```
  Cache incorporates the data
CacheCompleteFill(k) ≜
    ∧ cacheFillStates[k].state = "respondedto"
      Either the cache is empty for that key
    ∧ ∨ cache[k] ∈ CacheMiss
      or we are filling a newer version
      ∨ ∧ cache[k] ∉ CacheMiss
        ∧ cache[k].version < cacheFillStates[k].version
    ∧ cacheFillStates' = [cacheFillStates EXCEPT   Reset to 0
                            ![k].state = "inactive",
                            ![k].version = 0
                          ]
    ∧ cache' = [cache EXCEPT
                  ![k] = [
                    cache value is now a hit
                    type ↦ "hit",
                    set to whatever came back in response
                    version ↦ cacheFillStates[k].version
                  ]
                ]
    ∧ UNCHANGED ⟨database, invalidationQueue⟩

CacheIgnoreFill(k) ≜
    ∧ cacheFillStates[k].state = "respondedto"
    If we have a newer version in cache, ignore fill
    ∧ ∧ cache[k] ∈ CacheHit
      ∧ cache[k].version ≥ cacheFillStates[k].version
    ∧ cacheFillStates' = [cacheFillStates EXCEPT   Reset to 0
                            ![k].state = "inactive",
                            ![k].version = 0
                          ]
    Don't update cache
    ∧ UNCHANGED ⟨cache, database, invalidationQueue⟩

Handle invalidation message. Assume it is not taken off queue in case of
failure. Therefore failure modeled as CacheHandleInvalidationMessage not
occuring
CacheHandleInvalidationMessage ≜
    ∧ ∃ message ∈ invalidationQueue :   Deque invalidation queue in any order
        Key must be in cache
      ∧ ∧ cache[message.key] ∈ CacheHit
        Message needs to be newer then the cache
        ∧ cache[message.key].version < message.version
      Update item in cache
```

```
        ∧ cache' = [cache EXCEPT
                        ![message.key] = [
                            type ↦ "hit",
                            Update to version in invalidation message
                            version ↦ message.version
                        ]]
        Remove message from queue because handled
        ∧ invalidationQueue' = invalidationQueue \ {message}

    ∧ UNCHANGED ⟨cacheFillStates, database⟩

CacheIgnoreInvalidationMessage ≜
    ∧ ∃ message ∈ invalidationQueue :  Deque invalidation queue in any order
        Ignore invalidation messages for messages not in cache
        ∧ ∨ cache[message.key] ∈ CacheMiss
          Or when the cache already has the same or larger version
          ∨ ∧ cache[message.key] ∉ CacheMiss
            ∧ cache[message.key].version ≥ message.version
        Remove message from queue to ignore
        ∧ invalidationQueue' = invalidationQueue \ {message}
    Don't update cache
    ∧ UNCHANGED ⟨cacheFillStates, database, cache⟩

\ * Modification History
\ * Last modified Wed Jun 15 13:58:25 MST 2022 by elliotswart
\ * Created Wed Jun 15 13:58:13 MST 2022 by elliotswart
```