Lock-by-Wire Door Lock Requirements and Specification

Steven Lawrance
Version 1.5
November 14, 2006

Document Revisions

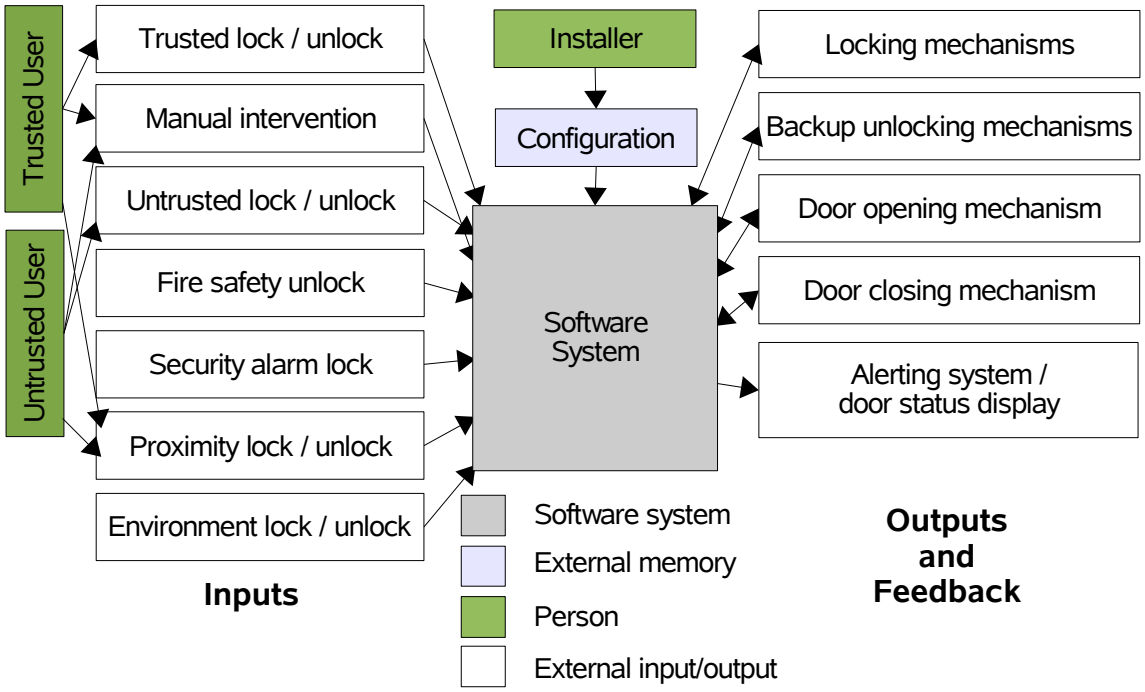
Revision	Changes
1.0	Initial draft
1.5	Added revision section, included more quality attributes, clarified actors in the use cases, added fault tolerance information, added another view of the timing information, added more use cases, made the use case names concise, and added more diagrams for illustration

Table of Contents

Context

Traditional pervasive door locks found in buildings ranging from residences to businesses typically combine a knob or handle with a separate locking mechanism. That lock ranges widely from dead-bolts to simple chains, and rotations differ between models and installations. Sometimes, security or safety are compromised due to confusion on how to properly operate the lock, defeating its purpose. To address accidental lapses in security or safety, this document proposes the software requirements of a new type of door lock that performs its operations electronically while keeping the user interface as simple as possible.

In airplanes and, more recently, in automobiles, placing a computer system between the controls and the plant permits that computer to promote several quality attributes including safety, reliability, optimization, security, and modifiability. This "lock-by-wire" door lock system seeks to achieve a similar level of integration. Instead of standing on its own, this door lock system can use inputs from other systems such as security, fire detection, and the environment. Remote opening and closing become possible as well as proximity-based detection of authorized persons.



Drawing 1: Context diagram

As depicted in the context diagram above, the software will take inputs from multiple sources to drive the desired states. The desired states will drive physical mechanisms, and sensors from those physical mechanisms feed input back into the software as the actual states. The system's current status can be displayed on a notification output along with any alerts that the user should know about. When necessary, the software can activate backup unlocking mechanisms to force the door to unlock in the event of a primary locking mechanism's failure to unlock.

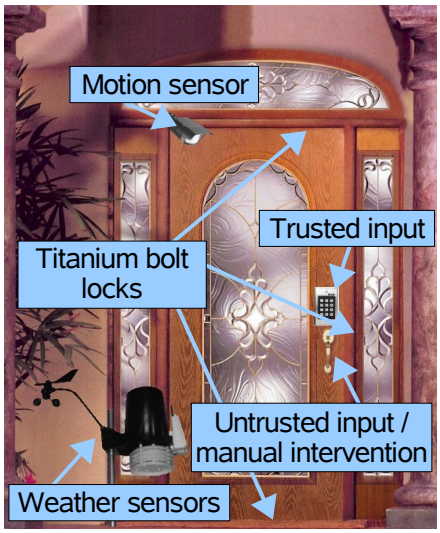
Each output device that controls a motion actuator will have at least two associated states: a desired state for that output device and the actual state for that input device. Differences between these two states over time can indicate a failure in the output device.

The configuration storage can be implemented as memory, I/O, disk storage, or any other type of device that can persist state.

The system generally classifies people into three different categories: trusted users, untrusted users, and installers. Trusted users are typically either authenticated and authorized or simply on the trusted side of the door, such as the inside of a house's front door. Untrusted users are those who are not authorized to affect the doors state when the door is locked. This models the outside of a residential door or the inside of a prison cell door. Installers are people who can change the configuration. The mechanism that is used to change the configuration is beyond the scope of this requirements document and is assumed to be implemented by an external system that provides the configuration to this system.

The diversity of input types permits versatility in the door's configuration. Not all inputs and outputs are required for the door's proper operation, though the software will need to be aware of which inputs and outputs are connected. Examples of each input type appear below:

- **Trusted lock / unlock:** User turns the inside door knob or types in the correct code onto a numeric keypad. User authentication and authorization is outside the scope of the lock software, though the output of those steps can be used to determine if an input is trusted or not for the purpose of the lock software.
- **Untrusted lock / unlock:** User attempts to open the door from the outside by turning the handle while the door is locked. As another example, if a door was told to be unlocked by a trusted user, then a locking attempt by an untrusted user can be denied.
- **Manual intervention:** User is manually turning a door knob or is pushing or pulling on the door.
- **Fire safety unlock:** Fire alarm system detected a fire, requiring all occupants to leave the building immediately.
- **Security alarm lock:** Security alarm detected an intruder and advises the appropriate doors to close and lock themselves.
- **Proximity lock / unlock:** A user walks near the door, which gets sensed by motion sensors. This can unlock and open a door, though some situations might exist where motion should prompt a closure and locking of the door.
- **Environment lock / unlock:** The temperature outside equals the desired inside room temperature, making it advantageous to unlock and open the door. If cleaning or other chemicals were used on the inside of a building and the fumes need to be removed, opening the door can also help. Similarly, if the weather is raining, hailing, snowing, or windy, the door should probably be closed and locked.



Drawing 2: A possible configuration

Examples of outputs that feed back their actual statuses into the system include the following:

- **Locking mechanisms:** When engaged, the locking mechanisms will lock the door in place such that it cannot move. In most configurations, the locks can engage only when the door is closed, but this is not universally true.
- **Backup unlocking mechanisms:** When engaged, the backup unlocking mechanism will override a failed locking mechanism to unlock the door. This is used in emergency situations.
- **Door opening mechanism:** When engaged, the door will open.
- **Door closing mechanism:** When engaged, the door will close.

To inform the user of the current state, alerts and notifications are used. These can be audible, visual, or both, ranging from simple horns and light emitting diodes (LEDs) to immersive flat-panel liquid crystal displays (LCDs) and in-wall surround sound speakers.

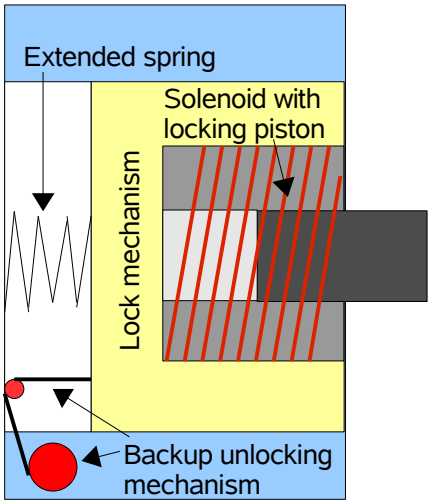
To coordinate these pieces together, a configuration will tell the software which inputs to use and which to ignore.

With the context established, it's possible to delve into the high-level requirements of the locking software, which appears in the next section.

High-Level Requirements

The software component of this system has the following high-level requirements, sorted by decreasing priority:

- **R1: State change when desired states do not equal the actual states:** The system shall attempt to change the states of connected mechanisms to match desired states when the actual states differ. The desired states are determined by the software after processing the inputs.
- **R2: State awareness:** The system shall know the present states of the door as well as what they should be. When these states differ, the system shall direct the appropriate outputs to change so that the actual states will match the desired states.
- **R3: Prioritize trusted actions over other inputs:** The system shall prioritize trusted actions over other inputs when determining the desired state.
- **R4: Inputs separated by type:** The system shall accept inputs from systems differentiated by quality type. Input types can include fire safety, security, trusted, untrusted, environmental, and proximity.
- **R5: Input signal prioritization to determine desired state:** The prioritized list shall be the following in decreasing order of consideration: fire safety, security alarm, proximity, environment, and untrusted user actions.
- **R6: Manual intervention:** When the manual intervention input is set from a trusted source, such as an inside door knob, the door lock will follow the desired action of the user.
- **R7: User alerts:** The system shall alert the user of any problems that the door encounters in performing its operations as well as any state that could benefit from user awareness. This can be done audibly, visually, or both.
- **R8: Backup unlocking mechanism for emergencies:** If a locking mechanism fails to unlock the door, the user will be alerted and be given a chance to engage the backup unlocking mechanisms. The device depicted in Drawing 3 is an example of a reusable backup unlocking mechanism that uses a motor to pull the main locking mechanism into the door against the force of the extended spring.



Drawing 3: Example locking assembly with backup unlocking mechanism

- **R9: Door status visibility:** The system shall let the user know what the operational state of the door is, such as locked, unlocked, opened for weather, or similar states.
- **R10: Configuration memory:** The system shall read a configuration that tells it which inputs, outputs, and behaviors to use in its processing.

Functional Requirement Interdependencies

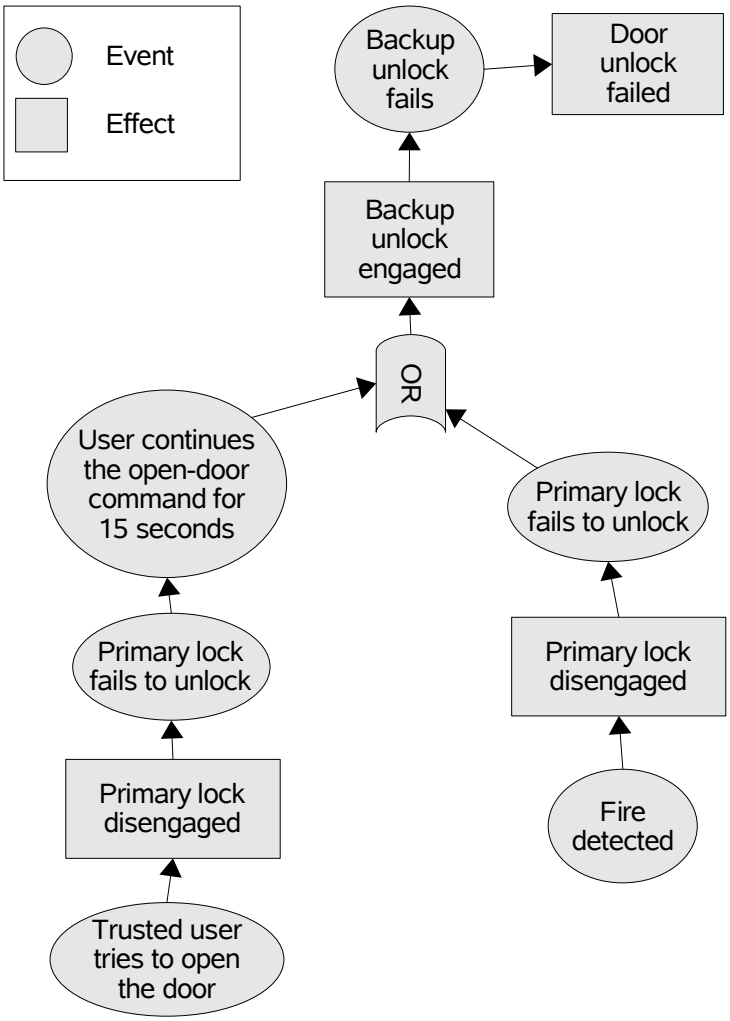
The functional requirements listed in the previous section depend on other functional requirements to be implemented properly:

Functional Requirement	Dependencies
R2: State awareness	• R5: Input signal prioritization to determine desired state
R1: State change when desired states do not equal the actual states	• R2: State awareness • R8: Backup unlocking mechanism for emergencies • R7: User alerts
R4: Inputs separated by type	• R10: Configuration memory
R6: Manual intervention	• R1: State change when desired states do not equal the actual states
R8: Backup unlocking mechanism for emergencies	• R7: User alerts • R6: Manual intervention
R5: Input signal prioritization to determine desired state	• R4: Inputs separated by type • R10: Configuration memory
R10: Configuration memory	
R7: User alerts	
R3: Prioritize trusted actions over other inputs	• R5: Input signal prioritization to determine desired state
R9: Door status visibility	

Note that if one starts with the “manual intervention” functional requirement, the dependencies will cause all other functional requirements to become relevant. Similarly, if one starts with the “state change when desired states do not equal the actual states” functional requirement, it will depend on the other functional requirements, too. These requirements work together to operate the door’s locking and opening/closing mechanisms.

Fault Tolerance

This system permits fault tolerance in its unlocking operation. A door can have backup unlocking mechanisms to complement the normal unlocking mechanisms, permitting the door to open in the event of an unlock fault on the primary locking mechanism. The following fault tree depicts how the backup unlocking mechanism can engage.



Quality Attributes

This software system has the following quality attributes ordered by decreasing priority:

- **QA1: Safety:** When the fire safety system input indicates a fire, the door shall disengage the locking mechanism within 50ms of receiving the input signal. If an error prevents the door from unlocking, which is detected within or at the end of 3 seconds, then the backup door unlocking mechanism must be actuated within 50ms of determining that the main locking mechanism failed.
- **QA2: Responsiveness:** When an input signal arrives, the system shall recognize and prioritize it within 10ms. Any operation that was being performed is preempted if its priority is lower than the new input signal.
- **QA3: Reliability:** The system shall include a backup unlocking mechanism that can begin to mechanically override a failed locking mechanism within 50ms of a primary locking mechanism failure. The backup unlocking mechanism can be a one-time use device, but does not need to be. The system shall detect a failure of a primary locking mechanism within or at the end of 3 seconds of its disengagement.
- **QA4: Security:** When the security alarm input indicates that the building should be locked down and a higher-priority input signal is not present, the door shall actuate a door closing mechanism within 50ms of receiving that input signal.
- **QA5: Configurability:** The software shall adapt to the configuration of the inputs, outputs, and behaviors. This controls which input lines to listen to, which output lines to drive, and which behaviors to exhibit.
- **QA6: Modifiability:** The software shall be updatable to address system defects or add new features by an authorized installer without having to physically deconstruct the door. Such updates should be able to take place in five minutes or less.

Timing Requirements

The following table summarizes the system's timing requirements in terms of requirements and timeouts, sorted by increasing order of deadline:

Requirement	Deadline
The system shall recognize and prioritize input signals within 10ms	10 milliseconds
The backup unlocking mechanism shall begin to be activated within 50ms of determining that the backup unlocking mechanism should be activated	50 milliseconds
When the security alarm input signal indicates that the building is on fire, then the door shall begin to disengage its primary locking mechanisms within 50ms	50 milliseconds
The primary locking mechanism shall change its state from locked to unlocked or vice-versa within 3 seconds by default unless if its configuration states different maximum timeout values	3 seconds
When the manual intervention signal is set, it will remain set for five seconds since the time that the signal becomes unset	5 seconds
When a trusted user is trying to open the door and the system determines that the primary locking mechanisms failed, then the system will wait for 15 seconds before it engages the backup unlocking mechanisms	15 seconds
Software updates can occur in five minutes or less	5 minutes

Constraints

The system has the following constraints:

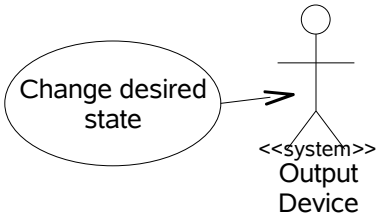
Constraint	Type	Impact	Flexibility
Cost minimization	Business	Limits the type of CPU as well as the memory size, number of input lines, and circuit complexity	Some, which can translate into different model types
Input line diversity	Technical	Having a large number of input lines to differentiate signals by priority narrows down which CPU types are acceptable as well as how complex the supporting circuits will be	Some, which can translate into different model types
Real-time scheduling requirements	Technical	The use of a real-time operating system or the creation of a similar system is necessary to ensure that the quality attribute measures and functional requirements are met	Not much, unless if safety or security are willing to be compromised
Physical output device response time variances	Physical	The response times of output devices can vary. This might require output response time information to be stored in the configuration for the system to know when an output device has failed	Some, which can be done by requiring specific output device model numbers, though variances might exist with that, too
Human interaction with the door	Physical	Because the door interacts with humans physically moving the door in some door opening and closing scenarios, the system must be able to respect trusted human wishes when the door wants to perform an action that differs from the human	None, unless if safety or security are willing to be compromised

Use Cases

The locking system has the following use cases, separated by high-level functional requirement.

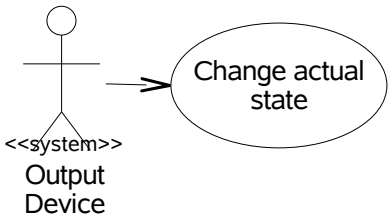
R1: State change when desired states do not equal the actual states

- **UC1.1: Actuate the output devices**
  - **Description:** When a desired state changes, drive the output devices to get the actual state to equal the new desired state
  - **Trigger:** The system is initialized or the desired state changes
  - **Precondition:** The system is online
  - **Actors:** Output device
  - **Flow of events:**
    1. Desired state changes or the system is initialized
    2. Output devices and alerts are changed in a manner that will lead to the actual state matching the desired state. The output device is signaled to change within 50ms of determining a new desired state that does not match the actual state
  - **Postcondition:** The output device knows that it needs to change its state
  - **Priority:** High



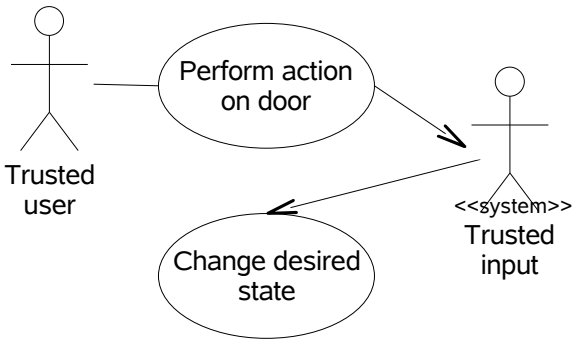
R2: State awareness

- **UC2.1: Read feedback from output devices**
  - **Description:** When the feedback input from an output device changes, update the known actual state to match the new reality
  - **Trigger:** One or more feedback inputs change or the system is initialized
  - **Precondition:** The system is online
  - **Actors:** Output device
  - **Flow of events:**
    1. Feedback inputs from the output devices are read
    2. Actual states are updated to match the feedback inputs
  - **Postcondition:** The actual states reflect the feedback inputs
  - **Priority:** High



R3: Prioritize trusted actions over other inputs

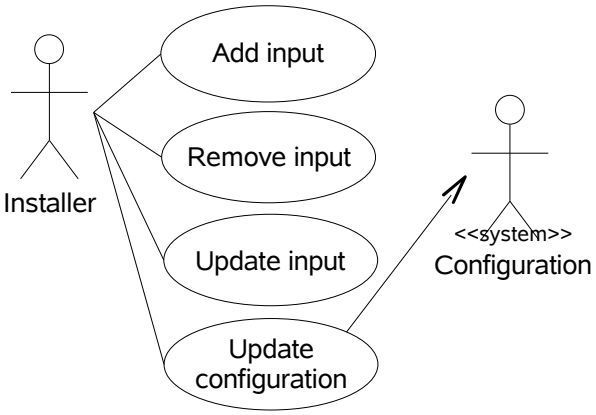
- **UC3.1: Prefer the actions of trusted users over other actions**
  - **Description:** When a trusted user sets a state such as the locked/unlock status, those state changes are prioritized above other actions. Examples include an untrusted user trying to open a locked door or a trusted user closing a door that was opened by an environmental sensor to let fresh air into the building.



  - **Trigger:** A trusted user performs an action on a door
  - **Precondition:** The system is online
  - **Actors:** Trusted user and trusted user input
  - **Flow of events:**
    1. The trusted user input is read by the software system
    2. The desired states are updated to match the trusted user input and is held in that state for either a preconfigured amount of time or indefinitely, depending on the configuration
  - **Postcondition:** The desired states reflect the trusted user input
  - **Priority:** High

R4: Inputs separated by type

- **UC4.1: Connect input devices and update the configuration**
  - **Description:** When the system is connected to its inputs, those inputs are connected to special lines that are differentiated by priority and type



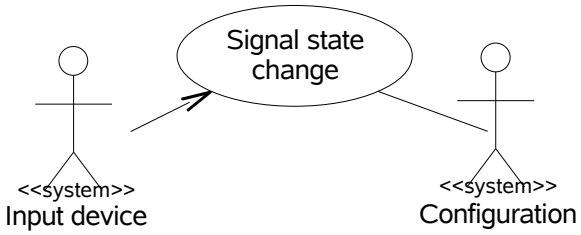
  - **Trigger:** Installation or update of a system's inputs
  - **Precondition:** The system is offline
  - **Actors:** Installer, configuration, and input devices
  - **Flow of events:**
    1. Installer adds, removes, or changes the connected inputs
    2. The installer updates the configuration to match the updated input line configuration

3. The installer turns the system on and tests it

- **Postcondition:** The door lock system is configured and operational with respect to its input signals
- **Priority:** Medium

R5: Input signal prioritization to determine desired state

- **UC5.1: Respond to changed inputs from input devices**
  - **Description:** When input signals change, the new desired state shall be computed, which depends on the input priorities
  - **Trigger:** One or more input signals change or the system is initialized
  - **Precondition:** System is online
  - **Actors:** Input device and configuration memory



  - **Flow of events:**
    1. Input device's signal changes
    2. If that signal is configured to be read, then proceed. Otherwise, don't proceed with the following steps
    3. New desired states are computed based on priorities. These states include the lock/unlock and door open/closed statuses. By default, the door will be locked and closed, though it will become unlocked and/or open if a condition sets those states
    4. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired states reflect the proper states given the changed inputs and priorities
  - **Priority:** High
- **UC5.1.1: Input signal prioritization**
  - **Description:** Input signals are prioritized
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online
  - **Actors:** Input device
  - **Flow of events:**
    1. Input device state is read in
    2. The input signals are prioritized using the following order of decreasing priorities: trusted inputs, fire safety, security alarm, proximity, environment, and untrusted inputs. The manual intervention input is considered separately as a modifier to the trusted and untrusted inputs

3. The desired states are updated to reflect the new desired states

- **Postcondition:** The desired states reflect the proper states given the changed inputs and priorities
- **Priority:** High

- **UC5.1.2: Trusted input changed**
  - **Description:** Trusted user, such as an inside user or an authorized outside user, opens the door electronically or physically, causing the trusted input to change
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online
  - **Actors:** Trusted user input device
  - **Flow of events:**
    1. The door's new desired locked/unlocked state is "unlocked"
    2. If the manual intervention input signal is not on, the door's new open/closed desired state is set to "open" if the input signal indicates that the door should be electronically opened, too
    3. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired locked/unlock state is "unlocked," and the desired open/closed state is "open" if and only if the manual intervention input signal is off
  - **Priority:** High
- **UC5.1.3: Fire safety input changed**
  - **Description:** The fire alarm indicates that the building is on fire, which requires that all doors be unlocked, causing the fire safety input to change
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online
  - **Actors:** Trusted user input device
  - **Flow of events:**
    1. The door's new desired locked/unlocked state is "unlocked"
    2. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired locked/unlock state is "unlocked"
  - **Priority:** High
- **UC5.1.4: Security alarm input changed**
  - **Description:** The security alarm is in its alarmed status, requiring that doors be closed and locked unless if unlocked and opened by a trusted user or the fire safety system. Note that in this state, the lower-priority inputs are ignored for inhabitant safety
  - **Trigger:** UC5.1: Respond to changed inputs from input devices

- **Precondition:** System is online
- **Actors:** Trusted user input device
- **Flow of events:**
  1. The door's new desired open/closed state is set to "closed"
  2. The door's new desired locked/unlocked state is set to "locked"
  3. The desired states are updated to reflect the new desired states
- **Postcondition:** The desired locked/unlock state is "locked," and the desired open/closed state is "closed"
- **Priority:** Medium

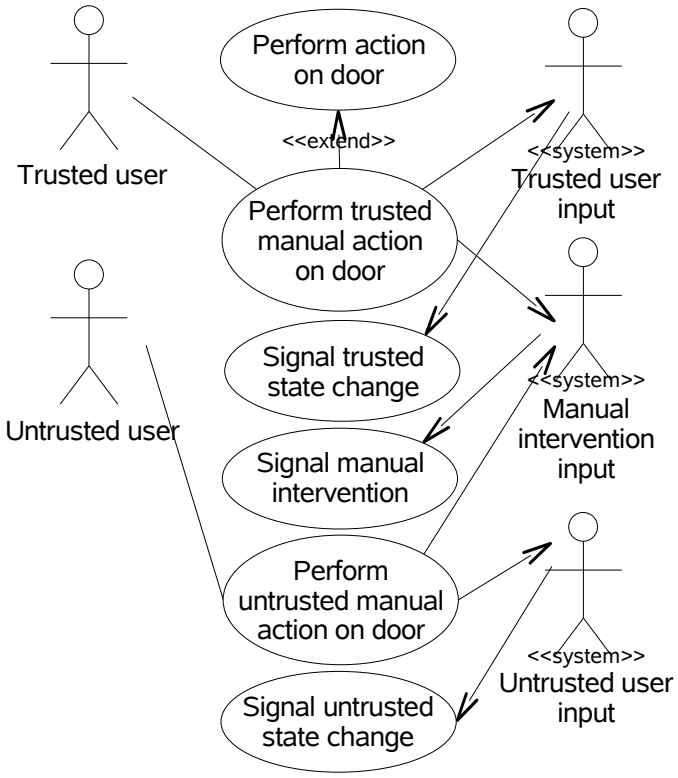
- **UC5.1.5: Trusted proximity detection input changed**
  - **Description:** A trusted motion sensor or similar device detects a person near the door. In this case, the door can be opened
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online and the actual open/closed state is "closed"
  - **Actors:** Trusted user input device
  - **Flow of events:**
    1. The door's new desired open/closed state is set to "open"
    2. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired open/closed state is "open"
  - **Priority:** Low
- **UC5.1.6: Untrusted proximity detection input changed on unlocked door**
  - **Description:** An untrusted motion sensor or similar device detects a person near the door. The door can be opened if the door was unlocked by a trusted user. It's also possible to have the proximity sensor close and lock the door when it detects person nearby on an untrusted motion sensor, which can help keep the building secure if the door was open only for weather-related purposes
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online, the actual locked/unlocked state is "unlocked," and the actual open/closed state is "closed"
  - **Actors:** Trusted user input device
  - **Flow of events:**
    1. The door's new desired open/closed state is set to "open"
    2. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired open/closed state is "open"
  - **Priority:** Low

- **UC5.1.7: Untrusted proximity detection input changed on a locked door**
  - **Description:** An untrusted motion sensor or similar device detects a person near the door. The door can be closed if the door was locked by a trusted user but is currently open for weather-related purposes
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online, the actual locked/unlocked state is "locked," and the actual open/closed state is "open"
  - **Actors:** Trusted user input device
  - **Flow of events:**
    1. The door's new desired open/closed state is set to "closed"
    2. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired open/closed state is "closed"
  - **Priority:** Low
- **UC5.1.8: Environmental opening of the door**
  - **Description:** If wind, temperature, and rainwater sensors detect that the outside weather is similar to the desired inside weather, then the door can optionally open. Note that the door can optionally close when it detects an untrusted person using a motion sensor
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online
  - **Actors:** Trusted user input device
  - **Flow of events:**
    1. The door's new desired open/closed state is set to "open"
    2. The desired states are updated to reflect the new desired states
  - **Postcondition:** The desired open/closed state is "open"
  - **Priority:** Low
- **UC5.1.9: Untrusted opening of the door**
  - **Description:** If an untrusted user performs an action, it is typically ignored, though it can be considered in some circumstances. An example is if the door was unlocked by a trusted user, the door is closed, and an untrusted user tries to open the door using a door-open button, such as a wheelchair access button. In that scenario, the door can be opened automatically
  - **Trigger:** UC5.1: Respond to changed inputs from input devices
  - **Precondition:** System is online and the door's actual locked/unlocked state is "unlocked"
  - **Actors:** Trusted user input device

- **Flow of events:**
  1. The door's new desired open/closed state is set to "open"
  2. The desired states are updated to reflect the new desired states
- **Postcondition:** The desired open/closed state is "open"
- **Priority:** Low

R6: Manual intervention

- **UC6.1: Manually controlling the door**
  - **Description:** User causes both the manual intervention input to turn on through a physical action on the door
  - **Trigger:** Pushing on the door, pressing a handle, or turning a knob
  - **Precondition:** The system is online
  - **Actors:** Trusted user or untrusted user, trusted or untrusted input, and the manual intervention input
  - **Flow of events:**
    1. User pushes on the door, presses a handle, or turns a knob on the door
    2. The manual intervention input is signaled, which causes the software system to leave that input signal on for at least five seconds
    3. After five seconds have passed since the last manual intervention, the manual intervention input turns off
  - **Postcondition:** The manual intervention input signal is off
  - **Priority:** High



R7: User alerts

- **R7.1: Actuate alerts from the software system**
  - **Description:** Alerts are displayed, audibly presented to the user, or both
  - **Trigger:** An alert is signaled by the software system
  - **Precondition:** The system is online

- **Actors:** Software system and the alert system
- **Flow of events:**
  1. Software system enters a state that requires attention from the user
  2. Display is updated to show the error message that the caller wants to display if not blank or null. Simultaneously, an audible alert is sounded at the desired frequency and on/off period if not zero. Note that the display and buzzer are optional, but at least one form of alerting mechanism must exist
     1. Making the display message blank and the audible alert frequency zero effectively turns off the alert
- **Postcondition:** The error message and/or audible alert is activated or deactivated, depending on what the input values were
- **Priority:** High

R8: Backup unlocking mechanism for emergencies

- **UC8.1: Emergency unlock during a fire when the primary lock does not unlock**
  - **Description:** The fire alarm input is on and the normal locking mechanism failed to disengage. The normal locking mechanism was determined to have failed within or at the end of three seconds since the time the locking mechanism was told to disengage
  - **Trigger:** The fire alarm input is on and the normal locking mechanism failed to disengage
  - **Precondition:** The system is online, the fire alarm input is on, and the normal locking mechanism failed to disengage
  - **Actors:** Fire alarm input, backup unlocking mechanism, and the alert system
  - **Flow of events:**
    1. Alert is activated to inform people nearby of the unlock difficulties
    2. Backup unlocking mechanism is engaged
    3. Backup unlocking status is checked every 500ms to see if it succeeded. Once it succeeds, it updates the alert to indicate the door's unlocked status
    4. If more than 5 seconds have passed since the door's backup unlocking mechanism was engaged, then the alert status is updated to indicate that the door failed as well as indicate which components failed
  - **Postcondition:** The door is unlocked if the backup unlocking mechanism succeeded or is locked if the backup unlocking mechanism failed
  - **Priority:** Medium
- **UC8.2: Emergency unlock during manual intervention when the primary lock does not unlock**
  - **Description:** The fire alarm input is off and the normal locking mechanism failed to disengage. The normal locking mechanism was determined to have failed within or at the end of three seconds since the time the locking mechanism was told to disengage

- **Trigger:** The fire alarm input is off and the normal locking mechanism failed to disengage
- **Precondition:** The system is online, the fire alarm input is off, and the normal locking mechanism failed to disengage
- **Actors:** Backup unlocking mechanism, manual intervention input, and alert system
- **Flow of events:**
  1. Alert is activated to inform people nearby of the unlock difficulties. Manual intervention from a trusted user is requested to force the backup unlocking mechanism to engage
  2. If a trusted user engages a manual intervention device such as a doorknob, then this sequence proceeds
  3. The alert status is updated to reflect how much longer the manual intervention input needs to be held on, which counts down from 15 seconds to 0 seconds
  4. If the manual intervention input was held on for 15 seconds, the backup unlocking mechanism is engaged
  5. Backup unlocking status is checked every 500ms to see if it succeeded. Once it succeeds, it updates the alert to indicate the door's unlocked status
  6. If more than 5 seconds have passed since the door's backup unlocking mechanism was engaged, then the alert status is updated to indicate that the door failed as well as indicate which components failed
- **Postcondition:** If a trusted user chose to engage the backup unlocking mechanism, then the door is unlocked if the backup unlocking mechanism succeeded or is locked if the backup unlocking mechanism failed. Otherwise, this use case does not change the state
- **Priority:** Medium

R9: Door status visibility

- **UC9.1: Update door status display**
  - **Description:** When the door's status changes, update the door status display if one is installed
  - **Trigger:** The door's status changes, such as changing from locked to unlocked
  - **Precondition:** The system is online
  - **Actors:** Door status display and the software system
  - **Flow of events:**
    1. The status on the door changes, such as the locked/unlock state
    2. The software tells the door status display to update its display with the new status
  - **Postcondition:** The door status display shows the new status
  - **Priority:** Low

R10: Configuration memory

- **UC10.1: Read the state of the configuration when desired**
  - **Description:** When the software wants to read the state of the configuration, it will get the configuration
  - **Trigger:** The software wants to read the state of the configuration
  - **Precondition:** The system is online
  - **Actors:** Configuration and the software system
  - **Flow of events:**
    1. The software system signals its desire for a configuration element using its key
    2. The state of the value for the given key is read
       1. If not successful, return an error
    3. That value for the requested key is returned
  - **Postcondition:** The software system receives the value for the desired configuration key if successful. If not successful, an error is returned
  - **Priority:** Medium
- **UC10.2: Update the configuration**
  - **Description:** A configuration is updated to reflect changes to the installation
  - **Trigger:** An installer changes the configuration
  - **Precondition:** The system is offline
  - **Actors:** Installer and the configuration
  - **Flow of events:**
    1. Installer changes the configuration, which is implementation-dependent
  - **Postcondition:** The configuration matches the updated state, which will be reflected when the configuration is read after the system is put online
  - **Priority:** Medium

References

Images in Drawing 2 came from the following web sites:

http://www.securitybase.com/Web_store/keypads/images/9212.jpg
http://www.peakwindow.com/images/Stanley_Premium_Fiberglass_Entry_Door.jpg
http://us.st11.yimg.com/us.st.yimg.com/I/elights_1918_4094556
http://www.elights.com/delfloodsen.html
http://us.st11.yimg.com/us.st.yimg.com/I/yhst-37697109791737_1922_495912
http://www.ambientweather.com/amwewe.html