Beispielszenario für die Modellierung von Räumen

**Beschrieb der Situation:**

Der Spieler betritt Raum A (Kontrollraum). Raum A ist dunkel, dennoch sind (an der Wand) drei Schalter zu erkennen:

1. Ein Schalter um in A die Beleuchtung anzuschalten
2. Ein Schalter um auf allen Ebenen, die Stromversorgung ein- und auszuschalten
3. Ein Schalter um in Raum B die Stromversorgung selektiv zu de-/aktivieren

**Raum A:** **Raum B:**

**Umgebung:** **Umgebung:**

Bei Eintritt dunkel, nach Betätigung des Bei Eintritt dunkel

Schalters A beleuchtet

**Gegenstände: Gegenstände:**

- Schalter 1, Schalter 2, Schalter 3

- Lichtschalter - Tür als Ein- und Ausgang

- Lampe für die Beleuchtung - Lichtschalter

- Tür als Ein- und Ausgang

**Handlungsstrang:**

Initialsituation:

**RoomA *.globalState.getPowerState-> onEmergency***

**; //deshalb hat man Zutritt**

**RoomA *.currentState = [Accessible, NonVisited];***

**RoomB *.globalState.getPowerState-> powerless***

**; //deshalb *NonAccessible***

**RoomB*.currentState = [NonAccessible, NonVisited];***

Handlung:

**Der Spieler betätigt Schalter 1, welcher die Beleuchtung in A aktiviert.**

Die Betätigung des Schalters wird getriggert und die Attribute des Raumes resp. der Items im Raum verändert:

RoomA.*switch1 = true;* RoomA.*lamp = true;*

Nach Betätigung des Schalters 1 wird Raum A durch die Beleuchtung erhellt.

**Der Spieler betätigt den Schalter 2, welcher auf dem ganzen Level die Stromzufuhr einschaltet.**

Objekt Schalter2:  
boolean switch: on

if(pushed && playerinrange)

send Msg an Verteilerliste: *DL\_powerConsumer*.

In dieser Liste befinden sich alle Komponenten / Entities, die auf eine konstante Stromversorgung angewiesen sind, also nicht mit Batterie laufen.

Diese Entities, die sich beim Instanzieren in *die DL\_powerConsumer*-Liste eingetragen haben, erhalten die Nachricht GOT\_POWER. Sie akzeptieren diese und leiten Sie an die StateMachine weiter.

Die StateMachines, erhalten diese Message und geben sie an den GlobalState weiter, der diese als Erster handelt. Bei uns managt der GlobalState Dinge wie „Stromunterbruch“, „Sprinkleranlage an“ etc. Die onMessage() Methode des GlobalState – hier am Beispiel des bisher stromlosen Raum B’s - liefert somit ein *true* zurück.

onMessage() {

if(powerON) {

switch(hasStateX) {

case0: ..

case1: ..

default: FSM.saveCurrentState(), FSM.changeState(**accessible**) }

}

}

RoomB.*globalState.setPowerState() -> onPower.*

Raum B war deshalb inaccessible, weil kein Strom zur Verfügung stand.

Durch das Handling der Message durch den GlobalState, ist der Raum nun zugänglich -> Siehe Code als nächstes. Falls der Stromunterbruch während dem State [Accessible, Visited] geschah und der Raum auf diese Weise auf inaccessible geschaltet wurde, wird zurück zu diesem Zustand gewechselt der als previousState bekannt ist

Der inaccessible-State nimmt die Message entgegen und führt nun (aufgerufen durch das Update der State-Machine) die execute Methode aus, in der geprüft wird  
(triggering), ob Strom vorhanden ist. Wenn ja – und wenn ev. noch andere Bedingungen erfüllt werden, wechselt er in den entsprechenden Status:  
if(parentRoom.*globalState.getPowerState()==onPower && door is unlocked) {*

*SM.changeState(Accessible);*

*}*

Falls Raum B vor dem Stromzusammenbruch einen eingeschalteten Lichtschalter besass, ist das *lampSwitch Attribut des Raumes* true & wenn wir davon ausgehen, dass die Lampe zum einen auf den Switch prüft und zum anderen entweder:

1. ebenfalls auf den GlobalState zugreift, um den Powerstand abzufragen oder
2. als Empfänger der der Message GOT\_POWER über den wieder erlangten Saft informiert wird.

Dann sollte sie wieder zu leuchten beginnen. Das Handling der Items ist noch Gegenstand der Diskussion. Grundsätzlich finden wir es sinnvoll, wenn alle elektrischen betriebenen, vom Strom abhängigen Items in der DL\_powerConsumer Liste sind. Dann könnte man auch von der Abhängigkeit des GlobalStates wegkommen.

**Der Spieler betätigt Schalter 3, mit dem in Raum B, einzeln der Strom ein- und ausgeschaltet wird.**

Strom wird ausgeschaltet (im Moment ON) und an Raum B wird die Message: NO\_POWER geschickt.

Die Entity RaumB gibt dies weiter an die im Raum vorhandenen Entities wie Lampe, Maschine1, etc. (oder eben alternativ sind alle Geräte Empfänger der Verteilerliste).

Die eigene State-Machine, wird vom GlobalState veranlasst, den Zustand anzupassen. Der aktuelle Status „Accessible“ wird verlassen, auf die gleiche Weise wie er vorhin eintrat und es wird in den Inaccessible Zustand gewechselt.

Beim Wechsel wird der PreviousState abgespeichert, damit man weiss, ob der Raum besucht wurde, oder nicht.