Danger-System: Exploring New Ways to Manage Occupants Safety in Smart Building

Andrea Piscitello, Francesco Paduano, Alessandro A. Nacci, Danny Noferi, Marco D. Santambrogio, Donatella Sciuto

{andrea.piscitello, francesco1.paduano}@mail.polimi.it
{alessandro.nacci,marco.santambrogio, donatella.sciuto}@polimi.it
danny.noferi@telecomitalia.it

Politecnico di Milano, Telecom Italia S.p.A.

***Abstract*—Safety has always been a main concern in complex buildings: automated solutions to respond to crisis events limiting damages and victims have been researched and adopted in commercial and private buildings since late 19th century. However, the methods available have some limitations: for example the impossibility to promptly detect false alarms and to provide context-dependent notification, which are fundamental to persuade tenants to a fast egress. In this paper we present the architecture of a system for the detection and management of emergencies in smart buildings. The designed architecture shows some novelties and key features which distinguish it from already existing solutions. Indeed, the proposed architecture exploits the existing building management systems leveraging mobile crowd sensing to collect valuable information during a crisis. The new generation of mobile devices are used to dispatch notification and collect users feedback. In order to test and validate the system, we have built a simulation framework which simulates the building environment.**

I. Introduction

Automated solutions to respond to crisis events limiting damages and victims have been researched and adopted in commercial and private buildings since late 19th century. The primary task of these solution is to detect evidences of a possibly harmful circumstance and promptly provide a reaction. The reaction includes one or more countermeasures to limit the damages, such as the trigger of an alarm or the activation of the sprinklers. Escape routes and emergency exits might be highlighted with labels and lighting panels. However standard implementations have some limitations for example the impossibility to promptly detect false alarms and to provide context-dependent notification, which are fundamental to persuade tenants to a fast egress. Indeed, studies of the human behaviour during an emergency evacuation have revealed that it can take up to 15 minutes to escape when being notified of a possible harmful situation [1].

The expectations for an emergency management system in a modern building increased, as well as the technological support this system can rely upon. The main point of the transformation of a traditional building into a *smart* building is the installation of sensor and actuators, supported by IT infrastructures, which can take care of various tasks, including energy efficiency improvements, services and emergency management. The improvements of technology has introduced new devices in our houses, hospitals, schools and workplaces. Indeed, the technologies that are fully exploitable for this purpose include inexpensive and reliable sensors for occupancy detection, surveillance cameras, computational efficient embedded devices and a new generation of mobile devices. The widespread of mobile devices has brought a new contribution to the existing infrastructures in smart buildings. They can be exploited not only as a mere interface to communicate with the user, but they can provide new means for collecting data. The evolution of emergency systems has introduced new challenges in infrastructure design, complexity management and configuration flexibility that traditional solutions are free from. An emergency system requires an infrastructure design which exhibits optimal performance in terms of scalability and reliability. Furthermore it requires to be adapted and tuned according to the context where it is installed in through a sort of configuration interface. Lastly, the set up of a complex emergency system undoubtedly requires a careful testing to ensure its correct functioning. The testing entails some practical difficulties such as the cost and inconvenience of performing evacuation tests. In this paper we propose a solution which addresses the challenge of building an extensible and reliable system for emergency management including a flexible architecture and a full-fledged simulation framework. The aim of our work is to offer a comprehensive method which takes care of two different aspects: the set up and configuration phase of the system, which we will refer to as *offline management*; and the emergency detection, notification to the users, feedback collection and monitoring which we will refer to as *online management*. In the following section we provide an overview of the already existing solution and the related work. Then, in Section III, we will introduce the proposed solution, explaining both the theoretical background and the implementation details. In Section V we present the results we have collected to test the systems. The performed test have been made in a co-simulated environment, in order to observe how the DANGER system behaves with complex buildings. The simulation system is presented in Section IV. In the end, we will derive our conclusions, also proposing future extensions.

II. Related Work

Commonly, mandatory safety infrastructures are installed in public and commercial building, depending on the current regulations of the country. The safety systems are composed by three groups of components: initiating devices, actuators and control panels. Popular initiating devices include smoke detectors and heat detectors, but other less common devices are loud noise detector and chemical sensors. Sounds alarms, sprinklers and automatic fire door retainers are examples of common actuators. The functioning of control panels differs according to the technological advancement of the system.

This work is partially funded by Telecom Italia S.p.A., Strategy and Innovation / Open Innovation Research, Joint Open Lab S-Cube.

Typical tasks enabled by a control panel are the customization and tuning of the system, monitoring of malfunctions and historical data collection. However, very basic systems may not include a control panel and implement the behavior directly on the actuator or on the initiating device. Another family of security systems is the intrusion detection system. Intrusion detection systems are often featured with Passive Infra-Red (PIR) sensors and surveillance cameras. Some systems operate automatically and forward the alarm directly to the police department, others require a human supervisor. The traditional solutions for a safety system include some limitations. Architectural ones restrict the interoperability between different components; oversimplified control panels limit the configurability of the system and technological restrictions hamper a user-centered notification and feedback system. The insufficiency of the traditional building safety systems have motivated the development of new technologically advanced solutions for building security.

In the last years, sensors and other technological devices have become less expensive. This has facilitated their adoption in Smart Building and brought new potential to the existing safety systems. Some of them limit to augment the power of existing hardware by simply wrapping it inside a *smarter* software layer. For instance computer vision exploits already existing video surveillance systems and analyzes the video streams coming out from these. This technique has been used for security purposes by Toreyin et al. [2], to detect fires. Another usage example is presented by Tomastik et al. [3], who describes a model for providing real-time estimates of building occupancy to first responders during emergencies. Although Computer Vision strength is based on the economic advantage of using already existing hardware, it is constrained to the high computational effort that is required and to the sensitivity to false positives. Since the interest in finding a way to determine the occupancy in buildings is strong, other approaches have been presented. While GPS is totally unsuitable for indoor applications [4], a model for exploiting Passive Infra-Red (PIR) sensors is presented in [5]. Similarly to computer vision this approach considers the employment of already existing sensors. Another way to provide occupancy detection is provided by iBeacon, a Bluetooth Low-Energy protocol by Apple [6].

In the context of emergencies management, Filippoupolitis et al. [7] deal with the possibility to have a distributed computation of the shortest escape path dynamically taking into account hazard spreading. Furthermore, they consider the possibility to exploit Smart Buildings infrastructures (e.g., panels and PDAs) for broadcasting important communication during emergencies. Along the same lines of the previous work, the exploitation of mobile devices is becoming popular in the academic community. Smart-phones, tablets and even wearable devices, are really attractive because they can both be employed as a medium to reach the user and to collect information from him. In the former case, an interesting research [8] (related more closely to natural disasters) highlights the possibility to build an ad-hoc network exploiting the Wi-Fi and Bluetooth connections of smart-phones in order to propagate information even without cellular connectivity. Mobile devices are well known for their sensing capacities. Being endowed with many sensors (e.g., microphone, camera, accelerometer...) they can be used to obtain several types of information. This kind of applications are interesting also because they can take advantage of the computational power of the sensing devices in order to perform context-aware tasks and make inferences on the surrounding environment. In [9], an interesting field of application is represented by the identification of a car accident exploiting smart-phones accelerometers. This is accomplished in the perspective of speeding up and increasing the efficacy of rescuers. Another usage example is provided by [10] and [11], that propose a technique to detect the fall of a person. This task can be pursued through the exploitation of smart-phones sensors and can result quite important in fields such as the health of the elderly. Some of the aforementioned techniques have been used to build complete systems for the purpose of emergency management. The SaveAlert system proposed by Tuncay et al. [12] encompasses some concepts illustrated in this paper. The key idea in common between SaveAlert and our work is to exploit users smartphone for sensing the environment for both detecting possible dangers and to interact with the users. SaveAlert implements a very straight-forward architecture composed by sensors, smartphones and the server. Itria et. al illustrate a quite complex system for a crisis management [13]. In this system, information is processed and classified by means of a sql-like event programming language. The main concern of these two systems is the lack of integration with already existing building management systems in order to let administrators define custom reactions.

From the analysis of the state of the art of emergency management systems two considerations emerged. Firstly, although different approaches are proposed, many of them are limited because of the specificity of their purposes. Secondly, many of the broad and complete systems proposed are not customizable enough to take care of all the task involved in the crisis management. The need of a global view appears indispensable for dealing with emergency management. Moreover, the exploitation of distributed mobile systems seems to be a key point in these kind of applications but it should be exploited in a more extensive way. Indeed, mobile phones can be used to implement crowd-sensing. This term refers to applications where individuals with sensing and computing devices collectively share data and extract information to measure and map phenomena of common interest [14]. This technique, together with the integration with existing smart buildings infrastructures and management systems, seems to represent the future trend in this research ares and and thus is one of DangerSystems's strong points. The authors' aim is to provide a cross-cutting tool for emergency management that is based on ubiquitous devices and that can take advantage of many different kind of information to detect an emergency scenario. It is intended for supporting building administrators, adapting to their necessity and being easily customized and extended. The proposed system is also able to manage the following phase of the emergency providing a concrete support to tenants in order to help them escaping in the shortest time possible.

III. Architecture

Today smart buildings abound in structures, systems and technologies which collect information but offer features that are not fully exploited. These infrastructures could allow to identify safety threats and perform countermeasures, which represent the core of emergency management. Furthermore, they could achieve two other important features in order to reduce the amount of damages: a context-based notification and a widespread monitoring and feedback collection. The purpose of our research is hence to design a very general system that offers a trasversal support to emergency management, easily interfacing with existing system and infrastructures. The aim of the authors is to present the main concepts at the base of the system providing some functional application examples.

DangerSystem basically manages the following fundamental phases of an emergency.

■ ***Detection Of The Safety Threats*** — The potential security threats are described in a rule-based form. Every rule is activated when the output of the sensors in the building, together with additional information elaborated by the system, matches prearranged patterns. When a rule is triggered the system detects a safety threat. The various

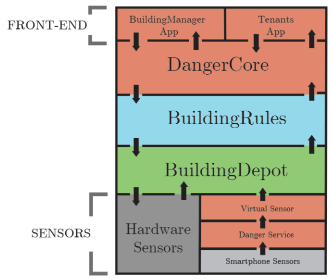


Fig. 1: Architecture of *DangerSystem*

kinds of sensors connected to the system enable the possibility of designing sophisticated rules to detect any type of safety threat.

■ ***Notification*** — Beside the traditional means of notifications such as fire alarms, we take advantage of new technological channels to reach people. The notifications are transmitted either over Internet or over the traditional SMS network (such as GSM) and displayed on people's mobile devices. Furthermore, the content of the notification is differentiated according to the location of the user in the building and the group which the user belongs to. We group people in three major groups: *Building Managers* (which receive detailed notification about every relevant event in the building), *Tenants* (which are informed in case of a possible security threat) and *Rescue Services* (which are alerted in case of security threat with all the relevant details to organize a fast and effective rescue).

■ ***Monitoring And Feedback Collection*** — The system is designed to collect all the sensitive information during an emergency. The location of the users and the level of noise in every room are just examples of the important information that are gathered with the only support of mobile devices. Furthermore, during an emergency, users are prompted with an accessible interface which allows a direct call to the emergency number and to submit basic information to the system, such as the presence of injuries or a false alarm report.

■ ***Reaction*** — The system has access to the resources of the building. Thus, depending on the type of existing infrastructures, the system can be programmed to activate fire alarms, custom notification on smart screens, automatic doors opening, etc... Reactions are described in a rule-based language where the antecedent of the rule is the type of emergency and the consequent is a set of actions on the building infrastructure.

Starting from the aforementioned phases, we designed an architecture which guarantees high flexibility and interoperability with other external components. One of the main architectural requirements is to expand already existing infrastructures instead of providing a brand new solution, in order to facilitate the adoption of the system. Therefore, the functioning of the proposed architecture is backed up by two components: **BuildingRules** [15], a trigger-action system for management of commercial buildings and **BuildingDepot** [16], an extensible and distributed architecture for building data storage, access and sharing. *BuildingRules* provides architecture that is needed to handle the rule-set and takes care of the conflict management among the rules. *BuildingDepot* provides an abstraction layer for data management regarding both rules and building features. Although we assume that the building is already featured with the two mentioned components, the architecture design can be adapted to any building management system and rule based system.

Below, a brief overview of the main architectural components that have been introduced is provided (Fig. 1).

■ ***DangerCore*** — DangerCore is the kernel of DangerSystem. Its main purpose is to take care of all the emergency implications in order to provide other components of the system with high level information. This component is in charge of collecting all data related to the emergency and to directly interact with BuildingRules. DangerCore sends notifications and collects feedbacks mainly from two mobile applications: Building Manager App and Tenants App and elaborates all the data retrieved by the users in order to provide a final verdict to BuildingRules. For instance, depending on the user feedbacks, DangerCore can decide (among the other options) to set the state of an emergency to concluded, to specify the kind of emergency, or to constrain it to just an area of the building. This decision is sent to BuildingRules which changes the state of the antecedent of the associated rule accordingly. All the communications are performed by exchanging JSON-encoded messages. The DangerCore has been implemented with a web-server structure which exposes REST APIs and persistent socket connections for few high responsive services. The REST APIs leverage the Flask microframework [17].

■ ***Building Manager App*** — Developed for Android, the Building Manager App allows the Building Manager to perform his tasks. If a possible emergency situation is detected by the system, DangerCore notifies the Building Manager about this event. In this scenario, the Building Manager's role consists in verifying whether a real emergency is occurring or not. In order to accomplish this task he can obtain further information about the emergency retrieving it from users device sensors or from physical ones. For instance he can record audio from smart-phones microphones and depending on the collected information, he can decide whether confirm or ignore the emergency. This decision is then forwarded to DangerCore which will update the status of the emergency.

■ ***Tenants App*** — Developed for Android, the Tenants App is mostly used to provide building occupants with real-time notifications about the occurring emergencies and to collect feedbacks from them, in respect of their privacy preferences. Whenever an emergency is confirmed by the Building Manager, all the tenants are notified by DangerCore. The application shows all the relevant information about the danger and prompts an option to call directly the emergency services. Depending on the position of the user, customized notifications can be provided in order, for instance, to avoid obstacles or threats that can be close to him. The application is also used for retrieving from users information about the danger, in order to help other users or rescue services. Since DangerCore is aware of both users and threats positions, the users that are strictly close to the danger can be asked to confirm or ignore the danger or to insert more detailed information about it. The system keeps collecting feedbacks and dispatching notifications during a crisis. This increase the security of the tenants and the efficiency of the rescuers.

■ ***Danger Sensing Service*** — Danger Sensing Service embodies the ubiquitous features of DangerSystem. Crowd sensing, (that is the concept of retrieving and analyzing data from a huge user base) is accomplished by making Danger Sensing Service run on every building occupant mobile device. It collects data from the device sensors and elaborates it in order to detect particular behaviours: i.e. it is able to detect the run of a person or if a person has fallen thanks to a pedometer algorithm based on [18] that has been implemented to monitor the user's movements. Danger Sensing Service can use the microphone of the phones to identify loud noises and, if requested, to stream audio samples to the Building Manager App. This feature is intended to be used in extremely dangerous situations where the tenant are not able to directly communicate with the rescuers (e.g.

sequestration, kidnapping).

■ ***VirtualSensor*** — VirtualSensor is a cluster of simple sensors that are managed in order to derive high quality information exploiting multiple data. The key point of a VirtualSensor is the aggregation of data. This process consists in retrieving data from the simple sensors and using it in order to find general patterns. It permits to obtain information from data that singularly is not significant. For instance, a VirtualSensor can be built to aggregate data from mobile devices. It can be set-up in order to manage all the devices inside a single room. In this way if a statistically significant number of devices detect a run, the VirtualSensor triggers a mass-movement event in BuildingRules.

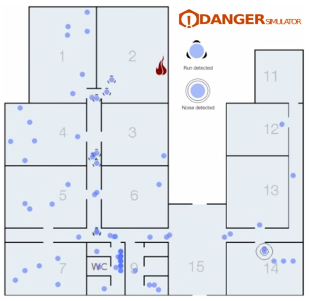
A peculiarity of the system is that the connections between users and system can dynamically change channel of communication, in case of malfunctioning of the network. This behaviour is implemented by switching between the Wi-Fi connection, to the mobile 4G/3G/2G Internet connection and even to the common SMS-based networks when necessary. Indeed, in case that the Internet connection results to be unreliable, the system can still dispatch notifications and collect feedback from users' devices by means of SMSs. In order to understand which communication channel has to be used (WiFi, 3G, GSM-SMS, etc...), the BuildingManager adopts the following technique: given ***N*** (the number of devices), ***PollingTime*** (time interval between a device request and another one) and the ***Accuracy*** (level of accuracy requested by the system in detecting a connection lost), there are basically two different cases: the former is related to the case when due to some major issues, the connection is not available for every device; the latter is the case when a single user, due to some problem with its connection, is not able to connect to the system. If every device usually make a polling request every *PollingTime* seconds, and suddenly no one of the devices is able to access the service, the service can estimate how long to wait before considering the connection down with the given *Accuracy*. For example, if the system hasn't received any request in t seconds, it can suppose that the connection is down with this accuracy:

$$Accuracy = 1 - \left[\frac{PollingTime - t}{PollingTime}\right]^N \quad \text{with } t \leq PollingTime$$

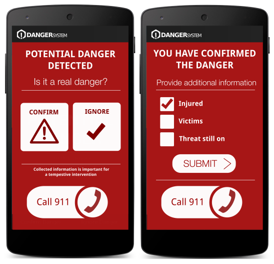
According to our measurements, an Android based device performs a polling request every *PollingTime* seconds within a certain confidence level. Hence, our system can estimate how long to wait before classifying that single device as disconnected from the network with the requested *Accuracy*.

Apart from the *online managment*, the proper functioning of an emergency system also relies on a correct *offline managment*. As previously mentioned, our system is configured by means of rules to both describe what an emergency is and to specify the system reactions. The set of rules involved in a building can become very large, and this raises the issue of how to validate the system configuration. For this purpose we equipped our system with supporting tools that deal with several validation tasks. Indeed, we provided the system with a simulation framework that can reproduce the characteristics of the real building and of its occupiers allowing to run various kinds of simulations. It is intended for supporting and validation tasks as for instance:

- **Sensor and actuators placement:** the simulator can be used to design the alarm system of the building, identifying the optimal location for sensors and actuators or testing their effectiveness.
- **Rules and alarms trigger testing:** a simulation of a danger can be run in order to verify that the correct procedures are triggered and everything from the alarms to the notifications to the users works properly.



(a) Simulator: fire evacuation



POTENTIAL DANGER DETECTED

Is it a real danger?

Collected information is important for a tempestive intervention

YOU HAVE CONFIRMED THE DANGER

Provide additional information

- [✓] Injured
- [ ] Victims
- [ ] Threat still on

(b) *Tentants App*

Fig. 2: Demonstrative screenshots

- **Evacuation simulation:** the behavior of building occupiers can be analyzed in order to validate the effectiveness of the escape paths and of the notifications systems.
- **Personnel training:** this tool can interact directly with real devices. Thus, it can be used to train the occupiers of the buildings. It can be adopted to instruct the Building Manager to become familiar with all the features he is endowed with during an emergency; or to teach tenants on how to properly interact with the emergency system.

IV. Danger Simulation Framework

In order to test the designed system, we created an agent-based simulation environment which simulates people behaviour in a building and which is able to interact in real time with an actual deployment of the *DangerSystem*. The underline idea was to crate a *fake simulated building* and to make it interact with real mobile devices, in order to test the proposed emergency detection methodologies. Already existing simulation solutions, such as the popular NetLogo [19], did not fit our needs because of the inability to interact with an external system. The simulator is written in Python 2.7 and it is composed by three main components: back-end, front-end and interaction layer. The **back-end** is composed of a building and a set of agents which "live" inside it. The planimetry of the building is loaded from a set of file images which represent the walkable and not walkable areas and the rooms subdivision. The agents represent the building occupiers. Each agent has a behavior, which is a high level control of the agents movements. A behavior guides the agent in performing a task by assembling one or more basic actions. For example, in case of an emergency, the agent's behavior tries to make the agent escape by assembling actions such as run out of the closest door and run through the shortest path to outside the building. This examples of behaviors are described inside modules and as already mentioned can be easily extended. Built-in in the back-end there is a trigger-event system which is meant to be exploited by the modules. A module may subscribe to a certain kind of trigger in order to react to the activation of this. For instance a trigger can be represented by the detection of a security threat. In this way a simulation for that particular threat can be launched. **The front-end** is mainly composed of both a graphical and a very simple command line user interfaces. The GUI has been developed using PyGame, a cross-platform set of Python modules designed for writing video games (Figure 2a). Its main task is to visualize planimetry and components of the building. It also shows the state of the building and the agents moving inside it. Emergency threats are also shown when they are detected and they are defined inside modules so that new ones can be easily added. In order to let

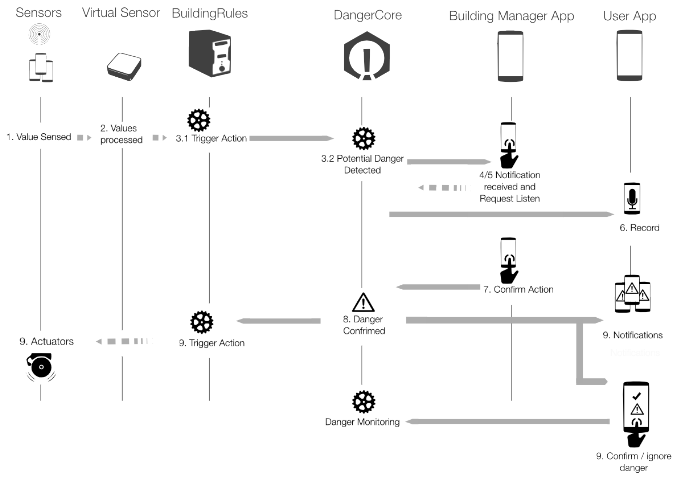


Fig. 3: Sequence of interactions in the *DangerSystem* in the scenario described

the user easily interact to the simulator, a CLI is provided to him. By using it, a user could easily simulate the presence of a threat in a particular room to test the behavior of the building. This would activate a trigger inside the back-end thus activating modules which are subscribed to it, as described in the previous section.

All these behaviors have been arbitrarily defined by us and are not close to real behaviors. Some studies about real behaviors of people inside building (especially during emergencies) can be used to build a more realistic model.

V. Experimental Setup and Results

In this section we briefly describe the setup we have done to test the proposed system, presenting the use case scenario that has been experimented with the simulation framework and inside a real building. We will then discuss the collected results.

■ ***Scenario*** — In this section we present the interaction among all the components of *DangerSystem* during a simple scenario. The scenario involves a building with four rooms; we will refer to them as *Room A, B, C, D*. In room A some loud noise is produced and people start escaping. Thanks to the ability of *DangerSystem* to perceive whenever loud noise and fast mass-movement happen in a room, the Building Manager is promptly notified about this suspicious situation. After collecting feedbacks from the users and listening some audio samples from their smartphones, the Building Manager uses the *Building Manager App* to trigger the reaction of the emergency system. Then, the system dispatches a notification to all the tenants in *Room B, C, D* and activates the sound alarm of the building. In this situation, as illustrated in Figure 3, *Danger* System operates in the following way:

1) The mobile application, installed in users' smartphones, detects that the user is running and that a loud noise is present in the surroundings. They send this information to the *VirtualSensor*;
2) *VirtualSensor* processes and aggregates the information of multiple user;
3) *BuildingRules* accesses the data of the *VirtualSensor* via *BuildingDepot*. *BuildingRules* has a rule whose trigger specifies that loud noise or fast movements in that room has to be considered a potential emergency. The action of this rule is to notify *DangerCore* of this potential emergency;
4) *Building Manager App* receives the notification of the potential emergency. Thus, it automatically opens a full screen view which let the Building Manager to interact with the DangerSystem;
5) *Building Manager App* displays the feedback collected from the user, and the Building Manager requests to listen an audio sample from an user's smartphone. The request is sent to the DangerCore, which forward the request to the user's application;
6) The Building Manager decides to confirm the emergency. The request is forwarded to the *DangerCore*;
7) *DangerCore* marks the emergency as confirmed, and sends the notifications to all the users;
8) *BuildingRules* is informed of the confirmed danger. Then, in this set up it has a trigger-action rule which has as antecedent "confirmed emergency" and the corresponding action is "switch the alarm on". It forwards the request to switch the alarm on to *BuildingDepot*;
9) *BuildingDepot* switches the alarm on;

■ ***Preliminary user study*** — The previously described scenario was tested with a group of 15 tenants and one Building Manager. Five tenants were located in *room A* , three in *room B* and *room C* and four in *room D*. People did not have the ability to communicate with nobody else than who they were sharing the room with. The tenants were carrying out office activities. They had been informed that they were going to participate in a test of an emergency system but they did not know the design and the functioning of the application. People in *room A* had been asked to run away, as in an actual dangerous situation, after that a loud noise was emitted from some speakers. The test was repeated a second time changing the roles of the tenants. Table I reports some relevant time interval measured in the two instances of the test. *Notification delay to Building Manager* is the time interval since when the loud noise has been emitted and the potential dangerous situation has been detected and notified to the Building Manager's mobile device. *Average time to interact with the phone* is the time interval since when the notification has been dispatched to the mobile device of the tenants in *room B,C,D* and they actually has read the notification. After the test the Building Manager and the tenants were asked to rate the effectiveness of the system according to different metrics. Particularly relevant is the question: *"How big has the impact of the notification on the mobile application been in determining the necessity of a quick escape? (1 no impact - 5 fundamental)"*. The average score of 4.2 suggests that providing actual information about the danger may be determining for a quick escape of tenants, that otherwise would hesitate because of the lack of certainty about the danger.

Event	Round 1	Round 2	mean (s)
Notification delay to Building Manager	3.5	4.2	3.85
Average time to interact with the phone	7.1	8.5	7.8
Room B evacuation time after notification	22.2	20.0	21.1
Room C evacuation time after notification	19.0s	21.7	20.35
Room D evacuation time after notification	18.2	17.0	17.6

TABLE I: Interval time measured during the two rounds

VI. Limitations and Future Works

As previously mentioned in other sections of this paper, *DangerSystem* is a general system that has been built in an extensible and modular fashion. The main purpose of the implementation that we have presented is to prove of the functioning of all the built components and to validate the whole system. However, the implemented

Metric	Users	Average Grade (1-5)
App intuitiveness	Building Manager	4
App intuitiveness	Room A	4
App intuitiveness	Room B,C,D	3.7
Influence of app on escape	Room B,C,D	4.2

TABLE II: Questionnaire Evaluation Results

features encompass very basic behaviors of the system. Further studies in the perspective of improving the overall accuracy of the system (or to extend it) can be conducted. It is important to highlight that the results coming from these studies can be easily included in the system thanks to the intrinsic extensible nature of this. A very interesting approach to detect a potential danger situation could be to learn which is the normal situation in the building. With the terms normal situations we refer to the probability distribution of all the input from the sensors related to the danger detection. Thus, an emergency could be detected simply testing the current distribution of values against the normal situations and check whether there is a considerable difference. Another possible extension is represented by the detection of other key movements or even patterns in crowd behavior. This can be demanded to both mobile user application (for individual movements) or to *VirtualSensors* (for collective movements). In this perspective many studies may be conducted or already existing ones may be exploited.

VII. Conclusions

In this paper we presented *DangerSystem*, a solution for emergency detection, management and monitoring in smart buildings. *DangerSystem* is able to handle all the phases of an emergency, gathering data from several sources and aggregating it in order to obtain accurate and precise information about dangers in a building What emerges from this study is the potential of a system which leverages both the existing building infrastructures and mobile devices. Indeed, mobile devices are used both as a mean to collect data and to dispatch notifications. A rule-based definition of emergencies and reactions makes our system extremely flexible. Indeed, an appropriate set of rules facilitates the description of the behavior of the system in many different and complex scenarios. The effectiveness of the configuration can be tested with a simulation framework. The simulation framework can facilitate the adoption of the system in real buildings by offering tools for the training of the Building Manager and the tenants. Users that joined the study reported a high level of satisfaction with the usability and usefulness of the system and this is an encouraging signal for future works and enhancements of the system itself. However, we validated the system with just a small set of users, and in particularly simple scenarios. It is therefore difficult to draw definitive conclusions and it should be seen as a promising initial field trial. We are currently planning a larger study involving more users as well as analyzing possible effects that the use of *DangerSystem* can have on the tenants of a building.

VIII. Acknowledgement

This work was partially supported by the Joint Open Lab S-Cube - Telecom Italia S.p.A., Open Innovation Research division, Italy.

References

[1] G. Proulx, “Evacuation time and movement in apartment buildings,” *Fire safety journal*, vol. 24, no. 3, pp. 229–246, 1995.

[2] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Cetin, “Computer vision based method for real-time fire and flame detection,” *Pattern recognition letters*, vol. 27, no. 1, pp. 49–58, 2006.

[3] R. Tomastik, Y. Lin, and A. Banaszuk, “Video-based estimation of building occupancy during emergency egress,” in *American Control Conference, 2008*. IEEE, 2008, pp. 894–901.

[4] H. Liu, H. Darabi, P. Banerjee, and J. Liu, “Survey of wireless indoor positioning techniques and systems,” *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 6, pp. 1067–1080, 2007.

[5] B. Song, H. Choi, and H. S. Lee, “Surveillance tracking system using passive infrared motion sensors in wireless sensor network,” in *Information Networking, 2008. ICOIN 2008. International Conference on*. IEEE, 2008, pp. 1–5.

[6] G. Conte, M. De Marchi, A. A. Nacci, V. Rana, and D. Sciuto, “Bluesentinel: a first approach using ibeacon for an energy efficient occupancy detection system,” in *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2014.

[7] A. Filippoupolitis and E. Gelenbe, “A distributed decision support system for building evacuation,” in *Human System Interactions, 2009. HSI’09. 2nd Conference on*. IEEE, 2009, pp. 323–330.

[8] X. Wu, M. Mazurowski, Z. Chen, and N. Meratnia, “Emergency message dissemination system for smartphones during natural disasters,” in *ITS Telecommunications (ITST), 2011 11th International Conference on*. IEEE, 2011, pp. 258–263.

[9] C. Thompson, J. White, B. Dougherty, A. Albright, and D. C. Schmidt, “Using smartphones to detect car accidents and provide situational awareness to emergency responders,” in *Mobile Wireless Middleware, Operating Systems, and Applications*. Springer, 2010, pp. 29–42.

[10] C. Tacconi, S. Mellone, and L. Chiari, “Smartphone-based applications for investigating falls and mobility,” in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*. IEEE, 2011, pp. 258–261.

[11] F. Sposaro and G. Tyson, “ifall: an android application for fall monitoring and response,” in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 2009, pp. 6119–6122.

[12] G. S. Tuncay, K. Varshavskiy, and R. Kravets, “Demo: Savealert: Design for a sensor-driven crowdwatch danger detection system,” in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014, pp. 362–363.

[13] M. L. Itria, A. Daidone, and A. Ceccarelli, “A complex event processing approach for crisis-management systems,” *arXiv preprint arXiv:1404.7551*, 2014.

[14] R. K. Ganti, F. Ye, and H. Lei, “Mobile crowdsensing: current state and future challenges,” *Communications Magazine, IEEE*, vol. 49, no. 11, pp. 32–39, 2011.

[15] A. Nacci, B. Balaji, P. Spoletini, V. Rana, R. Gupta, D. Sciuto, and Y. Agarwal. (2014) Buildingrules: A trigger-action based system to manage complex commercial buildings. [Online]. Available: https://energybox.necst.it/buildingrules

[16] Y. Agarwal, R. Gupta, D. Komaki, and T. Weng, “Buildingdepot: an extensible and distributed architecture for building data storage, access and sharing,” in *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. ACM, 2012, pp. 64–71.

[17] A. Ronacher, “Welcome— flask (a python microframework),” *URL: http://flask.pocoo.org/(visited on 02/02/2015)*, 2010.

[18] N. Zhao, “Full-featured pedometer design realized with 3-axis digital accelerometer,” *Analog Dialogue*, vol. 44, no. 06, 2010.

[19] S. Tisue and U. Wilensky, “Netlogo: A simple environment for modeling complexity,” in *International conference on complex systems*, 2004, pp. 16–21.