1 Gecode classes

Trying to get an overview of the different classes in gecode and how they interact.

1.1 Space

Space contains all variables and constraints. The space is cloned (meaning objects in space are copied) each time a propagator is run. Space implements SharedHandle that again implements Object which is virtual. Object is used during copying of the space and SharedHandle handles shared objects when updating instead of copy when doing non-shared updates.

```
/*
* Space cloning
*
* Cloning is performed in two steps:
*  - The space itself is copied by the copy constructor. This
*    also copies all propagators, branchers, and variables.
*    The copied variables are recorded.
*  - In the second step the dependency information of the recorded
*    variables is updated and their forwarding information is reset.
*
*/
```

1.2 Script

ScriptBase is an abstract class that has a template constructor where it inherit from a baseSpace this is all located in driver.hh (sort of header for the folder driver).
Script.hpp has several methods that that all are part of search and why a search is stopped.

1.3 Home

Takes a space and a propagator as argument and post propagator in that space. Can return the space and the propagator. Can check if the space is failed (nor more branches to explore but not solved). Can notice actor property (think propagators are actors).

1.4 Propagators

Takes a Home as argument for constructing and can be cloned. Following taken from description in code

```
\brief Propagation function
     *
     * The propagation function must return an execution status as
     * follows:
     *  - ES_FAILED: the propagator has detected failure
```

```
*  - ES_NOFIX: the propagator has done propagation
*  - ES_FIX: the propagator has done propagation and has computed
*    a fixpoint. That is, running the propagator immediately
*    again will do nothing.
*
* Apart from the above values, a propagator can return
* the result from calling one of the functions defined by a space:
*  - ES_SUBSUMED: the propagator is subsumed and has been already
*    deleted.
*  - ES_NOFIX_PARTIAL: the propagator has consumed some of its
*    propagation events.
*  - ES_FIX_PARTIAL: the propagator has consumed some of its
*    propagation events and with respect to these events is
*    at fixpoint
```

Propagators can uses advisors (not sure what they do). They should be specified for each propagator.

1.5 Branchers

Branchers are created with Home as argument. BranchHandlers are used to handle branching during updates.

```
stable:
   /*
    * Find the next brancher that has still alternatives left
    *
    * It is important to note that branchers reporting to have no more
    * alternatives left cannot be deleted. They cannot be deleted
    * as there might be choices to be used in commit
    * that refer to one of these branchers. This e.g. happens when
    * we combine branch-and-bound search with adaptive recomputation: during
    * recomputation, a copy is constrained to be better than the currently
    * best solution, then the first half of the choices are posted,
    * and a fixpoint computed (for storing in the middle of the path). Then
    * the remaining choices are posted, and because of the additional
    * constraints that the space must be better than the previous solution,
    * the corresponding Branchers may already have no alternatives left.
    *
    * The same situation may arise due to weakly monotonic propagators.
    *
    * A brancher reporting that no more alternatives exist is exhausted.
    * All exhausted branchers will be left of the current pointer b_status.
    * Only when it is known that no more choices
    * can be used for commit an exhausted brancher can actually be deleted.
    * This becomes known when choice is called.

    while (b_status != Brancher::cast(&bl))
     if (b_status->status(*this)) {
       // Brancher still has choices to generate
       s = SS_BRANCH; goto exit;
```

```
    } else {
      // Brancher is exhausted
      b_status = Brancher::cast(b_status->next());
    }
  // No brancher with alternatives left, space is solved
  s = SS_SOLVED;
```

1.6 Options

BaseOption is defined in Driver.hh as an abstract class several classes inherit from it such as stringvalueioption, intoption, unsignedintoption. Options inherit from BaseOption aswell but those options are mere search related, stop restart, branch, log.