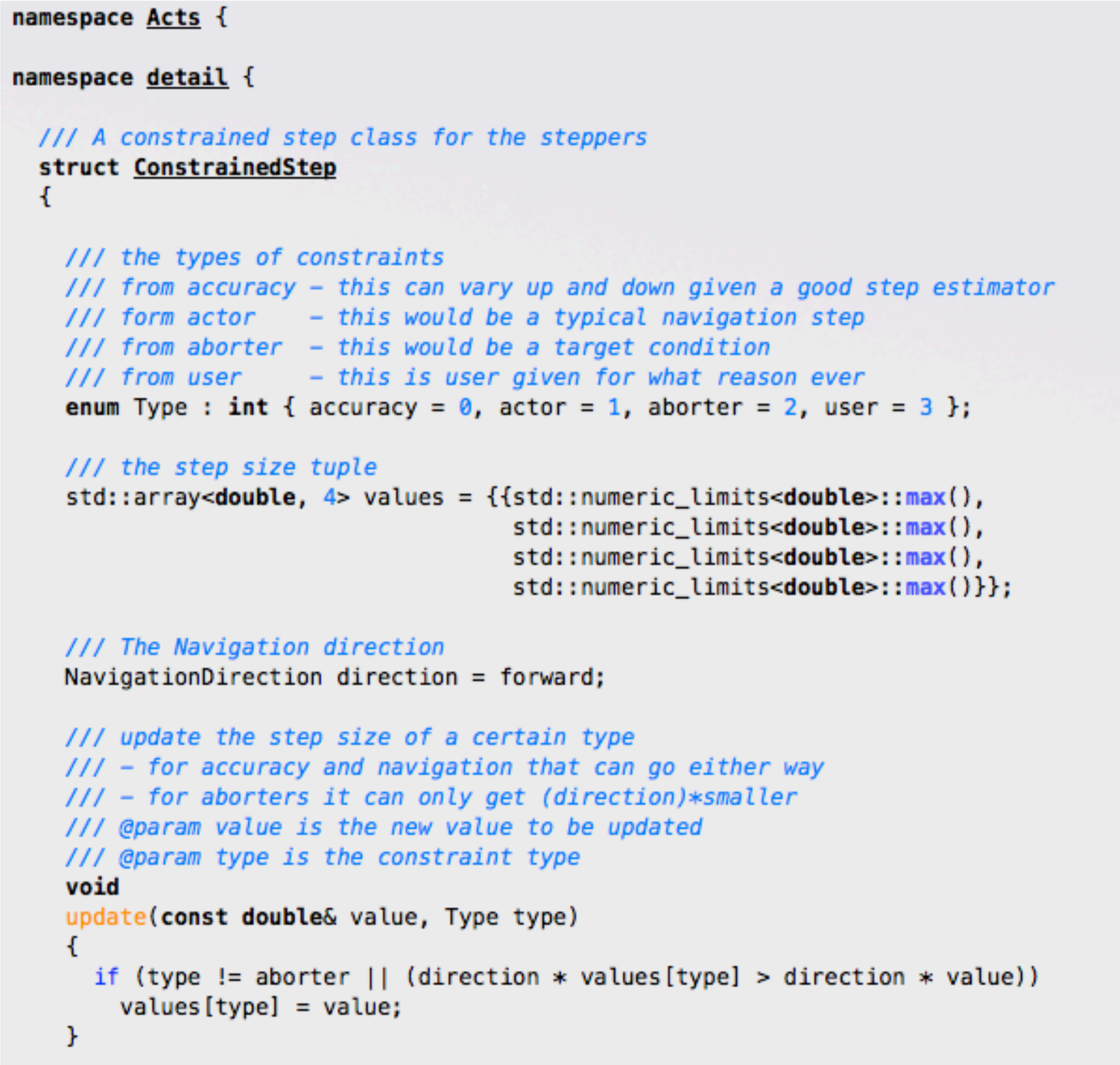
adaptive step estimation drives maximal step size according to **accuracy**

final step size determined by **target** aborter

navigation object set **navigation** step size

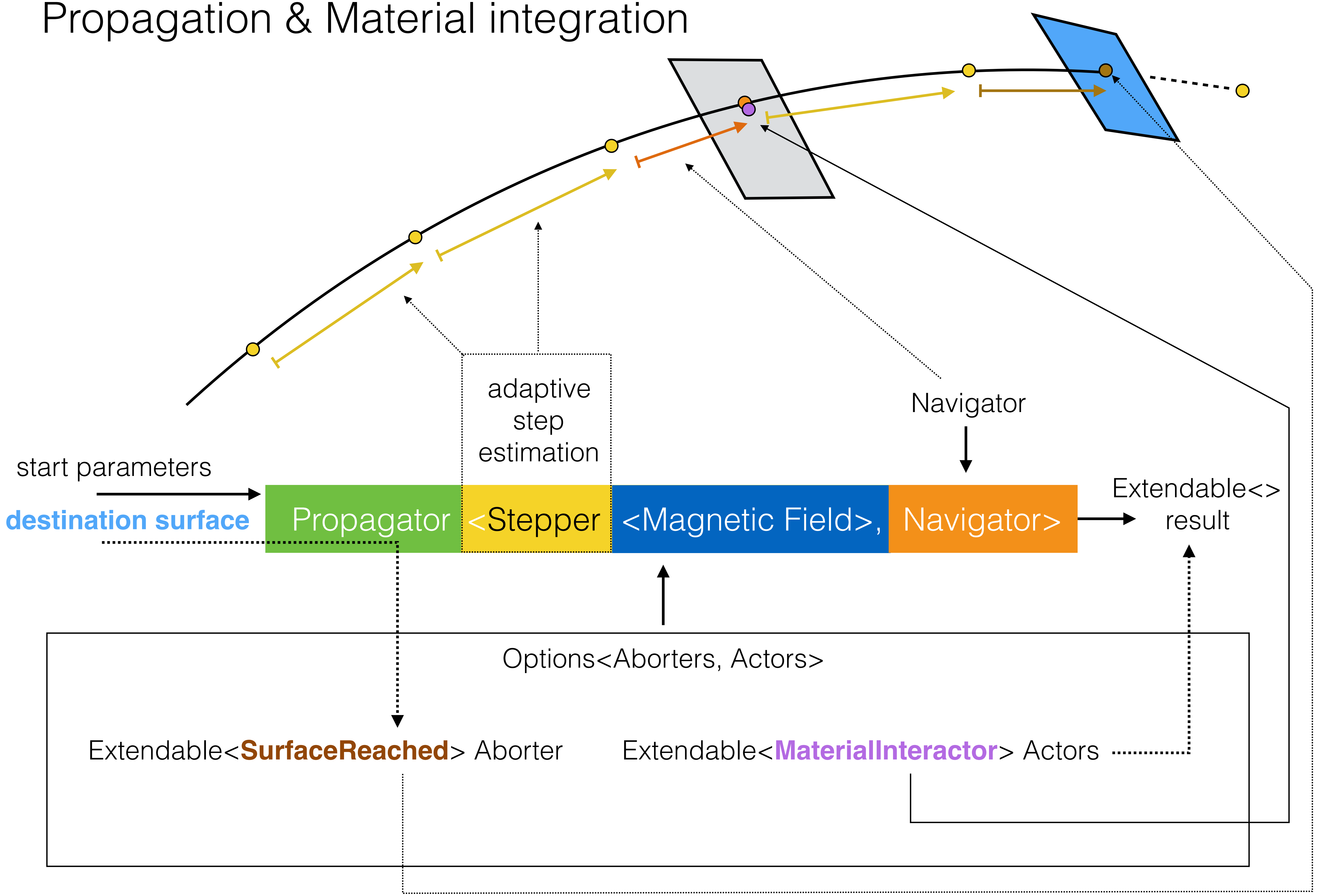
user can overwrite

ConstrainedStep



```
namespace Acts {

namespace detail {

  /// A constrained step class for the steppers
  struct ConstrainedStep
  {

    /// the types of constraints
    /// from accuracy - this can vary up and down given a good step estimator
    /// form actor    - this would be a typical navigation step
    /// from aborter  - this would be a target condition
    /// from user     - this is user given for what reason ever
    enum Type : int { accuracy = 0, actor = 1, aborter = 2, user = 3 };

    /// the step size tuple
    std::array<double, 4> values = {{std::numeric_limits<double>::max(),
                                     std::numeric_limits<double>::max(),
                                     std::numeric_limits<double>::max(),
                                     std::numeric_limits<double>::max()}};

    /// The Navigation direction
    NavigationDirection direction = forward;

    /// update the step size of a certain type
    /// - for accuracy and navigation that can go either way
    /// - for aborters it can only get (direction)*smaller
    /// @param value is the new value to be updated
    /// @param type is the constraint type
    void
    update(const double& value, Type type)
    {
      if (type != aborter || (direction * values[type] > direction * value))
        values[type] = value;
    }
```

Propagation & Material integration



adaptive step estimation

Navigator

start parameters

destination surface

Propagator <Stepper <Magnetic Field>, Navigator>

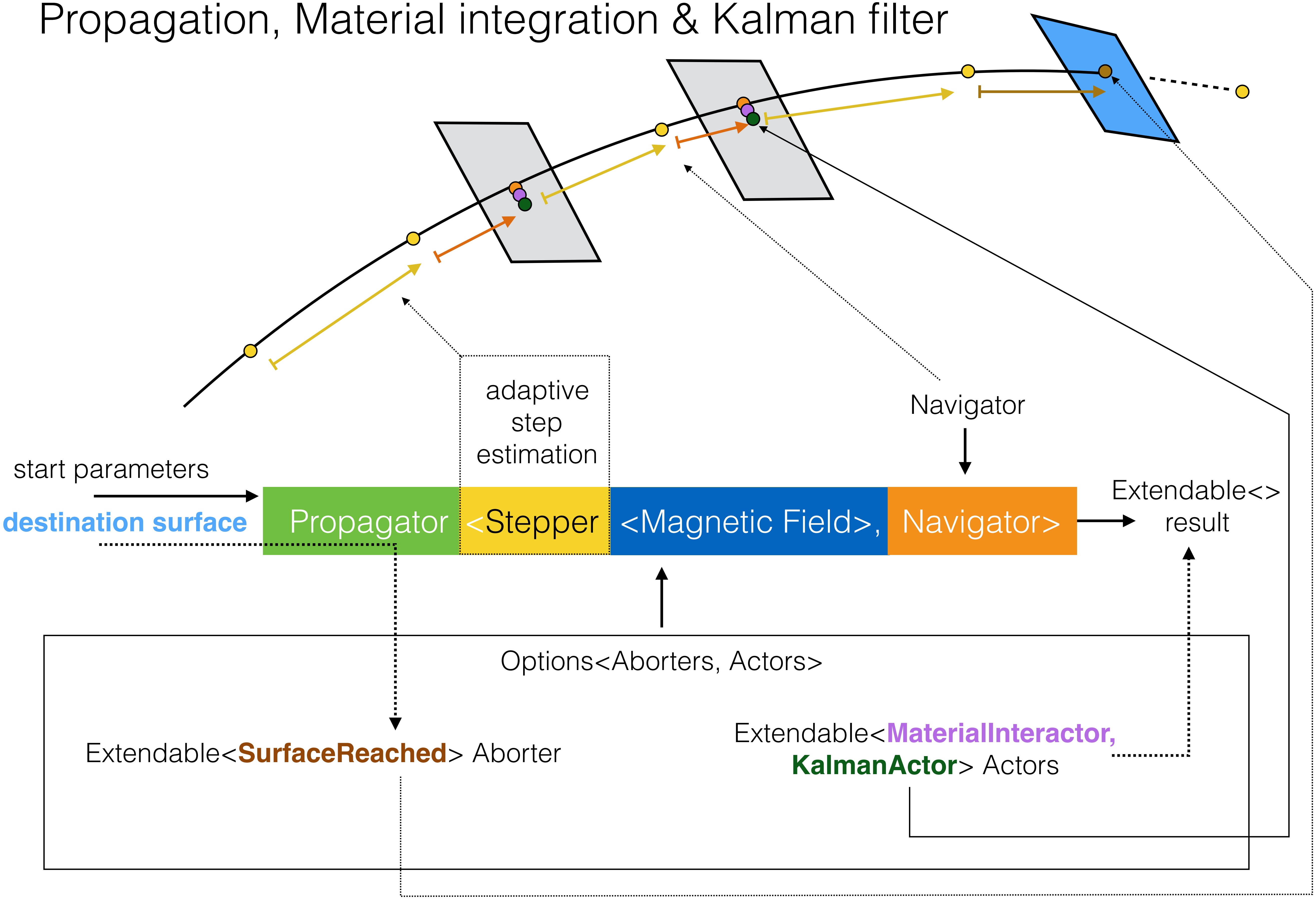
Extendable<> result

Options<Aborters, Actors>

Extendable<**SurfaceReached**> Aborter

Extendable<**MaterialInteractor**> Actors

Propagation, Material integration & Kalman filter



adaptive step estimation

Navigator

start parameters

destination surface

Propagator <Stepper <Magnetic Field>, Navigator>

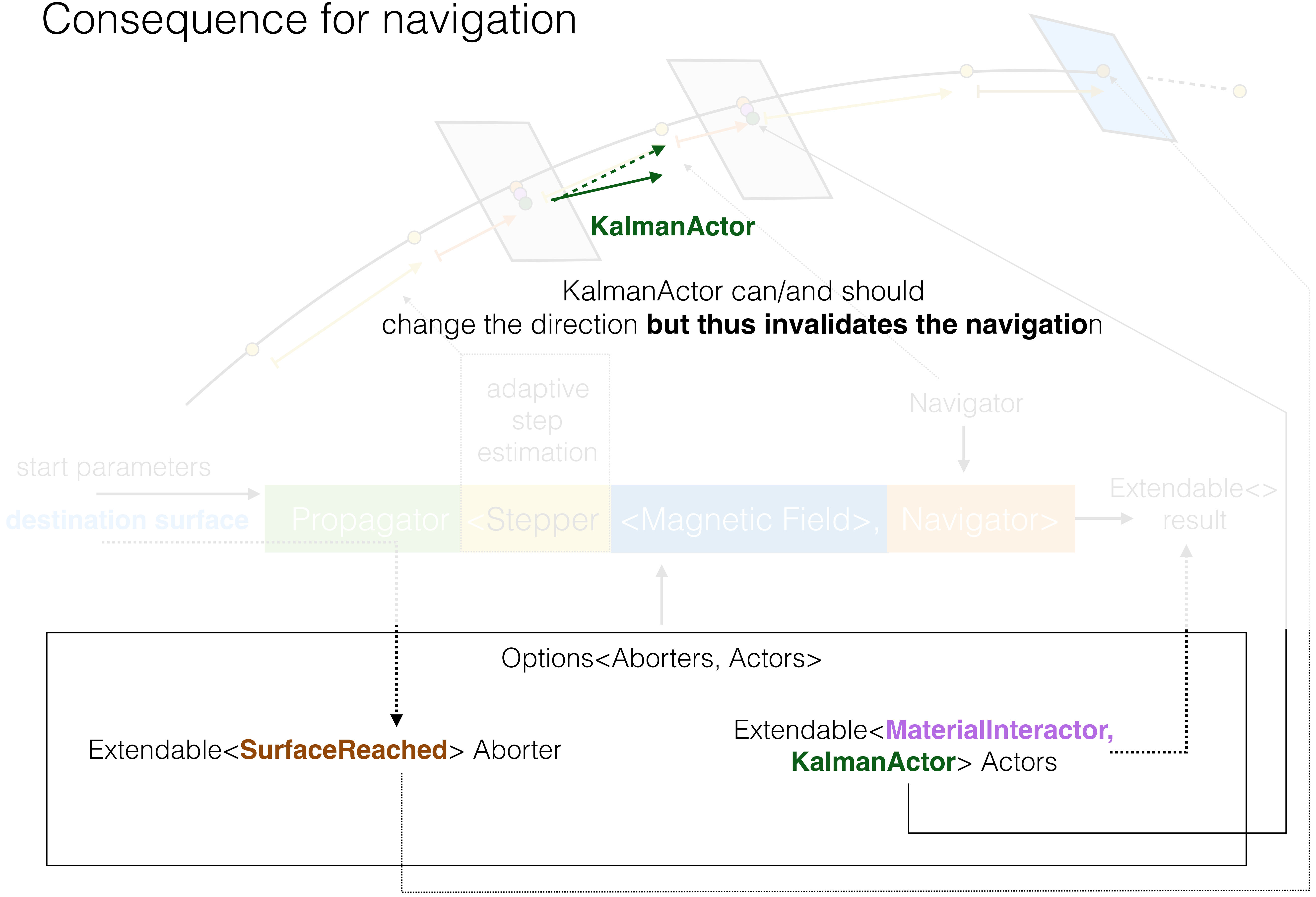
Extendable<> result

Options<Aborters, Actors>

Extendable<**SurfaceReached**> Aborter

Extendable<**MaterialInteractor,** **KalmanActor**> Actors

Consequence for navigation



KalmanActor

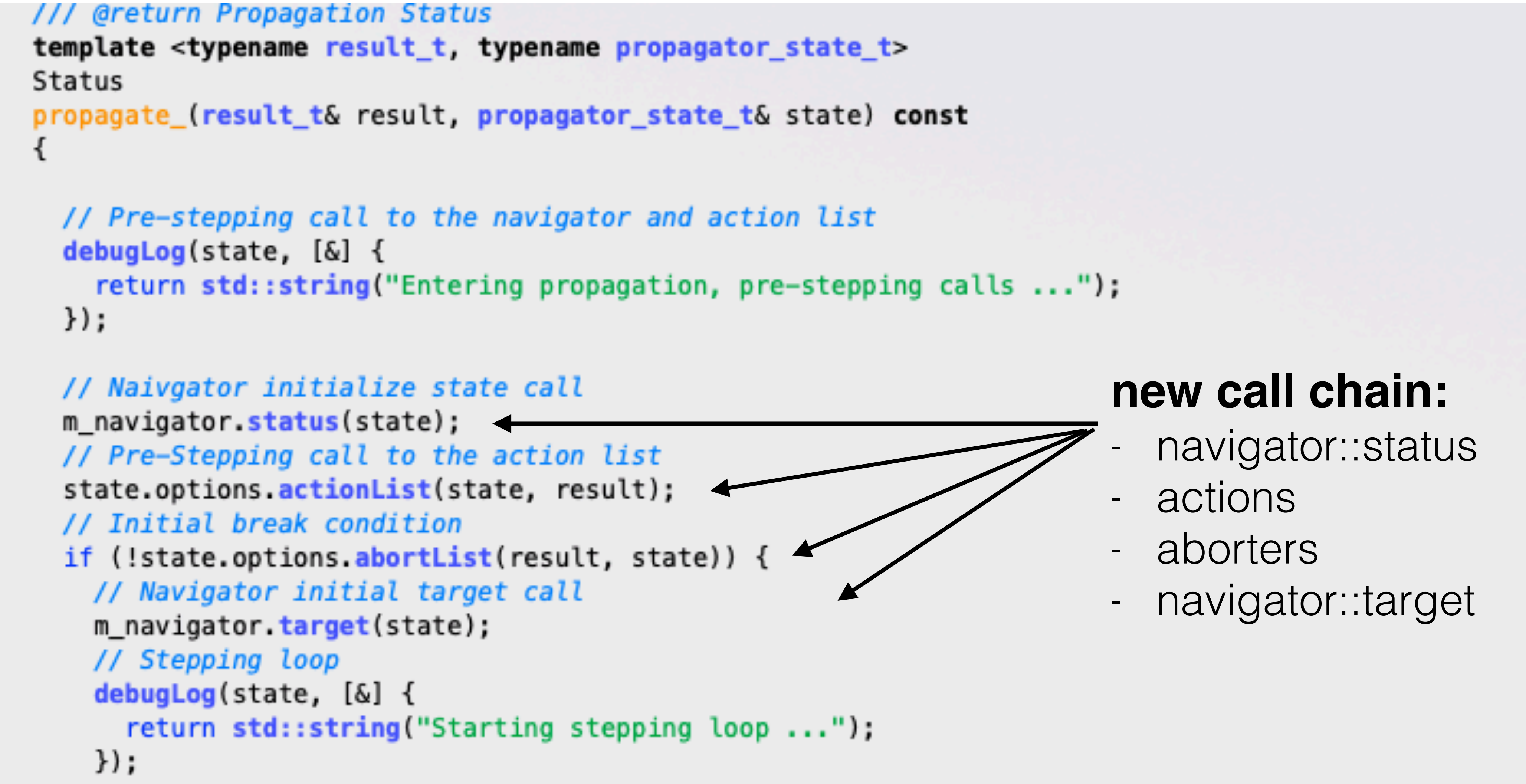
KalmanActor can/and should
change the direction **but thus invalidates the navigatio**n

Options<Aborters, Actors>

Extendable<**SurfaceReached**> Aborter

Extendable<**MaterialInteractor,**
KalmanActor> Actors

Consequence for navigation



```
/// @return Propagation Status
template <typename result_t, typename propagator_state_t>
Status
propagate_(result_t& result, propagator_state_t& state) const
{

  // Pre-stepping call to the navigator and action list
  debugLog(state, [&] {
    return std::string("Entering propagation, pre-stepping calls ...");
  });

  // Naivgator initialize state call
  m_navigator.status(state);
  // Pre-Stepping call to the action list
  state.options.actionList(state, result);
  // Initial break condition
  if (!state.options.abortList(result, state)) {
    // Navigator initial target call
    m_navigator.target(state);
    // Stepping loop
    debugLog(state, [&] {
      return std::string("Starting stepping loop ...");
    });
```

new call chain:

- navigator::status
- actions
- aborters
- navigator::target