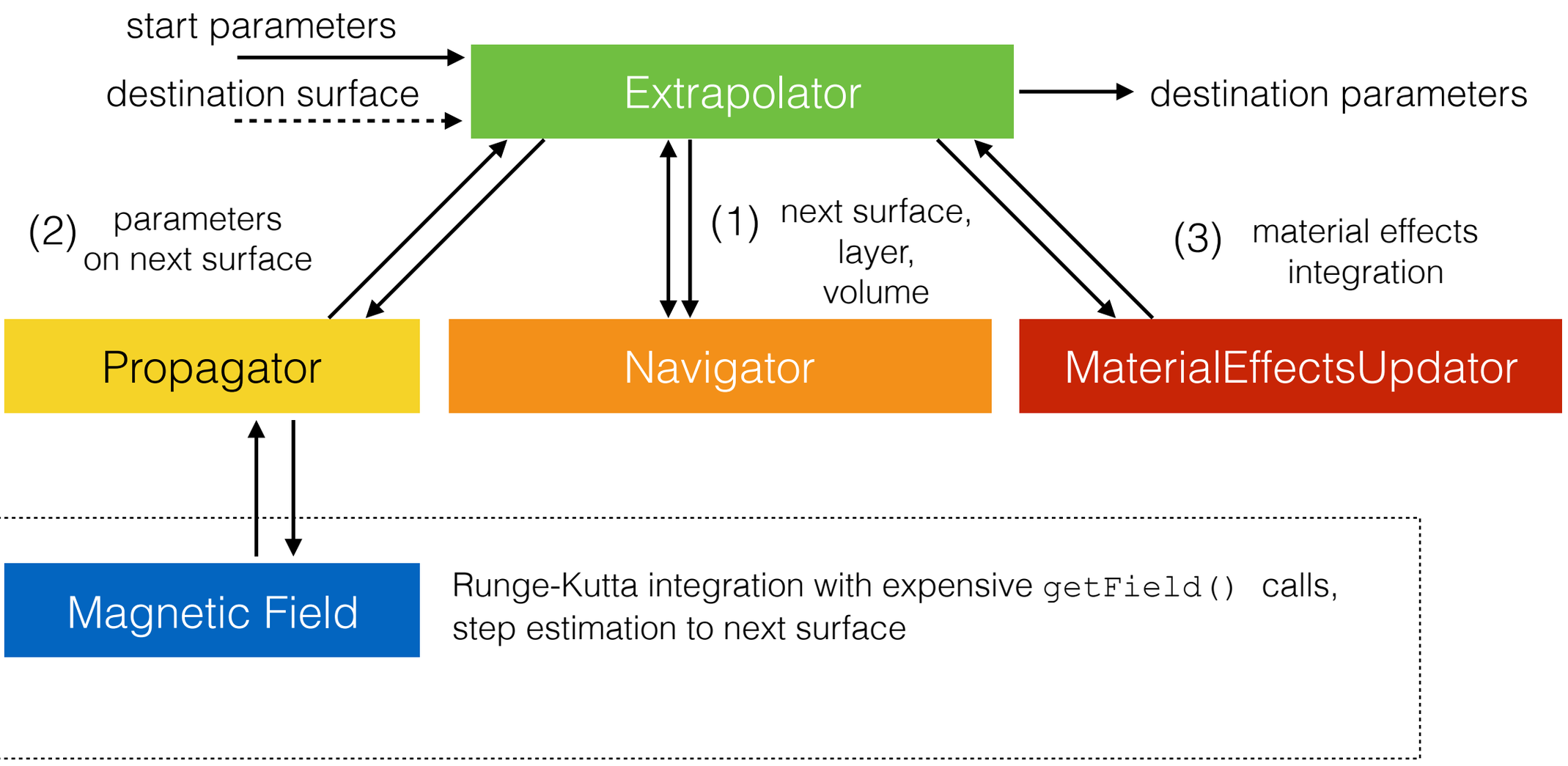
# Propagator - Update

A. Salzburger

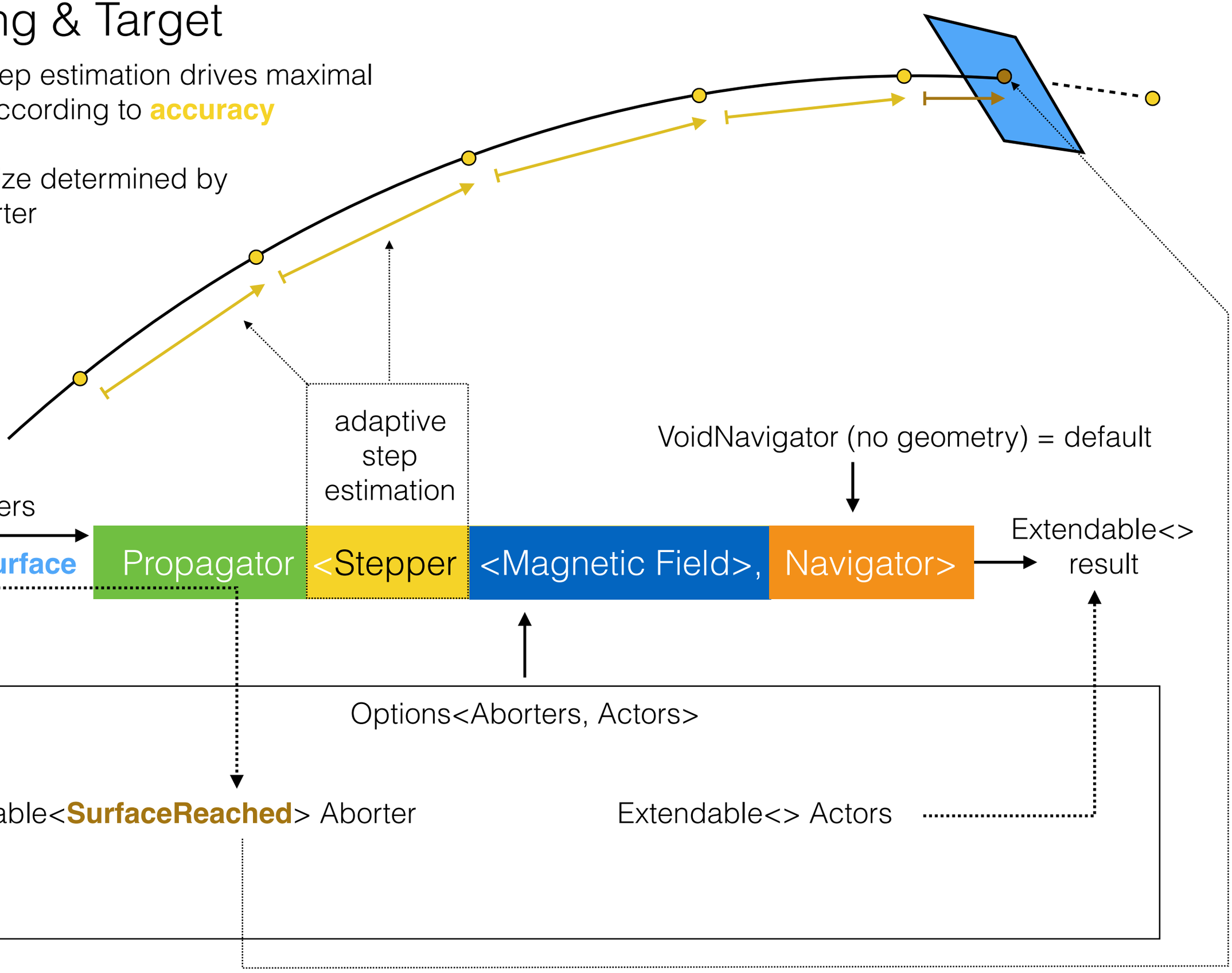
# Reminder: ATLAS Propagator/Extrapolator setup



# Stepping & Target

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

final step size determined by **target** aborter

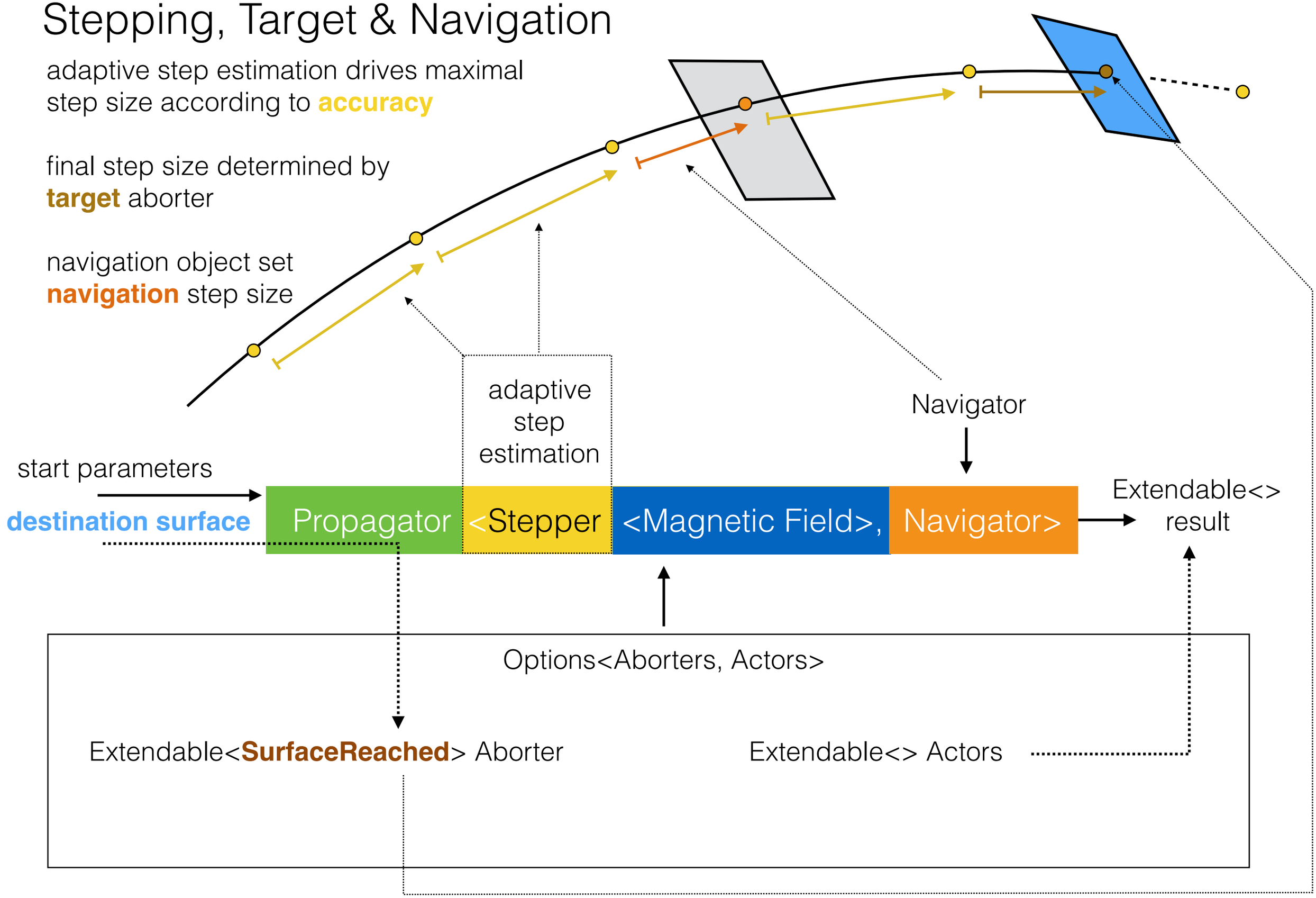


# Stepping, Target & Navigation

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

final step size determined by **target** aborter

navigation object set **navigation** step size



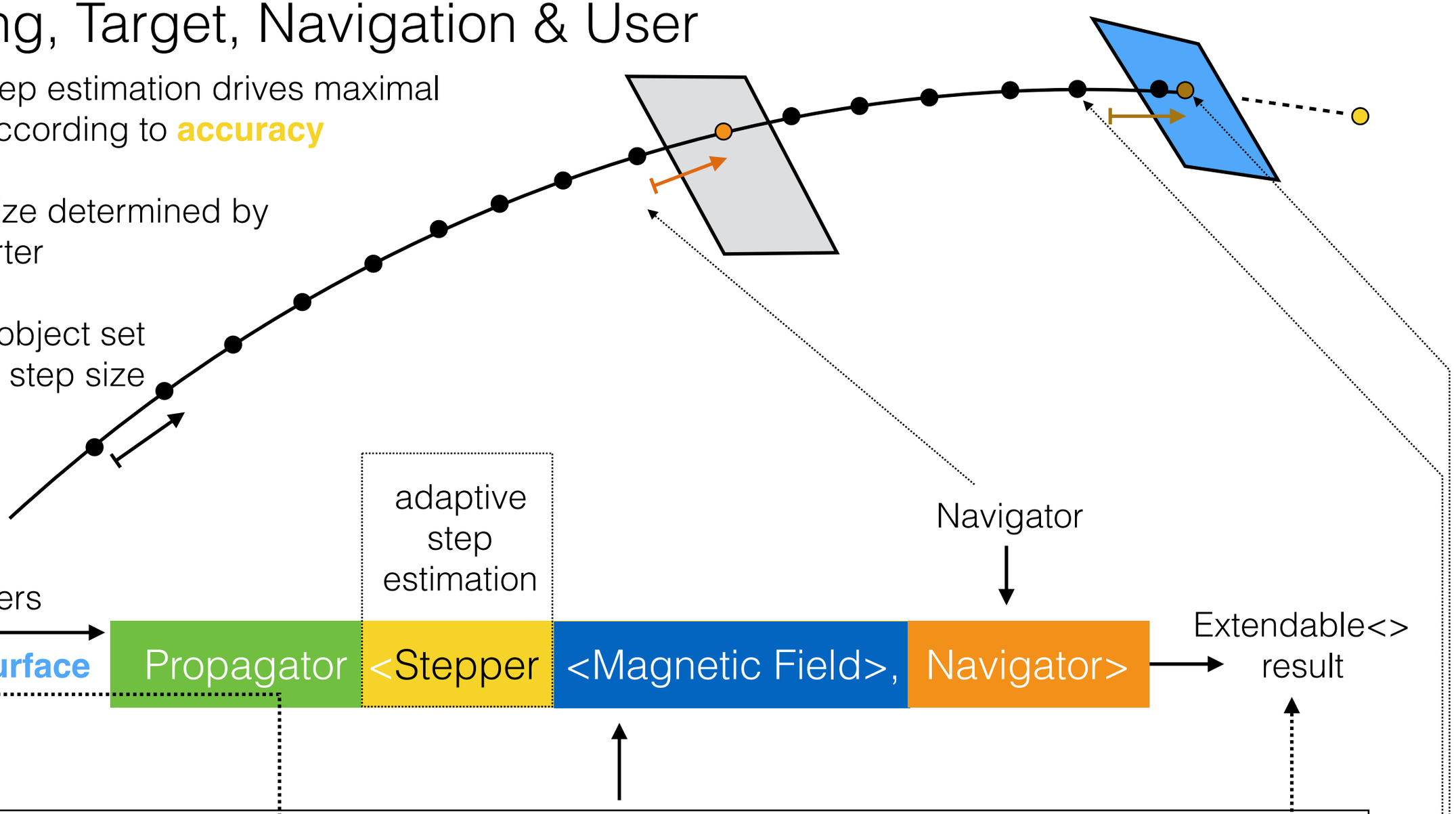
# Stepping, Target, Navigation & User

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

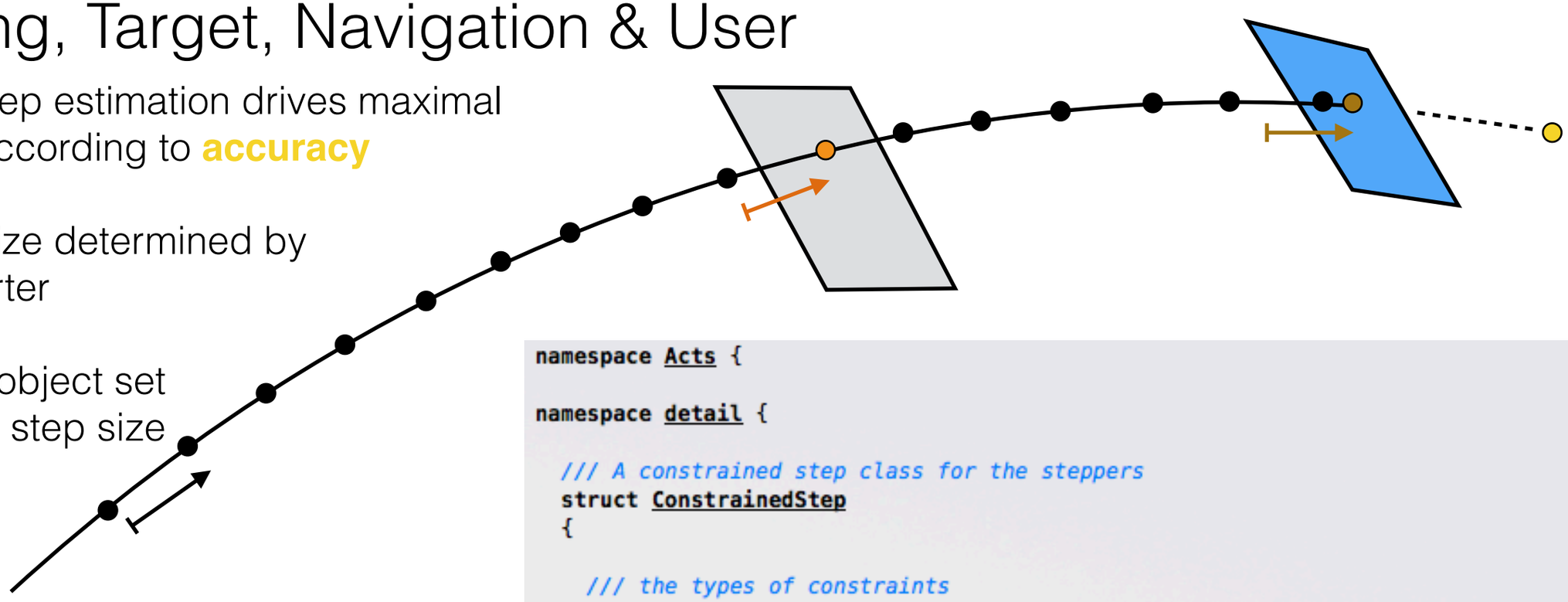
# Stepping, Target, Navigation & User

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

final step size determined by **target** aborter

navigation object set **navigation** step size

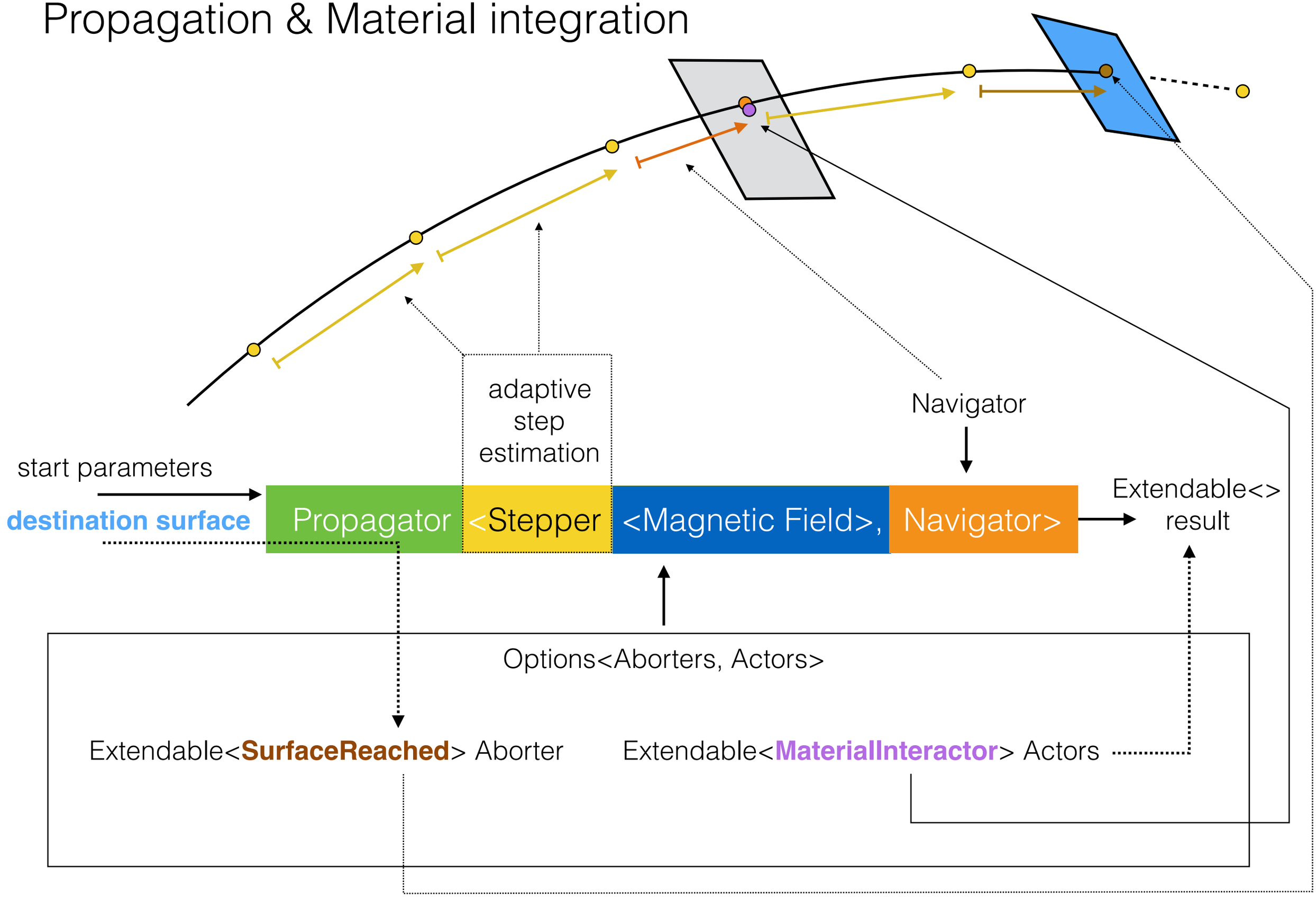
**user** can overwrite



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



# Propagation, Material integration & Kalman filter