EECS4312-F18 Project: Explosive Store Requirements Document

Michael Founk (michaelfounk@outlook.com)
Sean O'Brien (sean1025@my.yorku.ca)

April 20, 2021

Statistics: Submit only under one team member

- Under which account submitted Prism: mfounk
- Under which account submitted Moodle: Sean O'Brien
- Implementation Language: NodeJS
- Estimate hours on requirements document: 10 hours
- Estimate of hours on coding and acceptance testing: 20 hours
- Hours on debugging and testing: 10 hours out of 20 hours.
- What was the hardest part of the project?: Developing the the TLA+ specification

Revisions

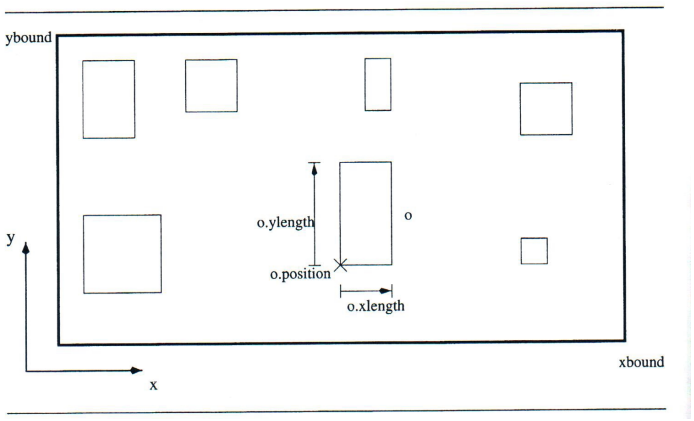
Date	Revision	Description
28 November 2018	1.0	Initial release of this document
2 December 2018	2.0	Added TLA+ Spec and its related components
2 December 2018	2.1	Updated components and added tables for error

Requirements Document:
Explosive System Store

Contents

1. Informal description of explosive system store system

The system under description is part of the controller for a robot which positions explosives such as dynamite and detonators in a storage area — according to UN safety regulations for a Small Explosive Store. Positions in the storage building are represented as coordinates with respect to the origin (bottom left corner). The store's dimension are represented as maximum x and y coordinates.



Explosive objects (such as dynamite and detonators) are in rectangular packages, aligned with the walls of the store (as in the figure). Each explosive object has dimensions in the x and y dimensions as shown. The position of an explosive object is represented as the coordinate of its lower left corner. All objects must fit within the store and there must be no overlap between objects so as to prevent unsafe explosions. The positioning controller must provide functions to suggest a position where an explosive object may be accommodated, update the store data model when adding an explosive object and update the store data model when an explosive object is removed.

It is premature at this point to specify a concrete user interface. However, our customer requires that we describe an abstract user interface that specifies the operations and resulting output that any concrete user interface must satisfy. This abstract description may also be used to convert Use Cases into formal Acceptance Tests.

Our customer requires the ability to initialize a new store of a given size. The largest small store is 10m x 10m. Packages (containing explosives) come in decimeter sizes (1 decimeter is one tenth of a meter). Regulations require that explosives in a package are wrapped with special high density protective pellets. Thus, packages may touch each other, but not overlap with each other. The pellet wrappings within packages ensures that explosives are sufficiently distanced from each other for storage safety.

Our customer requires operations to put new packages of explosives (via an identifier) in the store, and to later remove such packages. Also, the system must provide the ability to suggest positions in the store where packages may be placed, and to list explosives of a given size or larger.

2. High level goals

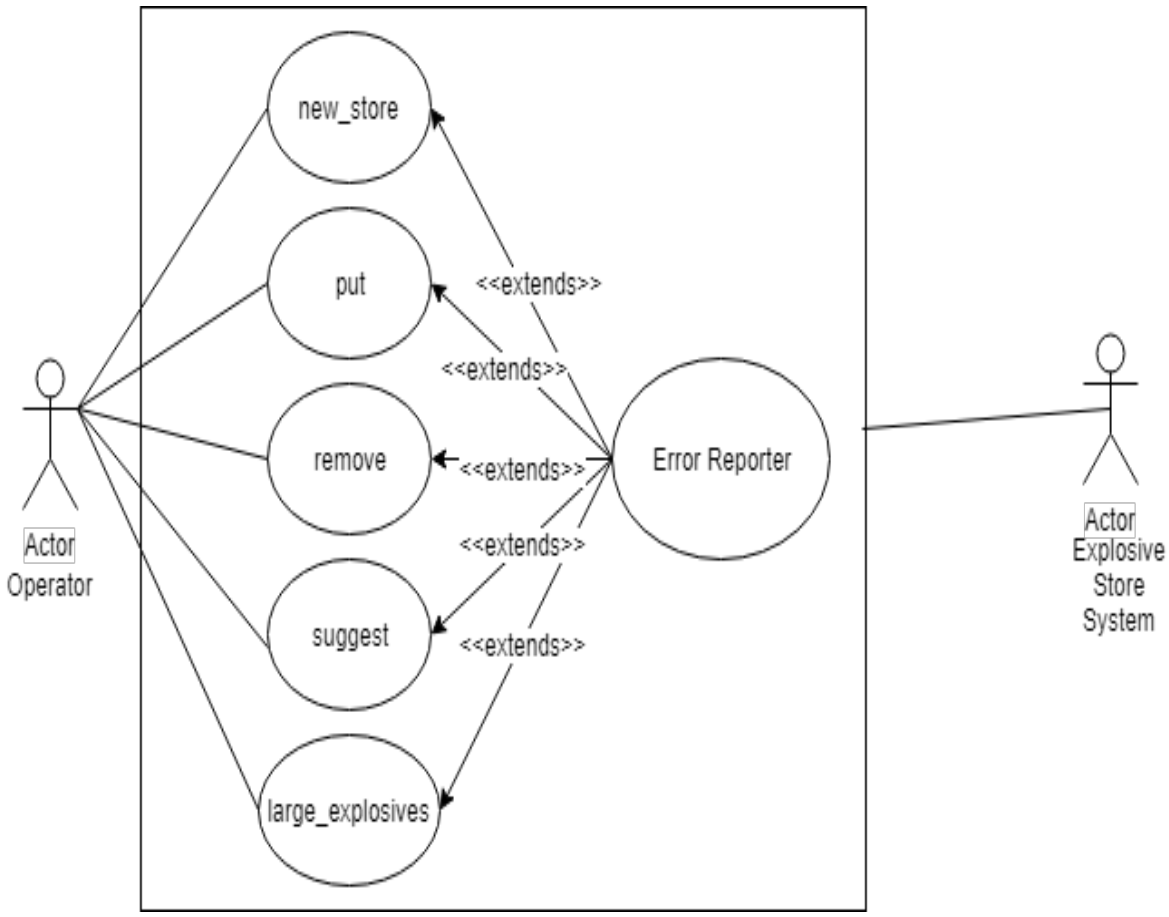
G1: Create that allows creating, putting and removing explosives in a store.

G2: Ensure that the system preserves the fact that no explosives should overlap with one another.

G3: The operator will be warned if anything they do will be not valid or break the safety invariant mentioned above.

G4: The system is handling explosives and as such should have no errors.

3. Use Case Diagram and Textual Use Cases



Use Case	Put Explosive
Actor	System
Pre-Condition	The store is created
Post-Condition	The explosive is put into the store
Main Success Scenario	
Actor	**System Response**
1. Operator chooses to put a new explosive	1. Prompts to input the id, length/width of explosive and position of the explosive
2. Inputs all the required information	2. System puts the specified explosive into the store
Deviating Scenarios:	
Deviate at 1	Actor decides not to put the explosive in. Use case terminates
Deviate at 2.1	ID already exists. System returns to 1.
Deviate at 2.2	ID longer than 2. System returns to 1.
Deviate at 2.3	ID doesn't have first character as a letter. System returns to 1.
Deviate at 2.4	ID doesn't have first character as a number. System returns to 1.
Deviate at 2.5	Explosive size isn't at least [1,1]. System returns to 1.
Deviate at 2.6	Explosive will overlap with another explosive. System returns to 1.
Deviate at 2.7	Explosive will be out of bounds of the store. System returns to 1.

Use Case	Creation of a new store
Actor	System
Pre-Condition	There is not existent store
Post-Condition	A store is created and explosives can be put in
Main Success Scenario	
Actor	**System Response**
1. Operator chooses to create a new store	1. System prompts to input length and width of the store to be created
2. Inputs all the required information	2. System creates a store of the specified dimensions
Deviating Scenarios:	
Deviate at 1	Actor decides not to create a store. Use case terminates
Deviate at 2.1	Length or width is equal to zero. System return to 1.

Use Case	Remove Explosive from store
Actor	System
Pre-Condition	There is a store with at least one explosive in it
Post-Condition	The store now has one less explosive than it did before
Main Success Scenario	
Actor	**System Response**
1. Operator chooses to remove an explosive	1. System prompts for the ID of the explosive to remove
2. Operator inputs all the required information	2. System removes the specified explosive from the store
Deviating Scenarios:	
Deviate at 1	Actor decides not to remove the explosive. Use case terminates
Deviate at 2.1	ID does not exist in the store. System returns to 1.

4. E-Descriptions

ENV1	Explosive objects are in rectangular packages, aligned with the walls of the store	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test.

ENV2	When explosives overlap, there is a risk of explosion.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test.

ENV3	Explosives are wrapped in protective pellets, allowing them to touch one another.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test.

5. R-Descriptions

REQ4	The system shall prevent overlapping of explosives.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ5	The system shall allow users to initialize a new store, if one has not yet been created or if there are no explosives in the current store	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ6	The system shall allow users to place explosives in the store, provided that no part of the explosive exists outside the bounds of the store and no explosives overlap.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

6. Abstract user interface grammar

After several rounds of requirements elicitation, the following user inputs are defined.

```
system explosive
  -- small explosive store according to UN regulations

type ID = STRING -- explosive identifiers

-- largest 10m x 10m, spacing is 1 decimeter i.e. one tenth of a
   meter
type UNIT = 0 .. 100

type POSITION = TUPLE[x:UNIT; y:UNIT]
  -- x and y non-negative

type EXPLOSIVE = TUPLE[xlength:UNIT; ylength: UNIT]
  -- a_xlength and a_ylength greater than zero

-- commands

new_store(xbound:UNIT; ybound:UNIT)
  -- create a new store with the given bounds
  -- [1,1] provides a store with one position [0,0]

put (id:ID; explosive: EXPLOSIVE; position: POSITION)
  -- place an explosive in the store

remove (id:ID)
  -- remove an explosive

-- queries

suggest(explosive: EXPLOSIVE)
  -- suggest a point in the store to place an explosive of the
     given dimension

large_explosives (xlength: UNIT; ylength: UNIT)
  -- list explosives with these dimensions or larger
```

7. Abstract State

Abstract Variable	Description	Initial value
$taken\_id \subseteq ID$	Set containing the ids currently in the store	{}
$explosive \in [taken\_id \rightarrow EXPLOSIVE]$	Function matching the set of ids currently in the store to a tuple containing their position and dimension	<<>>
$xbound \in Int$	Integer representing the x-dimension of the store	0
$ybound \in Int$	Integer representing the y-dimension of the store	0
$spaces \subseteq POSITION$	A set of spaces suitable for the storing an explosive of a given size	{}
$large \subseteq taken\_id$	A set of ids with dimensions greater than or equal to the given dimensions	{}

8. ASCII encoded output of the abstract state

The delivered program shall execute from a CentOS7 Terminal as follows:

```
>./explosive.exe -b tests/at0.txt
```

A sequence of user commands, e.g. at0.txt below, shall conform to the abstract UI grammar.

```
-- A basic example
new_store(6,6)
put("A1",[1,1],[0,0])
put("B1",[2,1],[1,0])
put("C1",[3,1],[3,0])
put("A2",[1,2],[0,1])
put("A3",[1,3],[0,3])
put("A3",[1,3],[0,3])  -- error ID already exists
put("a9",[1,3],[0,3])  -- error explosive will overlap
suggest([1,1])
suggest([2,2])
put("z0", [1,1],[3,4])
suggest([2,2])
put("B2",[2,2],[1,1])
put("B3",[2,3],[1,3])
put("C2",[3,2],[3,1])
put("C3",[3,3],[3,3])
remove("B2")
remove("A1")
remove("C3")
suggest([2,2])
suggest([1,1])
suggest([3,3])
```

Any acceptance test (authored by the Customer) such as `at0.txt` above, will not have syntax errors (i.e. will satisfy the abstract UI grammar).

The final software product **explosive.exe** (a command line program) shall produce the expected output (such as `at0.expected.txt`, see below) for any acceptance test supplied by the Customer.

The final software product shall produce (character-for-character) the output that the Customer expects. Even a one character difference will be deemed not to have satisfied the requirements. The following are unacceptable: (1) Semantic Errors (e.g. improper identifiers). (2) Functional Problems: user inputs that succeed but do the wrong thing or violate system safety invariants (TLA+ should be helpful in this regard). (3) The program crashes with or without exceptions for legal input. (4) Non-Termination.

From prior requirements elicitation (Phase1), the Customer has provided the following specification files in the course directory (`/eecs/course/4312/project`):

```
project
|-- at0.expected.txt
|-- at0.txt
|-- explosive-store-defns.txt
```

Part of `at0.expected.txt` (the expected output) is shown below. Please consult the course directory for the precise details (as there may be some further changes during elicitation).

```
...
->put("A3",[1, 3],[0, 3])
  System State: (6)
  A3 __ __ __ __ __
  A3 __ __ __ __ __
  A3 __ __ __ __ __
  A2 __ __ __ __ __
  A2 __ __ __ __ __
  A1 B1 B1 C1 C1 C1
->put("A3",[1, 3],[0, 3])
  System State: (7) e03: ID already exists
->put("a9",[1, 3],[0, 3])
  System State: (8) e09: Explosive will overlap with another explosive
->suggest([1, 1])
  System State: (9)
  _____[1,5][2,5][3,5][4,5][5,5]
  _____[1,4][2,4][3,4][4,4][5,4]
  _____[1,3][2,3][3,3][4,3][5,3]
  _____[1,2][2,2][3,2][4,2][5,2]
  _____[1,1][2,1][3,1][4,1][5,1]
  ___________________________________
->suggest([2, 2])
  System State: (10)
  ___________________________________
  _____[1,4][2,4][3,4][4,4]_____
  _____[1,3][2,3][3,3][4,3]_____
  _____[1,2][2,2][3,2][4,2]_____
  _____[1,1][2,1][3,1][4,1]_____
  ___________________________________

...
```

9. TLA+ specfication, invariants and validation

```
---------------------------- MODULE explosive ----------------------------
EXTENDS Integers, Sequences, TLC

CONSTANTS ID, N
ASSUME N > 0

VARIABLES
    taken_id,
    explosives,
    xbound,
    ybound,
    spaces,
    larger

vars == <<taken_id, explosives, xbound, ybound, spaces, larger>>

UNIT == 0 .. N

UNIT_RECORD == [x : UNIT, y : UNIT]

POSITION == {r \in UNIT_RECORD : r.x < N /\ r.y < N}

DIMENSION == {r \in UNIT_RECORD : r.x > 0 /\ r.y > 0}

EXPLOSIVE == [p: POSITION, d: DIMENSION]

TypeOk ==
    /\ taken_id \subseteq ID
    /\ explosives \in [taken_id -> EXPLOSIVE]
    /\ xbound \in Int
    /\ ybound \in Int
    /\ spaces \subseteq POSITION
    /\ larger \subseteq taken_id


Init ==
    /\ taken_id = {}
    /\ explosives = << >>
    /\ xbound = 0
    /\ ybound = 0
    /\ spaces = {}
    /\ larger = {}

IdOk == taken_id = DOMAIN explosives
```

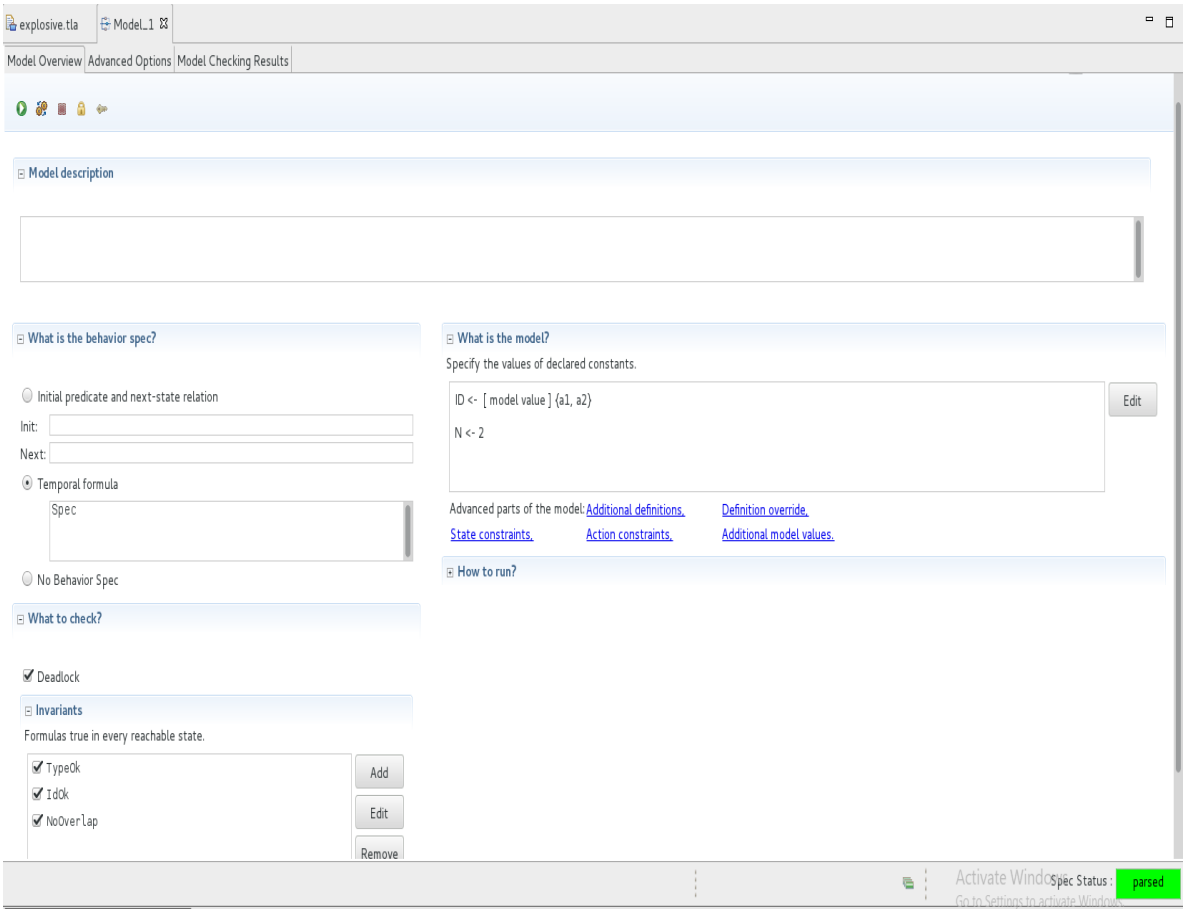
```
NoOverlap ==
  /\ \A i \in taken_id: \A j \in taken_id \ {i}:
     \/ explosives[i].p.x >= explosives[j].p.x + explosives[j].d.x
     \/ explosives[i].p.y >= explosives[j].p.y + explosives[j].d.y
     \/ explosives[j].p.x >= explosives[i].p.x + explosives[i].d.x
     \/ explosives[j].p.y >= explosives[i].p.y + explosives[i].d.y

NoOutside ==
    \A id \in taken_id :
       /\ explosives[id].p.x + explosives[id].d.x <= xbound
       /\ explosives[id].p.y + explosives[id].d.y <= ybound
       /\ explosives[id].p.x >= 0
       /\ explosives[id].p.y >= 0

new_store(xdim, ydim) ==
    /\ taken_id = {}
    /\ N >= xdim
    /\ xdim > 0
    /\ N >= ydim
    /\ xdim > 0
    /\ xbound' = xdim
    /\ ybound' = ydim
    /\ UNCHANGED explosives
    /\ UNCHANGED taken_id
    /\ UNCHANGED spaces
    /\ UNCHANGED larger
    /\ Print(<<"new_store",xdim,ydim>>, TRUE)

put(id,dim,pos) ==
    /\ id \in ID
    /\ id \notin taken_id
    /\ pos.x >= 0
    /\ pos.y >= 0
    /\ dim.x > 0
    /\ dim.y > 0
    /\ (pos.x + dim.x) <= xbound
    /\ (pos.y + dim.y) <= ybound
    /\ \A ide \in taken_id:
        \/ pos.x >= explosives[ide].p.x + explosives[ide].d.x
        \/ pos.y >= explosives[ide].p.y + explosives[ide].d.y
        \/ pos.x + dim.x <= explosives[ide].p.x
        \/ pos.y + dim.y <= explosives[ide].p.y
    /\ taken_id' = taken_id \union {id}
    /\ explosives' = explosives @@ id:>[p |-> pos, d |-> dim]
    /\ UNCHANGED xbound
    /\ UNCHANGED ybound
```

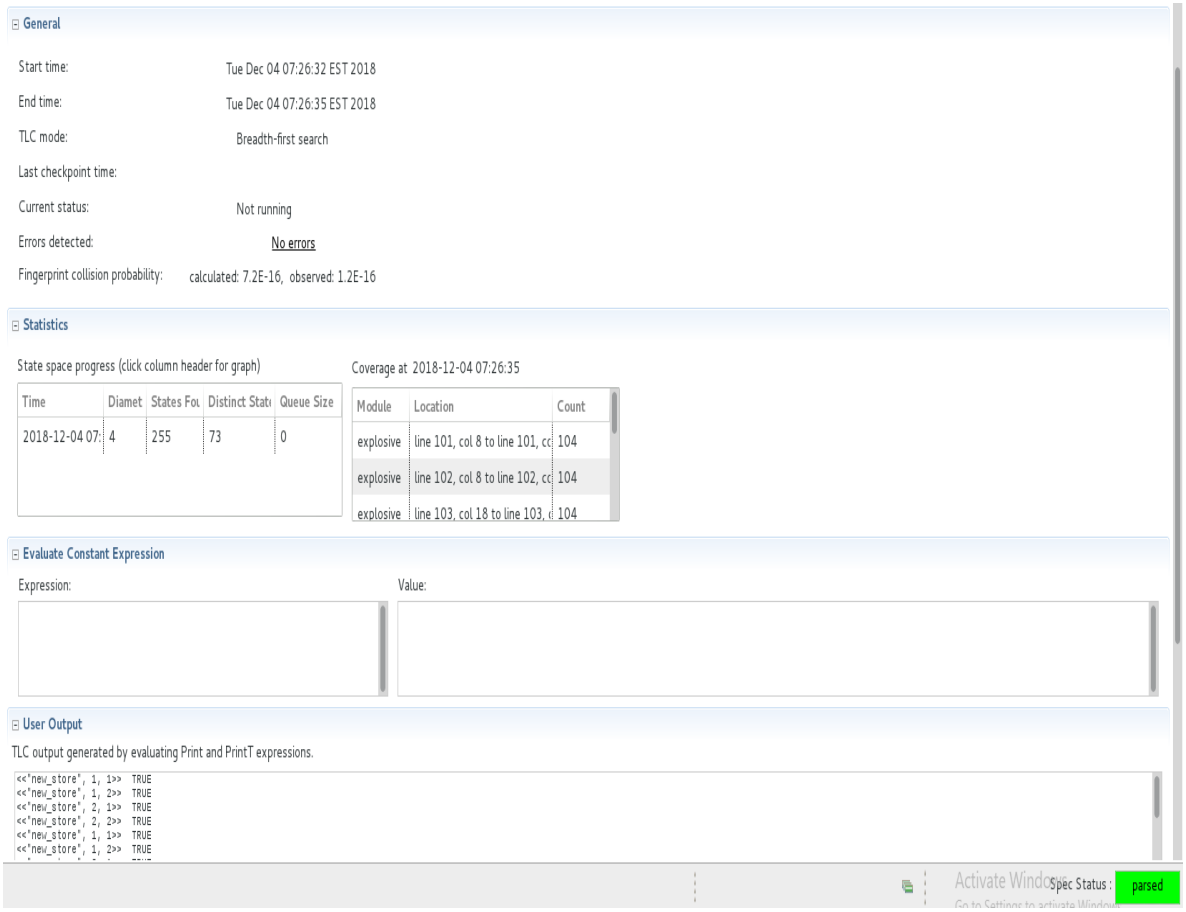
```
    /\ UNCHANGED spaces
    /\ UNCHANGED larger
    /\ Print(<<"put",id,dim,pos>>, TRUE)


remove(id) ==
    /\ xbound > 0
    /\ ybound > 0
    /\ id \in taken_id
    /\ explosives' = [x \in DOMAIN explosives \ {id} |-> explosives[x]]
    /\ taken_id' = taken_id \ {id}
    /\ UNCHANGED xbound
    /\ UNCHANGED ybound
    /\ UNCHANGED spaces
    /\ UNCHANGED larger
    /\ Print(<<"remove",id>>, TRUE)


suggest(s) ==
    /\ xbound > 0
    /\ ybound > 0
    /\ s \in DIMENSION
    /\ s.x =< xbound
    /\ s.y =< ybound
    /\ spaces' = {}
    /\ \A i \in 0 .. xbound, j \in 0 .. ybound, id \in taken_id:
         /\ \/ i >= explosives[id].p.x + explosives[id].d.x
            \/ j >= explosives[id].p.y + explosives[id].d.y
            \/ i + s.x <= explosives[id].p.x
            \/ j + s.y <= explosives[id].p.y
         /\ spaces' = spaces \union {<<i,j>>}
    /\ UNCHANGED xbound
    /\ UNCHANGED ybound
    /\ UNCHANGED explosives
    /\ UNCHANGED taken_id
    /\ UNCHANGED larger
    /\ Print(<<"suggest",s>>, TRUE)

large_explosives(xdim,ydim) ==
    /\ xbound >= xdim
    /\ xdim > 0
    /\ ybound >= ydim
    /\ xdim > 0
    /\ larger' = {}
    /\ \A i \in 0 .. xbound, j \in 0 .. ybound, id \in taken_id:
         /\ /\ i <= explosives[id].d.x
            /\ j <= explosives[id].d.y
```

```
          /\ larger' = larger \union {id}
    /\ UNCHANGED xbound
    /\ UNCHANGED ybound
    /\ UNCHANGED explosives
    /\ UNCHANGED taken_id
    /\ UNCHANGED spaces
    /\ Print(<<"large_explosives",xdim,ydim>>, TRUE)

Next ==
    \/ \E r \in DIMENSION: new_store(r.x,r.y)
    \/ \E id \in ID, d \in DIMENSION, p \in POSITION: put(id,d,p)
    \/ \E id \in ID: remove(id)
    \/ \E r \in DIMENSION: suggest(r)
    \/ \E r \in DIMENSION: large_explosives(r.x,r.y)

Spec ==
    /\ Init
    /\ [][Next]_vars

====
```

9.1. Validation

In our validation process we checked the types of the variables, as well as the system's knowledge of the contents of the store. We wrote two safety invariants, one to ensure that no explosives overlap, and another to ensure no explosive can be placed outside the bounds of the store.





10. Completeness, disjointness and well-definednes of the specification

$C1 =$
$taken\_id = \{\}$
$N >= xdim$
$xdim > 0$
$N >= ydim$
$xdim > 0$

$C2 =$
$id \in ID$
$id \notin taken_i d$

$xbound > 0$
$ybound > 0$
$pos.x >= 0$
$pos.y >= 0$
$dim.x > 0$
$dim.y > 0$
$(pos.x + dim.x) <= xbound$
$(pos.y + dim.y) <= ybound$

$C3 =$
$xbound > 0$
$ybound > 0$
$id \in taken\_id$

$C4 =$
$xbound > 0$
$ybound > 0$
$s \in DIMENSION$
$s.x =< xbound$
$s.y =< ybound$

$C5 =$
$xbound >= xdim$
$xdim > 0$
$ybound >= ydim$
$xdim > 0$

			taken_id	explosives	spaces	larger	xbound	ybound
i = 0			{}	<<>>	{}	{}	0	0
i >0	new_store(xdim, ydim)(i)	C1	{}	<<>>	{}	{}	xdim	ydim
		not C1	NC	NC	NC	NC	NC	NC
	put(id, dim, pos)(i)	C2	taken_id union *id*	explosives union$[id- > [p, d]]$	{}	{}	NC	NC
		not C2	NC	NC	NC	NC	NC	NC
	remove(id)(i)	C3	taken_id \id	explosive \e	{}	{}	NC	NC
		not C3	NC	NC	NC	NC	NC	NC
	suggest(s)(i)	C4	NC	NC	POSITION	NC	NC	NC
		not C4	NC	NC	NC	NC	NC	NC
	large_explosives(xdim, ydim)(i)	C5	NC	NC	NC	explosve[taken_id] > [xdim,ydim]	NC	NC
		not C5	NC	NC	NC	NC	NC	NC

11. Elicitation of error and status messages

Error Number	**Status message**
e01	e01: Store must have bounds of at least [1,1]
e02	e02: Store must exist
e03	e03: ID already exists
e04	e04: ID must be two characters in length
e05	e05: First character of ID must be a letter
e06	e06: Second character of ID must be a number
e07	e07: Explosive must be at least [1,1]
e08	e08: Explosive must be placed within store bounds
e09	e09: Explosive will overlap with another explosive
e10	e10: ID does not exist
e11	e11: Current store not empty

Function	**Priorities(Order)**
new_store	e11, e01
put	e02, e03, e04, e05, e06, e07, e08, e09
remove	e02, e10
suggest	e02, e07
large_explosives	e02, e07

12. Acceptance Tests

Use Cases	Description	Acceptance Tests
UC1	Put explosive into the store	at7.txt
		at1.txt
UC2	Creates a new store	at2.txt
		at4.txt
UC3	Removes an explosive from the store	at3.txt
		at5.txt
		at6.txt

Table 1: Acceptance Tests for each Use Case

Category	**File Name**	**Purpose**
Basic	at0.txt	Basic use case
Basic with errors	at1.txt	Complicated use case with some errors
Error	at2.txt	Errors on naming of explosives and creating the store
Order of elements	at4.txt	Check order returned in large_explosives
Error Priorities	at5.txt	Check the priorities of errors for functions
Positioning	at6.txt	Positioning in suggest function
Use Case	at7.txt	Use case 1

Table 2: Acceptance tests in submitted code

13. Traceability Matrix

REQ	**at0.txt**	**at1.txt**	**at2.txt**	**at4.txt**	**at5.txt**	**at6.txt**	**at7.txt**
REQ4	X	X	X		X		X
REQ5	X	X	X	X	X	X	X
REQ6	X	X	X	X	X	X	X
REQ10	X	X	X	X		X	
REQ11	X	X	X			X	
REQ12		X	X	X	X		
REQ13	X	X	X		X		X
REQ14			X		X		
REQ15	X	X	X		X		X
REQ16		X	X		X		

14. Final Requirements Elicitation vs. Phase1

The differences between the phase 1 and the final requirements elicitation were pretty big. The biggest thing that was different was the fact that we assumed that there were no tuples in our phase 1 and we assumed that all of the functions just took variables for height, width, x position and y position. Since there were no tuples, the unit was not specified either and also we didn't mention the max and min for the variables.

The other parts were similar, just named differently and the functions all acted similarly(except the missing tuples). We also forgot to describe what each function was doing and the description of each variable.

15. More E- and R-Descriptions

ENV7	The largest possible store size is 10m x 10m	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test.

ENV8	The smallest store is 0.1m x 0.1m	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test.

ENV9	The smallest explosive is 0.1m x 0.1m	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test.

REQ10	The system shall allow users to remove explosives from the store if such an explosive exists.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ11	The system shall suggest suitable positions to store an explosive if at least one exists.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ12	The system shall list all explosives and their positions larger than or equal to a desired size if such an explosive exists.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ13	No action will be taken if it will cause an explosive to be overlapping with another explosive or out of bounds of the store	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ14	No store will be created if there already exists a store that has at least one explosive in it	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ15	No two explosives shall have the same ID	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

REQ16	All ID's shall be 2 characters long, the first character will be a letter and the second a number.	Traceability reference: e.g. see TLA model in Section 9, or see such an such an acceptance test

16. Appendices

A. Expected Output

For the precise output, see `/eecs/course/4312/project` as details may change during requirements elicitation (as announced on the forum).

```
  System State: (0) Create a new store
->new_store(6,6)
  System State: (1)
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
->put("A1",[1, 1],[0, 0])
  System State: (2)
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  A1 __ __ __ __ __
->put("B1",[2, 1],[1, 0])
  System State: (3)
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  A1 B1 B1 __ __ __
->put("C1",[3, 1],[3, 0])
  System State: (4)
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  A1 B1 B1 C1 C1 C1
->put("A2",[1, 2],[0, 1])
  System State: (5)
  __ __ __ __ __ __
  __ __ __ __ __ __
  __ __ __ __ __ __
  A2 __ __ __ __ __
```

```
  A2 __ __ __ __ __
  A1 B1 B1 C1 C1 C1
->put("A3",[1, 3],[0, 3])
  System State: (6)
  A3 __ __ __ __ __
  A3 __ __ __ __ __
  A3 __ __ __ __ __
  A2 __ __ __ __ __
  A2 __ __ __ __ __
  A1 B1 B1 C1 C1 C1
->put("A3",[1, 3],[0, 3])
  System State: (7) e03: ID already exists
->put("a9",[1, 3],[0, 3])
  System State: (8) e09: Explosive will overlap with another explosive
->suggest([1, 1])
  System State: (9)
  _____[1,5][2,5][3,5][4,5][5,5]
  _____[1,4][2,4][3,4][4,4][5,4]
  _____[1,3][2,3][3,3][4,3][5,3]
  _____[1,2][2,2][3,2][4,2][5,2]
  _____[1,1][2,1][3,1][4,1][5,1]
  ______________________________
->suggest([2, 2])
  System State: (10)
  ______________________________
  _____[1,4][2,4][3,4][4,4]_____
  _____[1,3][2,3][3,3][4,3]_____
  _____[1,2][2,2][3,2][4,2]_____
  _____[1,1][2,1][3,1][4,1]_____
  ______________________________
->put("z0",[1, 1],[3, 4])
  System State: (11)
  A3 __ __ __ __ __
  A3 __ __ z0 __ __
  A3 __ __ __ __ __
  A2 __ __ __ __ __
  A2 __ __ __ __ __
  A1 B1 B1 C1 C1 C1
->suggest([2, 2])
  System State: (12)
  ______________________________
  _____[1,4]_________[4,4]_____
  _____[1,3]_________[4,3]_____
  _____[1,2][2,2][3,2][4,2]_____
  _____[1,1][2,1][3,1][4,1]_____
  ______________________________
->put("B2",[2, 2],[1, 1])
```

```
  System State: (13)
  A3 __ __ __ __ __
  A3 __ __ z0 __ __
  A3 __ __ __ __ __
  A2 B2 B2 __ __ __
  A2 B2 B2 __ __ __
  A1 B1 B1 C1 C1 C1
->put("B3",[2, 3],[1, 3])
  System State: (14)
  A3 B3 B3 __ __ __
  A3 B3 B3 z0 __ __
  A3 B3 B3 __ __ __
  A2 B2 B2 __ __ __
  A2 B2 B2 __ __ __
  A1 B1 B1 C1 C1 C1
->put("C2",[3, 2],[3, 1])
  System State: (15)
  A3 B3 B3 __ __ __
  A3 B3 B3 z0 __ __
  A3 B3 B3 __ __ __
  A2 B2 B2 C2 C2 C2
  A2 B2 B2 C2 C2 C2
  A1 B1 B1 C1 C1 C1
->put("C3",[3, 3],[3, 3])
  System State: (16) e09: Explosive will overlap with another explosive
->remove("B2")
  System State: (17)
  A3 B3 B3 __ __ __
  A3 B3 B3 z0 __ __
  A3 B3 B3 __ __ __
  A2 __ __ C2 C2 C2
  A2 __ __ C2 C2 C2
  A1 B1 B1 C1 C1 C1
->remove("A1")
  System State: (18)
  A3 B3 B3 __ __ __
  A3 B3 B3 z0 __ __
  A3 B3 B3 __ __ __
  A2 __ __ C2 C2 C2
  A2 __ __ C2 C2 C2
  __ B1 B1 C1 C1 C1
->remove("C3")
  System State: (19) e10: ID does not exist
->suggest([2, 2])
  System State: (20)
  _____________________________
  _____________________[4,4]_____
```

```
  _____________________[4,3]_____
  _____________________[4,3]_____
  _____[1,1]___________[4,3]_____
  _____[1,1]___________[4,3]_____
->suggest([1, 1])
  System State: (21)
  _________________[3,5][4,5][5,5]
  _____________________[4,4][5,4]
  _________________[3,3][4,3][5,3]
  _____[1,2][2,2]_________________
  _____[1,1][2,1]_________________
  [0,0]___________________________
->suggest([3, 3])
  System State: (22)
  _____________________[4,3]_____
  _____[1,1]___________[4,3]_____
  _____[1,1]___________[4,3]_____
  _____[1,1]___________[4,3]_____
  _____[1,1]___________[4,3]_____
  _____[1,1]___________[4,3]_____
```

MODULE *explosive*

EXTENDS *Integers*, *Sequences*, *TLC*

CONSTANTS ID, N
ASSUME $N > 0$

VARIABLES
$taken\_id$,
$explosives$,
$xbound$,
$ybound$,
$spaces$,
$larger$

$vars \triangleq \langle taken\_id, explosives, xbound, ybound, spaces, larger \rangle$

$UNIT \triangleq 0 \,..\, N$

$UNIT\_RECORD \triangleq [x : UNIT, y : UNIT]$

$POSITION \triangleq \{r \in UNIT\_RECORD : r.x < N \land r.y < N\}$

$DIMENSION \triangleq \{r \in UNIT\_RECORD : r.x > 0 \land r.y > 0\}$

$EXPLOSIVE \triangleq [p : POSITION, d : DIMENSION]$

$TypeOk \triangleq$
$\land \; taken\_id \subseteq ID$
$\land \; explosives \in [taken\_id \to EXPLOSIVE]$
$\land \; xbound \in Int$
$\land \; ybound \in Int$
$\land \; spaces \subseteq POSITION$
$\land \; larger \subseteq taken\_id$

$Init \triangleq$
$\land \, taken\_id = \{\}$
$\land \, explosives = \langle\rangle$
$\land \, xbound = 0$
$\land \, ybound = 0$
$\land \, spaces = \{\}$
$\land \, larger = \{\}$

$IdOk \triangleq taken\_id = \text{DOMAIN } explosives$

$NoOverlap \triangleq$
$\land \forall \, i \in taken\_id : \forall \, j \in taken\_id \setminus \{i\} :$
$\lor \, explosives[i].p.x \geq explosives[j].p.x + explosives[j].d.x$
$\lor \, explosives[i].p.y \geq explosives[j].p.y + explosives[j].d.y$

$\lor\ explosives[j].p.x \geq explosives[i].p.x + explosives[i].d.x$
$\lor\ explosives[j].p.y \geq explosives[i].p.y + explosives[i].d.y$

$NoOutside\ \triangleq$
$\forall\, id \in taken\_id :$
$\land\ explosives[id].p.x + explosives[id].d.x \leq xbound$
$\land\ explosives[id].p.y + explosives[id].d.y \leq ybound$
$\land\ explosives[id].p.x \geq 0$
$\land\ explosives[id].p.y \geq 0$

$new\_store(xdim,\ ydim)\ \triangleq$
$\land\ taken\_id = \{\}$
$\land\ N \geq xdim$
$\land\ xdim > 0$
$\land\ N \geq ydim$
$\land\ xdim > 0$
$\land\ xbound' = xdim$
$\land\ ybound' = ydim$
$\land$ UNCHANGED $explosives$
$\land$ UNCHANGED $taken\_id$
$\land$ UNCHANGED $spaces$
$\land$ UNCHANGED $larger$
$\land\ Print(\langle$"new_store", $xdim,\ ydim\rangle,$ TRUE$)$

$put(id,\ dim,\ pos)\ \triangleq$
$\land\ id \in ID$
$\land\ id \notin taken\_id$
$\land\ pos.x\ \geq 0$
$\land\ pos.y\ \geq 0$
$\land\ dim.x > 0$
$\land\ dim.y > 0$
$\land\ (pos.x + dim.x) \leq xbound$
$\land\ (pos.y + dim.y) \leq ybound$
$\land\ \forall\, ide \in taken\_id :$
$\lor\ pos.x \geq explosives[ide].p.x + explosives[ide].d.x$
$\lor\ pos.y \geq explosives[ide].p.y + explosives[ide].d.y$
$\lor\ pos.x + dim.x \leq explosives[ide].p.x$
$\lor\ pos.y + dim.y \leq explosives[ide].p.y$
$\land\ taken\_id' = taken\_id \cup \{id\}$
$\land\ explosives' = explosives\ @@\ id :> [p \mapsto pos,\ d \mapsto dim]$
$\land$ UNCHANGED $xbound$
$\land$ UNCHANGED $ybound$
$\land$ UNCHANGED $spaces$
$\land$ UNCHANGED $larger$
$\land\ Print(\langle$"put", $id,\ dim,\ pos\rangle,$ TRUE$)$

$remove(id) \triangleq$
$\quad \land\ xbound > 0$
$\quad \land\ ybound > 0$
$\quad \land\ id \in taken\_id$
$\quad \land\ explosives' = [x \in \text{DOMAIN } explosives \setminus \{id\} \mapsto explosives[x]]$
$\quad \land\ taken\_id' = taken\_id \setminus \{id\}$
$\quad \land \text{ UNCHANGED } xbound$
$\quad \land \text{ UNCHANGED } ybound$
$\quad \land \text{ UNCHANGED } spaces$
$\quad \land \text{ UNCHANGED } larger$
$\quad \land\ Print(\langle \text{“remove”}, id \rangle, \text{ TRUE})$

$suggest(s) \triangleq$
$\quad \land\ xbound > 0$
$\quad \land\ ybound > 0$
$\quad \land\ s \in DIMENSION$
$\quad \land\ s.x \leq xbound$
$\quad \land\ s.y \leq ybound$
$\quad \land\ spaces' = \{\}$
$\quad \land\ \forall\, i \in 0 \,..\, xbound,\ j \in 0 \,..\, ybound,\ id \in taken\_id :$
$\qquad \land\ \lor\ i \geq explosives[id].p.x + explosives[id].d.x$
$\qquad\quad \lor\ j \geq explosives[id].p.y + explosives[id].d.y$
$\qquad\quad \lor\ i + s.x \leq explosives[id].p.x$
$\qquad\quad \lor\ j + s.y \leq explosives[id].p.y$
$\qquad \land\ spaces' = spaces \cup \{\langle i, j \rangle\}$
$\quad \land\ \text{UNCHANGED } xbound$
$\quad \land\ \text{UNCHANGED } ybound$
$\quad \land\ \text{UNCHANGED } explosives$
$\quad \land\ \text{UNCHANGED } taken\_id$
$\quad \land\ \text{UNCHANGED } larger$
$\quad \land\ Print(\langle \text{“suggest”}, s \rangle, \text{ TRUE})$

$large\_explosives(xdim,\ ydim) \triangleq$
$\quad \land\ xbound \geq xdim$
$\quad \land\ xdim > 0$
$\quad \land\ ybound \geq ydim$
$\quad \land\ xdim > 0$
$\quad \land\ larger' = \{\}$
$\quad \land\ \forall\, i \in 0 \,..\, xbound,\ j \in 0 \,..\, ybound,\ id \in taken\_id :$
$\qquad \land\ \land\ i \leq explosives[id].d.x$
$\qquad\quad \land\ j \leq explosives[id].d.y$
$\qquad \land\ larger' = larger \cup \{id\}$
$\quad \land \text{ UNCHANGED } xbound$
$\quad \land \text{ UNCHANGED } ybound$
$\quad \land \text{ UNCHANGED } explosives$

$\land$ UNCHANGED $taken\_id$
$\land$ UNCHANGED $spaces$
$\land$ $Print(\langle$ "large_explosives", $xdim$, $ydim\rangle$, TRUE)

$Next \triangleq$
$\lor \exists\, r \in DIMENSION : new\_store(r.x,\ r.y)$
$\lor \exists\, id \in ID,\ d \in DIMENSION,\ p \in POSITION : put(id,\ d,\ p)$
$\lor \exists\, id \in ID : remove(id)$
$\lor \exists\, r \in DIMENSION : suggest(r)$
$\lor \exists\, r \in DIMENSION : large\_explosives(r.x,\ r.y)$

$Spec \triangleq$
$\land\ Init$
$\land \Box[Next]_{vars}$