───────── MODULE *grpc* ─────────

Specification for the zebra-*grpc* crate design and it's relationship with the zebra-scanner crate and zebrad configuration file. It can handle the scan task functionality and how the *grpc* methods can add or delete information to the scanning database.

The spec is written in *PlusCal* and it's meant to be used with the *TLC* model checker.

The spec is divided in two parts: the first part is the *PlusCal* spec and the second part is translated TLA+ code.

The spec is divided in the following sections:

1. Configuration Constants
2. Global Variables
3. Type Invariants
4. Utility Functions
5. *gRPC* Methods
6. Services Process
7. Scan Task Process
8. Main Program Process

For more information visit: https://*github.com*/oxarbitrage/zebra-*grpc*-scan-spec)

EXTENDS *TLC*, *Integers*, *Sequences*, *Randomization*, *FiniteSets*, *Json*

CONFIGURATION CONSTANTS:

The set of keys as strings to be added to the scan task from the config file.
CONSTANT *ConfigViewingKeys*

We have 3 batches of keys so we can try different combinations, including duplicated keys.

A set of keys as strings.
CONSTANT *GrpcViewingKeysBatch*1
A second set of keys as strings.
CONSTANT *GrpcViewingKeysBatch*2
A third set of keys as strings.
CONSTANT *GrpcViewingKeysBatch*3
The maximum number of scan tasks that can be added to the scan task set.
CONSTANT *MaxScanTasks*

GLOBAL VARIABLES:

A sequence of batches with keys to call *grpc* methods. Currently we have 3 batches.
$GrpcViewingKeys \triangleq \langle GrpcViewingKeysBatch1, GrpcViewingKeysBatch2, GrpcViewingKeysBatch3 \rangle$

A dummy response to an *Info request*.
$info\_response \triangleq ToJson([saplingheight \mapsto 1])$
A random list of transations to be used as a Results response.
$results\_response \triangleq ToJson([transactions \mapsto RandomSetOfSubsets(1, 3, 1 .. 10)])$
An empty response to Register

$register\_response \triangleq ToJson([empty \mapsto \{\}])$
An empty response to Delete
$delete\_response \triangleq ToJson([empty \mapsto \{\}])$
An empty response to Clear
$clear\_response \triangleq ToJson([empty \mapsto \{\}])$
An empty response to Subscribe TODO: which should be a channel with updates.
$subscribe\_response \triangleq ToJson([empty \mapsto \{\}])$
An empty response to Status TODO: which should have some data from the scan task for the key.
$status\_response \triangleq ToJson([empty \mapsto \{\}])$
The set of statuses a scan task can be on at any given time.
$scan\_task\_statuses \triangleq \{\text{"waiting"}, \text{"adding"}, \text{"deleting"}\}$
The set of valid service requests.
$service\_requests \triangleq \{\text{"waiting"}, \text{"info"}, \text{"results"}, \text{"clear"}, \text{"status"}, \text{"register"}, \text{"delete"}, \text{"subscribe"}\}$

--algorithm *grpc*
variables
The scan tasks are a set that is initially empty.
$scan\_tasks = \{\}$;
A string that will be used as a response to any of the gRPC method calls.
$response = \text{""}$;
The status of the scan task, initially listening.
$scan\_task\_status = \text{"waiting"}$;
A key to be passed to any of the services, and also added or deleted to/from the scan task at a given instant, initially empty.
$key\_to\_be\_served = \text{""}$;
The current service request flag.
$service\_request = \text{"waiting"}$;
The number of batches the configuration has.
$number\_of\_batches = 0$;
The counter for the number of batches.
$counter = 1$;

define
THE TYPE INVARIANT :
$TypeInvariant \triangleq$
The response is always in the STRING *domain*
$\land response \in \text{STRING}$
The scan task status is always in the scan task statuses set.
$\land scan\_task\_status \in scan\_task\_statuses$
The key to be served is always in the STRING *domain.*
$\land key\_to\_be\_served \in \text{STRING}$
The service request is always in the service requests set.
$\land service\_request \in service\_requests$
LIVENESS PROPERTIES :
$ScanTaskLiveness \triangleq$

2

The ScanTask process always reachs a waiting state.
$\diamond(scan\_task\_status = \text{"waiting"})$

$ServiceLiveness \triangleq$
   The Services process always reachs a waiting state.
   $\diamond(service\_request = \text{"waiting"})$

end define ;

UTILITY FUNCTIONS::

Helper function to get the number of non empty batches the configuration has.
procedure $get\_config\_number\_of\_batches()$
begin
   $CheckBatch1$:
      if $GrpcViewingKeysBatch1 \neq \{\}$ **then**
         $number\_of\_batches := number\_of\_batches + 1$;
      end if ;
   $CheckBatch2$:
      if $GrpcViewingKeysBatch2 \neq \{\}$ **then**
         $number\_of\_batches := number\_of\_batches + 1$;
      end if ;
   $CheckBatch3$:
      if $GrpcViewingKeysBatch3 \neq \{\}$ **then**
         $number\_of\_batches := number\_of\_batches + 1$;
      end if ;
      return ;
end procedure ;

Call the scan task to add keys coming from the config file.
procedure $add\_config\_keys(keys)$
begin
   $AddConfigKeys$:
      with $key \in keys$ **do**
         $key\_to\_be\_served := key$;
         $scan\_task\_status := \text{"adding"}$;
         return ;
      end with ;
end procedure ;

GRPC METHODS :

The get_info grpc method.
procedure $get\_info()$
begin
   $InfoServiceRequest$:
      $service\_request := \text{"info"}$;
      return ;

3

end procedure ;

The get_results grpc method.
procedure *get_results*(*keys*)
begin
    ResultsServiceRequest:
        with *key* ∈ *keys* **do**
            key_to_be_served := *key* ;
            service_request := "results" ;
            return ;
        end with ;
end procedure ;

The clear_results grpc method.
procedure *clear_results*(*keys*)
begin
    ClearServiceRequest:
        with *key* ∈ *keys* **do**
            key_to_be_served := *key* ;
            service_request := "clear" ;
            return ;
        end with ;
end procedure ;

The get_status grpc method.
procedure *get_status*(*keys*)
begin
    StatusServiceRequest:
        with *key* ∈ *keys* **do**
            key_to_be_served := *key* ;
            service_request := "status" ;
            return ;
        end with ;
end procedure ;

The register_keys grpc method.
procedure *register_keys*(*keys*)
begin
    RegisterServiceRequest:
        with *key* ∈ *keys* **do**
            key_to_be_served := *key* ;
            service_request := "register" ;
            return ;
        end with ;
end procedure ;

The delete_keys grpc method.

procedure *delete_keys(keys)*
begin
    DeleteServiceRequest:
        with *key* ∈ *keys* **do**
            key_to_be_served := key ;
            service_request := "delete" ;
            return ;
        end with ;
end procedure ;

The scan grpc method.

The method call 3 services one next to the other.

procedure *scan(keys)*
begin
    RegisterServiceRequestFromScan:
        with *key* ∈ *keys* **do**
            key_to_be_served := key ;
            service_request := "register" ;
        end with ;
    ResultsServiceRequestFromScan:
        with *key* ∈ *keys* **do**
            key_to_be_served := key ;
            service_request := "results" ;
        end with ;
    SubscribeServiceRequestFromScan:
        with *key* ∈ *keys* **do**
            key_to_be_served := key ;
            service_request := "subscribe" ;
            return ;
        end with ;
end procedure ;

SERVICES PROCESS :

Listen for requests, send requests to scan task where is needed and provide responses.

process *services =* "SERVICES"
begin
    Services:
        if *service_request =* "info" **then**
            Info:
                response := info_response ;
        elsif *service_request =* "results" **then**
            Results:
                if *key_to_be_served* ∈ *scan_tasks* **then**
                    response := results_response ;

```
            else
                response := "Error: key not found." ;
            end if ;
    elsif service_request = "clear" then
        Clear:
            if key_to_be_served ∈ scan_tasks then
                response := clear_response ;
            else
                response := "Error: key not found." ;
            end if ;
    elsif service_request = "status" then
        Status:
            if key_to_be_served ∈ scan_tasks then
                response := status_response ;
            else
                response := "Error: key not found." ;
            end if ;
    elsif service_request = "register" then
        Register:
            if key_to_be_served ∈ scan_tasks then
                KeyError:
                    response := "Error: key already in scan task." ;
            else
                Success:
                    scan_task_status := "adding" ;
                    response := register_response ;
            end if ;
    elsif service_request = "delete" then
        Delete:
            if key_to_be_served ∈ scan_tasks then
                scan_task_status := "deleting" ;
                response := delete_response ;
            else
                response := "Error: key not found." ;
            end if ;
    elsif service_request = "subscribe" then
        Subscribe:
            if key_to_be_served ∈ scan_tasks then
                response := subscribe_response ;
            else
                response := "Error: key not found." ;
            end if ;
    end if ;
ClearRequestFlag:
    service_request := "waiting" ;
```

Make the process loops forever.
  ServicesLoop :
    goto *Services* ;
end process ;

SCAN TASK PROCESS :

Listen for requests from the services process, add or remove tasks to the scan task set.
fair process *scantask* = "SCAN TASK"
variables *inner_state* = {} ;
begin
  GetScanTasks :
    inner_state := *scan_tasks* ;
  ScanTask :
    if *Cardinality*(*scan_tasks*) > *MaxScanTasks* **then**
      BoundError :
        response := "Error: max scan tasks reached." ;
        scan_task_status := "waiting" ;
    elsif *scan_task_status* = "adding" **then**
      Adding :
        inner_state := *inner_state* ∪ {*key_to_be_served*} ;
        scan_task_status := "waiting" ;
    elsif *scan_task_status* = "deleting" **then**
      Deleting :
        scan_tasks := *scan_tasks* \ {*key_to_be_served*} ;
        scan_task_status := "waiting" ;
    end if ;
  StoreScanTasks :
    scan_tasks := *inner_state* ;
    Make the process loops forever.
  ScanTaskLoop :
    goto *ScanTask* ;
end process ;

MAIN PROCESS :

Calls all grpc methods with the given keys.
process *Main* = "MAIN"
begin
  ConfigGuard :
    if *ConfigViewingKeys* ≠ {} **then**
      FromZebradConfig :
        call *add_config_keys*(*ConfigViewingKeys*) ;
    end if ;
  ListeningGuard :
    if *GrpcViewingKeys* ≠ ⟨⟩ **then**

GetTotalIterationsToMake:
    call *get_config_number_of_batches*() ;
ListeningMode:
    while $counter \leq number\_of\_batches$ **do**
        GetInfoCall:
          call *get_info*() ;
        RegisterKeysCall:
          call *register_keys*(*GrpcViewingKeys*[*counter*]) ;
        GetStatusCall:
          call *get_status*(*GrpcViewingKeys*[*counter*]) ;
        GetResultsCall:
          call *get_results*(*GrpcViewingKeys*[*counter*]) ;
        ClearResultsCall:
          call *clear_results*(*GrpcViewingKeys*[*counter*]) ;
        DeleteKeysCall:
          call *delete_keys*(*GrpcViewingKeys*[*counter*]) ;
        ScanCall:
          call *scan*(*GrpcViewingKeys*[*counter*]) ;
        IncrementCounter:
          $counter := counter + 1$;
    end while ;
    goto *End* ;
  end if ;
End:
  skip ;

end process ;
end algorithm ;
BEGIN TRANSLATION(*chksum*(*pcal*) = "f6c668a3" $\land$ *chksum*(*tla*) = "48d8f3a2")
Parameter keys of procedure add_config_keys at line 130 *col* 27 *changed to keys_*
Parameter keys of procedure get_results at line 151 *col* 23 *changed to keys_g*
Parameter keys of procedure clear_results at line 162 *col* 25 *changed to keys_c*
Parameter keys of procedure get_status at line 173 *col* 22 *changed to keys_ge*
Parameter keys of procedure register_keys at line 184 *col* 25 *changed to keys_r*
Parameter keys of procedure delete_keys at line 195 *col* 23 *changed to keys_d*
CONSTANT *defaultInitValue*
VARIABLES *scan_tasks*, *response*, *scan_task_status*, *key_to_be_served*,
        service_request, *number_of_batches*, *counter*, *pc*, *stack*

define statement
TypeInvariant $\triangleq$

    $\land$ *response* $\in$ STRING

    $\land$ *scan_task_status* $\in$ *scan_task_statuses*

$\land key\_to\_be\_served \in \text{STRING}$

$\land service\_request \in service\_requests$

$ScanTaskLiveness \triangleq$

$\quad\Diamond(scan\_task\_status = \text{``waiting''})$
$ServiceLiveness \triangleq$

$\quad\Diamond(service\_request = \text{``waiting''})$

VARIABLES $keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state$

$vars \triangleq \langle scan\_tasks,\ response,\ scan\_task\_status,\ key\_to\_be\_served,$
$\qquad\qquad service\_request,\ number\_of\_batches,\ counter,\ pc,\ stack,\ keys\_,$
$\qquad\qquad keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state \rangle$

$ProcSet \triangleq \{ \text{``SERVICES''} \} \cup \{ \text{``SCAN TASK''} \} \cup \{ \text{``MAIN''} \}$

$Init \triangleq \quad$ *Global variables*
$\qquad\quad \land scan\_tasks = \{\}$
$\qquad\quad \land response = \text{``''}$
$\qquad\quad \land scan\_task\_status = \text{``waiting''}$
$\qquad\quad \land key\_to\_be\_served = \text{``''}$
$\qquad\quad \land service\_request = \text{``waiting''}$
$\qquad\quad \land number\_of\_batches = 0$
$\qquad\quad \land counter = 1$
$\qquad\quad$ *Procedure add_config_keys*
$\qquad\quad \land keys\_ = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Procedure get_results*
$\qquad\quad \land keys\_g = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Procedure clear_results*
$\qquad\quad \land keys\_c = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Procedure get_status*
$\qquad\quad \land keys\_ge = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Procedure register_keys*
$\qquad\quad \land keys\_r = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Procedure delete_keys*
$\qquad\quad \land keys\_d = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Procedure scan*
$\qquad\quad \land keys = [self \in ProcSet \mapsto defaultInitValue]$
$\qquad\quad$ *Process scantask*
$\qquad\quad \land inner\_state = \{\}$
$\qquad\quad \land stack = [self \in ProcSet \mapsto \langle\rangle]$
$\qquad\quad \land pc = [self \in ProcSet \mapsto \text{CASE } self = \text{``SERVICES''} \to \text{``Services''}$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \square \quad self = \text{``SCAN TASK''} \to \text{``GetScanTasks''}$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \square \quad self = \text{``MAIN''} \to \text{``ConfigGuard''}]$

$CheckBatch1(self)$ $\triangleq$ $\land pc[self] =$ "CheckBatch1"
$\quad\land$ IF $GrpcViewingKeysBatch1 \neq \{\}$
$\qquad\quad$ THEN $\quad\land number\_of\_batches' = number\_of\_batches + 1$
$\qquad\quad$ ELSE $\quad\land$ TRUE
$\qquad\qquad\qquad\land$ UNCHANGED $number\_of\_batches$
$\quad\land pc' = [pc$ EXCEPT $![self] =$ "CheckBatch2"$]$
$\quad\land$ UNCHANGED $\langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad counter,\ stack,\ keys\_,\ keys\_g,\ keys\_c,$
$\qquad\qquad\qquad\qquad keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
$\qquad\qquad\qquad\qquad inner\_state\rangle$

$CheckBatch2(self)$ $\triangleq$ $\land pc[self] =$ "CheckBatch2"
$\quad\land$ IF $GrpcViewingKeysBatch2 \neq \{\}$
$\qquad\quad$ THEN $\quad\land number\_of\_batches' = number\_of\_batches + 1$
$\qquad\quad$ ELSE $\quad\land$ TRUE
$\qquad\qquad\qquad\land$ UNCHANGED $number\_of\_batches$
$\quad\land pc' = [pc$ EXCEPT $![self] =$ "CheckBatch3"$]$
$\quad\land$ UNCHANGED $\langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad counter,\ stack,\ keys\_,\ keys\_g,\ keys\_c,$
$\qquad\qquad\qquad\qquad keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
$\qquad\qquad\qquad\qquad inner\_state\rangle$

$CheckBatch3(self)$ $\triangleq$ $\land pc[self] =$ "CheckBatch3"
$\quad\land$ IF $GrpcViewingKeysBatch3 \neq \{\}$
$\qquad\quad$ THEN $\quad\land number\_of\_batches' = number\_of\_batches + 1$
$\qquad\quad$ ELSE $\quad\land$ TRUE
$\qquad\qquad\qquad\land$ UNCHANGED $number\_of\_batches$
$\quad\land pc' = [pc$ EXCEPT $![self] = Head(stack[self]).pc]$
$\quad\land stack' = [stack$ EXCEPT $![self] = Tail(stack[self])]$
$\quad\land$ UNCHANGED $\langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad counter,\ keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,$
$\qquad\qquad\qquad\qquad keys\_r,\ keys\_d,\ keys,\ inner\_state\rangle$

$get\_config\_number\_of\_batches(self)$ $\triangleq$ $CheckBatch1(self)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \lor CheckBatch2(self)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \lor CheckBatch3(self)$

$AddConfigKeys(self)$ $\triangleq$ $\land pc[self] =$ "AddConfigKeys"
$\quad\land \exists\, key \in keys\_[self] :$
$\qquad\land key\_to\_be\_served' = key$
$\qquad\land scan\_task\_status' =$ "adding"
$\qquad\land pc' = [pc$ EXCEPT $![self] = Head(stack[self]).pc]$
$\qquad\land keys\_' = [keys\_$ EXCEPT $![self] = Head(stack[self]).keys\_]$

$$\wedge\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$$
$$\wedge \text{ UNCHANGED } \langle scan\_tasks,\ response,\ service\_request,$$
$$number\_of\_batches,\ counter,\ keys\_g,$$
$$keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$$
$$inner\_state \rangle$$

$add\_config\_keys(self) \triangleq AddConfigKeys(self)$

$InfoServiceRequest(self) \triangleq$
$$\wedge\ pc[self] = \text{``InfoServiceRequest''}$$
$$\wedge\ service\_request' = \text{``info''}$$
$$\wedge\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$$
$$\wedge\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$$
$$\wedge \text{ UNCHANGED } \langle scan\_tasks,\ response,$$
$$scan\_task\_status,\ key\_to\_be\_served,$$
$$number\_of\_batches,\ counter,\ keys\_,$$
$$keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,$$
$$keys\_d,\ keys,\ inner\_state \rangle$$

$get\_info(self) \triangleq InfoServiceRequest(self)$

$ResultsServiceRequest(self) \triangleq$
$$\wedge\ pc[self] = \text{``ResultsServiceRequest''}$$
$$\wedge\ \exists\, key \in keys\_g[self] :$$
$$\wedge\ key\_to\_be\_served' = key$$
$$\wedge\ service\_request' = \text{``results''}$$
$$\wedge\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$$
$$\wedge\ keys\_g' = [keys\_g \text{ EXCEPT } ![self] = Head(stack[self]).keys\_g]$$
$$\wedge\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$$
$$\wedge \text{ UNCHANGED } \langle scan\_tasks,\ response,$$
$$scan\_task\_status,$$
$$number\_of\_batches,\ counter,$$
$$keys\_,\ keys\_c,\ keys\_ge,\ keys\_r,$$
$$keys\_d,\ keys,\ inner\_state \rangle$$

$get\_results(self) \triangleq ResultsServiceRequest(self)$

$ClearServiceRequest(self) \triangleq$
$$\wedge\ pc[self] = \text{``ClearServiceRequest''}$$
$$\wedge\ \exists\, key \in keys\_c[self] :$$
$$\wedge\ key\_to\_be\_served' = key$$
$$\wedge\ service\_request' = \text{``clear''}$$
$$\wedge\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$$
$$\wedge\ keys\_c' = [keys\_c \text{ EXCEPT } ![self] = Head(stack[self]).keys\_c]$$
$$\wedge\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$$
$$\wedge \text{ UNCHANGED } \langle scan\_tasks,\ response,$$
$$scan\_task\_status,$$
$$number\_of\_batches,\ counter,\ keys\_,$$
$$keys\_g,\ keys\_ge,\ keys\_r,\ keys\_d,$$
$$keys,\ inner\_state \rangle$$

$clear\_results(self) \triangleq ClearServiceRequest(self)$

$StatusServiceRequest(self) \triangleq \wedge pc[self] = \text{``StatusServiceRequest''}$
$\qquad\qquad\qquad\qquad\qquad\quad \wedge \exists\, key \in keys\_ge[self] :$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge key\_to\_be\_served' = key$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge service\_request' = \text{``status''}$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge keys\_ge' = [keys\_ge \text{ EXCEPT } ![self] = Head(stack[self]).keys\_ge]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
$\qquad\qquad\qquad\qquad\qquad\quad \wedge \text{UNCHANGED } \langle scan\_tasks, response,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches, counter,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_, keys\_g, keys\_c, keys\_r,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_d, keys, inner\_state \rangle$

$get\_status(self) \triangleq StatusServiceRequest(self)$

$RegisterServiceRequest(self) \triangleq \wedge pc[self] = \text{``RegisterServiceRequest''}$
$\qquad\qquad\qquad\qquad\qquad\qquad \wedge \exists\, key \in keys\_r[self] :$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge key\_to\_be\_served' = key$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge service\_request' = \text{``register''}$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge keys\_r' = [keys\_r \text{ EXCEPT } ![self] = Head(stack[self]).keys\_r]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
$\qquad\qquad\qquad\qquad\qquad\qquad \wedge \text{UNCHANGED } \langle scan\_tasks, response,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches, counter,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_, keys\_g, keys\_c, keys\_ge,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_d, keys, inner\_state \rangle$

$register\_keys(self) \triangleq RegisterServiceRequest(self)$

$DeleteServiceRequest(self) \triangleq \wedge pc[self] = \text{``DeleteServiceRequest''}$
$\qquad\qquad\qquad\qquad\qquad\quad \wedge \exists\, key \in keys\_d[self] :$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge key\_to\_be\_served' = key$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge service\_request' = \text{``delete''}$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge keys\_d' = [keys\_d \text{ EXCEPT } ![self] = Head(stack[self]).keys\_d]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
$\qquad\qquad\qquad\qquad\qquad\quad \wedge \text{UNCHANGED } \langle scan\_tasks, response,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches, counter,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_, keys\_g, keys\_c, keys\_ge,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_r, keys, inner\_state \rangle$

$delete\_keys(self) \triangleq DeleteServiceRequest(self)$

$RegisterServiceRequestFromScan(self) \;\triangleq\; \wedge\; pc[self] =$ "RegisterServiceRequestFromScan"
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; \exists\, key \in keys[self] :$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; key\_to\_be\_served' = key$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; service\_request' =$ "register"
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; pc' = [pc \text{ EXCEPT } ![self] =$ "ResultsServiceRequestFromScan"$]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; \text{UNCHANGED } \langle scan\_tasks,\, response,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad counter,\, stack,\, keys\_,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_g,\, keys\_c,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_ge,\, keys\_r,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_d,\, keys,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad inner\_state \rangle$

$ResultsServiceRequestFromScan(self) \;\triangleq\; \wedge\; pc[self] =$ "ResultsServiceRequestFromScan"
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; \exists\, key \in keys[self] :$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; key\_to\_be\_served' = key$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; service\_request' =$ "results"
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; pc' = [pc \text{ EXCEPT } ![self] =$ "SubscribeServiceRequestFromScan"$]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; \text{UNCHANGED } \langle scan\_tasks,\, response,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad counter,\, stack,\, keys\_,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_g,\, keys\_c,\, keys\_ge,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_r,\, keys\_d,\, keys,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad inner\_state \rangle$

$SubscribeServiceRequestFromScan(self) \;\triangleq\; \wedge\; pc[self] =$ "SubscribeServiceRequestFromScan"
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; \exists\, key \in keys[self] :$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; key\_to\_be\_served' = key$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; service\_request' =$ "subscribe"
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; keys' = [keys \text{ EXCEPT } ![self] = Head(stack[self]).keys]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad \wedge\; \text{UNCHANGED } \langle scan\_tasks,\, response,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad counter,\, keys\_,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_g,\, keys\_c,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_ge,\, keys\_r,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_d,\, inner\_state \rangle$

$scan(self) \;\triangleq\; RegisterServiceRequestFromScan(self)$
$\qquad\qquad\qquad \vee\; ResultsServiceRequestFromScan(self)$
$\qquad\qquad\qquad \vee\; SubscribeServiceRequestFromScan(self)$

13

$Services \triangleq \land pc[\text{"SERVICES"}] = \text{"Services"}$
$\land$ IF $service\_request = \text{"info"}$
    THEN $\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"Info"}]$
    ELSE $\land$ IF $service\_request = \text{"results"}$
        THEN $\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"Results"}]$
        ELSE $\land$ IF $service\_request = \text{"clear"}$
            THEN $\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"Clear"}]$
            ELSE $\land$ IF $service\_request = \text{"status"}$
                THEN $\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"Stat}$
                ELSE $\land$ IF $service\_request = \text{"register"}$
                    THEN $\land pc' = [pc$ EXCEPT $![\text{"SERVIC}$
                    ELSE $\land$ IF $service\_request = \text{"delete"}$
                        THEN $\land pc' = [pc$ EXCEP
                        ELSE $\land$ IF $service\_reque$
                            THEN $\land pc'$
                            ELSE $\land pc'$
$\land$ UNCHANGED $\langle scan\_tasks,\ response,\ scan\_task\_status,$
                $key\_to\_be\_served,\ service\_request,$
                $number\_of\_batches,\ counter,\ stack,\ keys\_,\ keys\_g,$
                $keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state\rangle$

$Info \triangleq \land pc[\text{"SERVICES"}] = \text{"Info"}$
$\land response' = info\_response$
$\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"ClearRequestFlag"}]$
$\land$ UNCHANGED $\langle scan\_tasks,\ scan\_task\_status,\ key\_to\_be\_served,$
                $service\_request,\ number\_of\_batches,\ counter,\ stack,$
                $keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
                $inner\_state\rangle$

$Results \triangleq \land pc[\text{"SERVICES"}] = \text{"Results"}$
$\land$ IF $key\_to\_be\_served \in scan\_tasks$
    THEN $\land response' = results\_response$
    ELSE $\land response' = \text{"Error: key not found."}$
$\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"ClearRequestFlag"}]$
$\land$ UNCHANGED $\langle scan\_tasks,\ scan\_task\_status,\ key\_to\_be\_served,$
                $service\_request,\ number\_of\_batches,\ counter,\ stack,$
                $keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,$
                $keys,\ inner\_state\rangle$

$Clear \triangleq \land pc[\text{"SERVICES"}] = \text{"Clear"}$
$\land$ IF $key\_to\_be\_served \in scan\_tasks$
    THEN $\land response' = clear\_response$
    ELSE $\land response' = \text{"Error: key not found."}$
$\land pc' = [pc$ EXCEPT $![\text{"SERVICES"}] = \text{"ClearRequestFlag"}]$
$\land$ UNCHANGED $\langle scan\_tasks,\ scan\_task\_status,\ key\_to\_be\_served,$
                $service\_request,\ number\_of\_batches,\ counter,\ stack,$

14

$$\langle keys\_, keys\_g, keys\_c, keys\_ge, keys\_r, keys\_d, keys,$$
$$inner\_state \rangle$$

$Status \triangleq \;\land pc[\text{``SERVICES''}] = \text{``Status''}$
$\quad\quad\quad\; \land \text{IF } key\_to\_be\_served \in scan\_tasks$
$\quad\quad\quad\quad\quad \text{THEN} \;\; \land response' = status\_response$
$\quad\quad\quad\quad\quad \text{ELSE} \;\;\; \land response' = \text{``Error: key not found.''}$
$\quad\quad\quad\; \land pc' = [pc \text{ EXCEPT } ![\text{``SERVICES''}] = \text{``ClearRequestFlag''}]$
$\quad\quad\quad\; \land \text{UNCHANGED } \langle scan\_tasks, scan\_task\_status, key\_to\_be\_served,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad service\_request, number\_of\_batches, counter, stack,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad keys\_, keys\_g, keys\_c, keys\_ge, keys\_r, keys\_d, keys,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad inner\_state \rangle$

$Register \;\;\triangleq\; \land pc[\text{``SERVICES''}] = \text{``Register''}$
$\quad\quad\quad\quad\; \land \text{IF } key\_to\_be\_served \in scan\_tasks$
$\quad\quad\quad\quad\quad\quad \text{THEN} \;\; \land pc' = [pc \text{ EXCEPT } ![\text{``SERVICES''}] = \text{``KeyError''}]$
$\quad\quad\quad\quad\quad\quad \text{ELSE} \;\;\; \land pc' = [pc \text{ EXCEPT } ![\text{``SERVICES''}] = \text{``Success''}]$
$\quad\quad\quad\quad\; \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad key\_to\_be\_served, service\_request,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad number\_of\_batches, counter, stack, keys\_, keys\_g,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad keys\_c, keys\_ge, keys\_r, keys\_d, keys, inner\_state \rangle$

$KeyError \;\triangleq\; \land pc[\text{``SERVICES''}] = \text{``KeyError''}$
$\quad\quad\quad\quad\; \land response' = \text{``Error: key already in scan task.''}$
$\quad\quad\quad\quad\; \land pc' = [pc \text{ EXCEPT } ![\text{``SERVICES''}] = \text{``ClearRequestFlag''}]$
$\quad\quad\quad\quad\; \land \text{UNCHANGED } \langle scan\_tasks, scan\_task\_status, key\_to\_be\_served,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad service\_request, number\_of\_batches, counter, stack,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad keys\_, keys\_g, keys\_c, keys\_ge, keys\_r, keys\_d,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad keys, inner\_state \rangle$

$Success \;\triangleq\; \land pc[\text{``SERVICES''}] = \text{``Success''}$
$\quad\quad\quad\quad \land scan\_task\_status' = \text{``adding''}$
$\quad\quad\quad\quad \land response' = register\_response$
$\quad\quad\quad\quad \land pc' = [pc \text{ EXCEPT } ![\text{``SERVICES''}] = \text{``ClearRequestFlag''}]$
$\quad\quad\quad\quad \land \text{UNCHANGED } \langle scan\_tasks, key\_to\_be\_served, service\_request,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\; number\_of\_batches, counter, stack, keys\_, keys\_g,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\; keys\_c, keys\_ge, keys\_r, keys\_d, keys, inner\_state \rangle$

$Delete \;\triangleq\; \land pc[\text{``SERVICES''}] = \text{``Delete''}$
$\quad\quad\quad\quad \land \text{IF } key\_to\_be\_served \in scan\_tasks$
$\quad\quad\quad\quad\quad\quad \text{THEN} \;\; \land scan\_task\_status' = \text{``deleting''}$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad \land response' = delete\_response$
$\quad\quad\quad\quad\quad\quad \text{ELSE} \;\;\; \land response' = \text{``Error: key not found.''}$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad \land \text{UNCHANGED } scan\_task\_status$
$\quad\quad\quad\quad \land pc' = [pc \text{ EXCEPT } ![\text{``SERVICES''}] = \text{``ClearRequestFlag''}]$
$\quad\quad\quad\quad \land \text{UNCHANGED } \langle scan\_tasks, key\_to\_be\_served, service\_request,$

$$\langle number\_of\_batches,\ counter,\ stack,\ keys\_,\ keys\_g,$$
$$keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state \rangle$$

$Subscribe \triangleq\ \wedge pc[\text{``SERVICES''}] = \text{``Subscribe''}$
$\qquad\qquad\ \wedge \text{IF}\ key\_to\_be\_served \in scan\_tasks$
$\qquad\qquad\qquad\ \text{THEN}\ \wedge response' = subscribe\_response$
$\qquad\qquad\qquad\ \text{ELSE}\ \ \wedge response' = \text{``Error: key not found."}$
$\qquad\qquad\ \wedge pc' = [pc\ \text{EXCEPT}\ !\,[\text{``SERVICES''}] = \text{``ClearRequestFlag''}]$
$\qquad\qquad\ \wedge \text{UNCHANGED}\ \langle scan\_tasks,\ scan\_task\_status,\ key\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad service\_request,\ number\_of\_batches,\ counter,$
$\qquad\qquad\qquad\qquad\qquad stack,\ keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,$
$\qquad\qquad\qquad\qquad\qquad keys\_d,\ keys,\ inner\_state \rangle$

$ClearRequestFlag \triangleq\ \wedge pc[\text{``SERVICES''}] = \text{``ClearRequestFlag''}$
$\qquad\qquad\qquad\ \wedge service\_request' = \text{``waiting''}$
$\qquad\qquad\qquad\ \wedge pc' = [pc\ \text{EXCEPT}\ !\,[\text{``SERVICES''}] = \text{``ServicesLoop''}]$
$\qquad\qquad\qquad\ \wedge \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,\ number\_of\_batches,$
$\qquad\qquad\qquad\qquad\qquad counter,\ stack,\ keys\_,\ keys\_g,\ keys\_c,$
$\qquad\qquad\qquad\qquad\qquad keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state \rangle$

$ServicesLoop \triangleq\ \wedge pc[\text{``SERVICES''}] = \text{``ServicesLoop''}$
$\qquad\qquad\qquad \wedge pc' = [pc\ \text{EXCEPT}\ !\,[\text{``SERVICES''}] = \text{``Services''}]$
$\qquad\qquad\qquad \wedge \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ stack,\ keys\_,$
$\qquad\qquad\qquad\qquad\qquad keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
$\qquad\qquad\qquad\qquad\qquad inner\_state \rangle$

$services \triangleq\ Services \vee Info \vee Results \vee Clear \vee Status \vee Register$
$\qquad\qquad\ \vee KeyError \vee Success \vee Delete \vee Subscribe$
$\qquad\qquad\ \vee ClearRequestFlag \vee ServicesLoop$

$GetScanTasks \triangleq\ \wedge pc[\text{``SCAN TASK''}] = \text{``GetScanTasks''}$
$\qquad\qquad\qquad \wedge inner\_state' = scan\_tasks$
$\qquad\qquad\qquad \wedge pc' = [pc\ \text{EXCEPT}\ !\,[\text{``SCAN TASK''}] = \text{``ScanTask''}]$
$\qquad\qquad\qquad \wedge \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ stack,\ keys\_,$
$\qquad\qquad\qquad\qquad\qquad keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys \rangle$

$ScanTask \triangleq\ \wedge pc[\text{``SCAN TASK''}] = \text{``ScanTask''}$
$\qquad\qquad\ \wedge \text{IF}\ Cardinality(scan\_tasks) > MaxScanTasks$
$\qquad\qquad\qquad \text{THEN}\ \wedge pc' = [pc\ \text{EXCEPT}\ !\,[\text{``SCAN TASK''}] = \text{``BoundError''}]$
$\qquad\qquad\qquad \text{ELSE}\ \ \wedge \text{IF}\ scan\_task\_status = \text{``adding''}$
$\qquad\qquad\qquad\qquad\qquad \text{THEN}\ \wedge pc' = [pc\ \text{EXCEPT}\ !\,[\text{``SCAN TASK''}] = \text{``Adding''}]$
$\qquad\qquad\qquad\qquad\qquad \text{ELSE}\ \ \wedge \text{IF}\ scan\_task\_status = \text{``deleting''}$

$$\text{THEN} \quad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``Deleting''}]$$
$$\text{ELSE} \quad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``StoreScanTas}$$
$$\wedge \text{UNCHANGED } \langle scan\_tasks,\ response,\ scan\_task\_status,$$
$$key\_to\_be\_served,\ service\_request,$$
$$number\_of\_batches,\ counter,\ stack,\ keys\_,\ keys\_g,$$
$$keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state \rangle$$

$BoundError \triangleq \wedge pc[\text{``SCAN TASK''}] = \text{``BoundError''}$
$\qquad\qquad\quad \wedge response' = \text{``Error: max scan tasks reached.''}$
$\qquad\qquad\quad \wedge scan\_task\_status' = \text{``waiting''}$
$\qquad\qquad\quad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``StoreScanTasks''}]$
$\qquad\qquad\quad \wedge \text{UNCHANGED } \langle scan\_tasks,\ key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ stack,\ keys\_,\ keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
$\qquad\qquad\qquad\qquad\qquad\qquad inner\_state \rangle$

$Adding \triangleq \wedge pc[\text{``SCAN TASK''}] = \text{``Adding''}$
$\qquad\qquad \wedge inner\_state' = (inner\_state \cup \{key\_to\_be\_served\})$
$\qquad\qquad \wedge scan\_task\_status' = \text{``waiting''}$
$\qquad\qquad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``StoreScanTasks''}]$
$\qquad\qquad \wedge \text{UNCHANGED } \langle scan\_tasks,\ response,\ key\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad\qquad service\_request,\ number\_of\_batches,\ counter,\ stack,$
$\qquad\qquad\qquad\qquad\qquad\qquad keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys \rangle$

$Deleting \triangleq \wedge pc[\text{``SCAN TASK''}] = \text{``Deleting''}$
$\qquad\qquad\quad \wedge scan\_tasks' = scan\_tasks \setminus \{key\_to\_be\_served\}$
$\qquad\qquad\quad \wedge scan\_task\_status' = \text{``waiting''}$
$\qquad\qquad\quad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``StoreScanTasks''}]$
$\qquad\qquad\quad \wedge \text{UNCHANGED } \langle response,\ key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ stack,\ keys\_,\ keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,\ inner\_state \rangle$

$StoreScanTasks \triangleq \wedge pc[\text{``SCAN TASK''}] = \text{``StoreScanTasks''}$
$\qquad\qquad\qquad\quad \wedge scan\_tasks' = inner\_state$
$\qquad\qquad\qquad\quad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``ScanTaskLoop''}]$
$\qquad\qquad\qquad\quad \wedge \text{UNCHANGED } \langle response,\ scan\_task\_status,\ key\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad service\_request,\ number\_of\_batches,\ counter,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad stack,\ keys\_,\ keys\_g,\ keys\_c,\ keys\_ge,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_r,\ keys\_d,\ keys,\ inner\_state \rangle$

$ScanTaskLoop \triangleq \wedge pc[\text{``SCAN TASK''}] = \text{``ScanTaskLoop''}$
$\qquad\qquad\qquad \wedge pc' = [pc \text{ EXCEPT } ![\text{``SCAN TASK''}] = \text{``ScanTask''}]$
$\qquad\qquad\qquad \wedge \text{UNCHANGED } \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches,\ counter,\ stack,\ keys\_,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$

17

$$inner\_state\rangle$$

$scantask \triangleq GetScanTasks \lor ScanTask \lor BoundError \lor Adding \lor Deleting$
$\qquad\qquad \lor StoreScanTasks \lor ScanTaskLoop$

$ConfigGuard \triangleq \land pc[\text{``MAIN''}] = \text{``ConfigGuard''}$
$\qquad\qquad \land \text{IF } ConfigViewingKeys \neq \{\}$
$\qquad\qquad\qquad \text{THEN } \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``FromZebradConfig''}]$
$\qquad\qquad\qquad \text{ELSE } \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``ListeningGuard''}]$
$\qquad\qquad \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad number\_of\_batches, counter, stack, keys\_,$
$\qquad\qquad\qquad\qquad\qquad keys\_g, keys\_c, keys\_ge, keys\_r, keys\_d, keys,$
$\qquad\qquad\qquad\qquad\qquad inner\_state\rangle$

$FromZebradConfig \triangleq \land pc[\text{``MAIN''}] = \text{``FromZebradConfig''}$
$\qquad\qquad \land \land keys\_' = [keys\_ \text{ EXCEPT } ![\text{``MAIN''}] = ConfigViewingKeys]$
$\qquad\qquad\quad \land stack' = [stack \text{ EXCEPT } ![\text{``MAIN''}] = \langle[procedure \mapsto \text{``add\_config\_keys''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad pc \qquad\quad \mapsto \text{``ListeningGuard''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_ \quad\;\; \mapsto keys\_[\text{``MAIN''}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \circ stack[\text{``MAIN''}]]$
$\qquad\qquad \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``AddConfigKeys''}]$
$\qquad\qquad \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad number\_of\_batches, counter, keys\_g, keys\_c,$
$\qquad\qquad\qquad\qquad\qquad keys\_ge, keys\_r, keys\_d, keys, inner\_state\rangle$

$ListeningGuard \triangleq \land pc[\text{``MAIN''}] = \text{``ListeningGuard''}$
$\qquad\qquad \land \text{IF } GrpcViewingKeys \neq \langle\rangle$
$\qquad\qquad\qquad \text{THEN } \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``GetTotalIterationsToMake''}]$
$\qquad\qquad\qquad \text{ELSE } \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``End''}]$
$\qquad\qquad \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad number\_of\_batches, counter, stack, keys\_,$
$\qquad\qquad\qquad\qquad\qquad keys\_g, keys\_c, keys\_ge, keys\_r, keys\_d,$
$\qquad\qquad\qquad\qquad\qquad keys, inner\_state\rangle$

$GetTotalIterationsToMake \triangleq \land pc[\text{``MAIN''}] = \text{``GetTotalIterationsToMake''}$
$\qquad\qquad \land stack' = [stack \text{ EXCEPT } ![\text{``MAIN''}] = \langle[procedure \mapsto \text{``get\_config\_number.}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad pc \qquad\quad \mapsto \text{``ListeningMode''}]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ stack[\text{``MAIN''}]]$
$\qquad\qquad \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``CheckBatch1''}]$
$\qquad\qquad \land \text{UNCHANGED } \langle scan\_tasks, response,$
$\qquad\qquad\qquad\qquad\qquad scan\_task\_status, key\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad service\_request, number\_of\_batches,$
$\qquad\qquad\qquad\qquad\qquad counter, keys\_, keys\_g, keys\_c,$

$$\langle keys\_ge,\ keys\_r,\ keys\_d,\ keys,$$
$$inner\_state\rangle$$

$ListeningMode \triangleq\ \wedge\ pc[\text{"MAIN"}] = \text{"ListeningMode"}$
$\qquad\qquad\quad \wedge\ \text{IF}\ counter\ \leq number\_of\_batches$
$\qquad\qquad\qquad\quad \text{THEN}\ \ \wedge\ pc' = [pc\ \text{EXCEPT}\ ![\text{"MAIN"}] = \text{"GetInfoCall"}]$
$\qquad\qquad\qquad\quad \text{ELSE}\ \ \wedge\ pc' = [pc\ \text{EXCEPT}\ ![\text{"MAIN"}] = \text{"End"}]$
$\qquad\qquad\quad \wedge\ \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ stack,\ keys\_,$
$\qquad\qquad\qquad\qquad\qquad\qquad keys\_g,\ keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
$\qquad\qquad\qquad\qquad\qquad\qquad inner\_state\rangle$

$GetInfoCall \triangleq\ \wedge\ pc[\text{"MAIN"}] = \text{"GetInfoCall"}$
$\qquad\qquad\quad \wedge\ stack' = [stack\ \text{EXCEPT}\ ![\text{"MAIN"}] = \langle[procedure \mapsto\ \text{"get\_info"},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad pc\qquad\quad \mapsto\ \text{"RegisterKeysCall"}]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ stack[\text{"MAIN"}]]$
$\qquad\qquad\quad \wedge\ pc' = [pc\ \text{EXCEPT}\ ![\text{"MAIN"}] = \text{"InfoServiceRequest"}]$
$\qquad\qquad\quad \wedge\ \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ keys\_,\ keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad keys\_c,\ keys\_ge,\ keys\_r,\ keys\_d,\ keys,$
$\qquad\qquad\qquad\qquad\qquad\qquad inner\_state\rangle$

$RegisterKeysCall \triangleq\ \wedge\ pc[\text{"MAIN"}] = \text{"RegisterKeysCall"}$
$\qquad\qquad\qquad \wedge\ \wedge\ keys\_r'\ = [keys\_r\ \text{EXCEPT}\ ![\text{"MAIN"}] = GrpcViewingKeys[counter]]$
$\qquad\qquad\qquad\qquad \wedge\ stack' = [stack\ \text{EXCEPT}\ ![\text{"MAIN"}] = \langle[procedure \mapsto\ \text{"register\_keys"},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad pc\qquad\quad \mapsto\ \text{"GetStatusCall"},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_r\quad \mapsto\ keys\_r[\text{"MAIN"}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \circ stack[\text{"MAIN"}]]$
$\qquad\qquad\qquad \wedge\ pc' = [pc\ \text{EXCEPT}\ ![\text{"MAIN"}] = \text{"RegisterServiceRequest"}]$
$\qquad\qquad\qquad \wedge\ \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches,\ counter,\ keys\_,\ keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_c,\ keys\_ge,\ keys\_d,\ keys,\ inner\_state\rangle$

$GetStatusCall \triangleq\ \wedge\ pc[\text{"MAIN"}] = \text{"GetStatusCall"}$
$\qquad\qquad\quad \wedge\ \wedge\ keys\_ge' = [keys\_ge\ \text{EXCEPT}\ ![\text{"MAIN"}] = GrpcViewingKeys[counter]]$
$\qquad\qquad\qquad \wedge\ stack' = [stack\ \text{EXCEPT}\ ![\text{"MAIN"}] = \langle[procedure \mapsto\ \text{"get\_status"},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad pc\qquad\quad \mapsto\ \text{"GetResultsCall"},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_ge\quad \mapsto\ keys\_ge[\text{"MAIN"}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \circ stack[\text{"MAIN"}]]$
$\qquad\qquad\quad \wedge\ pc' = [pc\ \text{EXCEPT}\ ![\text{"MAIN"}] = \text{"StatusServiceRequest"}]$
$\qquad\qquad\quad \wedge\ \text{UNCHANGED}\ \langle scan\_tasks,\ response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches,\ counter,\ keys\_,\ keys\_g,$

$$\langle keys\_c, keys\_r, keys\_d, keys, inner\_state \rangle$$

$GetResultsCall \triangleq \land pc[\text{``MAIN''}] = \text{``GetResultsCall''}$
$\qquad\qquad\qquad \land \land keys\_g' = [keys\_g \text{ EXCEPT } ![\text{``MAIN''}] = GrpcViewingKeys[counter]]$
$\qquad\qquad\qquad\quad\ \land stack' = [stack \text{ EXCEPT } ![\text{``MAIN''}] = \langle[procedure \mapsto \text{``get\_results''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad pc \qquad \mapsto \text{``ClearResultsCall''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_g \quad \mapsto keys\_g[\text{``MAIN''}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ stack[\text{``MAIN''}]]$
$\qquad\qquad\qquad \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``ResultsServiceRequest''}]$
$\qquad\qquad\qquad \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches, counter, keys\_, keys\_c,$
$\qquad\qquad\qquad\qquad\qquad\qquad keys\_ge, keys\_r, keys\_d, keys, inner\_state \rangle$

$ClearResultsCall \triangleq \land pc[\text{``MAIN''}] = \text{``ClearResultsCall''}$
$\qquad\qquad\qquad\quad \land \land keys\_c' = [keys\_c \text{ EXCEPT } ![\text{``MAIN''}] = GrpcViewingKeys[counter]]$
$\qquad\qquad\qquad\qquad\ \land stack' = [stack \text{ EXCEPT } ![\text{``MAIN''}] = \langle[procedure \mapsto \text{``clear\_results''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad pc \qquad \mapsto \text{``DeleteKeysCall''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_c \quad \mapsto keys\_c[\text{``MAIN''}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ stack[\text{``MAIN''}]]$
$\qquad\qquad\qquad\quad \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``ClearServiceRequest''}]$
$\qquad\qquad\qquad\quad \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches, counter, keys\_, keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_ge, keys\_r, keys\_d, keys, inner\_state \rangle$

$DeleteKeysCall \triangleq \land pc[\text{``MAIN''}] = \text{``DeleteKeysCall''}$
$\qquad\qquad\qquad\quad \land \land keys\_d' = [keys\_d \text{ EXCEPT } ![\text{``MAIN''}] = GrpcViewingKeys[counter]]$
$\qquad\qquad\qquad\qquad\ \land stack' = [stack \text{ EXCEPT } ![\text{``MAIN''}] = \langle[procedure \mapsto \text{``delete\_keys''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad pc \qquad \mapsto \text{``ScanCall''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_d \quad \mapsto keys\_d[\text{``MAIN''}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ stack[\text{``MAIN''}]]$
$\qquad\qquad\qquad\quad \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``DeleteServiceRequest''}]$
$\qquad\qquad\qquad\quad \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad number\_of\_batches, counter, keys\_, keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad keys\_c, keys\_ge, keys\_r, keys, inner\_state \rangle$

$ScanCall \triangleq \land pc[\text{``MAIN''}] = \text{``ScanCall''}$
$\qquad\qquad\ \land \land keys' = [keys \text{ EXCEPT } ![\text{``MAIN''}] = GrpcViewingKeys[counter]]$
$\qquad\qquad\quad\ \land stack' = [stack \text{ EXCEPT } ![\text{``MAIN''}] = \langle[procedure \mapsto \text{``scan''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad pc \qquad \mapsto \text{``IncrementCounter''},$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys \qquad \mapsto keys[\text{``MAIN''}]]\rangle$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \circ stack[\text{``MAIN''}]]$
$\qquad\qquad\ \land pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``RegisterServiceRequestFromScan''}]$
$\qquad\qquad\ \land \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$

$$\langle key\_to\_be\_served, service\_request,$$
$$number\_of\_batches, counter, keys\_, keys\_g, keys\_c,$$
$$keys\_ge, keys\_r, keys\_d, inner\_state\rangle$$

$IncrementCounter \triangleq\ \wedge pc[\text{``MAIN''}] = \text{``IncrementCounter''}$
$\qquad\qquad\qquad\ \wedge counter' = counter + 1$
$\qquad\qquad\qquad\ \wedge pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``ListeningMode''}]$
$\qquad\qquad\qquad\ \wedge \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served, service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad number\_of\_batches, stack, keys\_, keys\_g,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad keys\_c, keys\_ge, keys\_r, keys\_d, keys,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad inner\_state\rangle$

$End \triangleq\ \wedge pc[\text{``MAIN''}] = \text{``End''}$
$\qquad\quad\ \wedge \text{TRUE}$
$\qquad\quad\ \wedge pc' = [pc \text{ EXCEPT } ![\text{``MAIN''}] = \text{``Done''}]$
$\qquad\quad\ \wedge \text{UNCHANGED } \langle scan\_tasks, response, scan\_task\_status,$
$\qquad\qquad\qquad\qquad key\_to\_be\_served, service\_request, number\_of\_batches,$
$\qquad\qquad\qquad\qquad counter, stack, keys\_, keys\_g, keys\_c, keys\_ge, keys\_r,$
$\qquad\qquad\qquad\qquad keys\_d, keys, inner\_state\rangle$

$Main \triangleq\ ConfigGuard \vee FromZebradConfig \vee ListeningGuard$
$\qquad\quad\ \vee GetTotalIterationsToMake \vee ListeningMode \vee GetInfoCall$
$\qquad\quad\ \vee RegisterKeysCall\ \vee GetStatusCall \vee GetResultsCall$
$\qquad\quad\ \vee ClearResultsCall\ \vee DeleteKeysCall \vee ScanCall$
$\qquad\quad\ \vee IncrementCounter \vee End$

Allow infinite stuttering to prevent deadlock on termination.
$Terminating \triangleq\ \wedge \forall\, self \in ProcSet : pc[self] = \text{``Done''}$
$\qquad\qquad\qquad\quad \wedge \text{UNCHANGED } vars$

$Next \triangleq\ services \vee scantask \vee Main$
$\qquad\quad\ \vee (\exists\, self \in ProcSet :\ \vee get\_config\_number\_of\_batches(self)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vee add\_config\_keys(self) \vee get\_info(self)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vee get\_results(self) \vee clear\_results(self)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vee get\_status(self) \vee register\_keys(self)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vee delete\_keys(self) \vee scan(self))$
$\qquad\qquad\ \vee Terminating$

$Spec \triangleq\ \wedge Init \wedge \Box[Next]_{vars}$
$\qquad\quad\ \wedge \text{WF}_{vars}(scantask)$

$Termination \triangleq\ \Diamond(\forall\, self \in ProcSet : pc[self] = \text{``Done''})$

END TRANSLATION

\ * *Modification History*

\ ∗ *Last modified Thu Mar* 07 18 : 13 : 46 *UYT* 2024 *by alfredo*
\ ∗ *Created Wed Feb* 21 10 : 40 : 53 *UYT* 2024 *by alfredo*