Towards a Digital Twin System for Human Crowd Motion Prediction

Ignacio Alba, Javier Troya, and Carlos Canal

Universidad de Málaga - ITIS Software, Spain
{nachoalav,jtroya,carloscanal}@uma.es

Abstract. In many local festivals and in events such as pop concerts or sporting events, crowds might generate dangerous situations like the case of panic stampedes. How to anticipate these situations is challenging. Despite there are simulation models of people motion, it is very difficult to predict real people behavior. In this paper, we advocate the application of digital twins to monitor and analyze real people motion. We implement the *Artax* framework, which stores people movements as traces through time. We describe the current state of our framework and explain how we aim to apply it for anticipating dangerous situations caused by crowds and warning people.

Keywords: Digital Twin · Smart Cities · Human Motion.

1 Introduction

In many countries and cities, tourism is one of the pillars of economy. In touristic cities open to the ocean or with a passable river, big cruise ships almost daily dock in their ports and thousands of tourists disembark and fill the city in peak season. If all the tourists decide to visit the same touristic points at the same time, they will likely produce crowds. There are also events in which crowds are always generated, such as pop concerts, sporting events and demonstrations. Even when celebrating the local festival in some cities crowds will for sure take place. We can think for instance of Sanfermines (Pamplona, Spain), Oktoberfest (Munich, Germany) or Carnival (Rio de Janerio, Brazil). Finally, there are also overpopulated cities where crowds are generated daily, such as Tokio (Japan), New Delhi (India), Shanghai (China) or Mexico City (Mexico).

Crowds might cause serious dangerous situations [23], especially in a panic stampede [4]. Therefore, there is the need to find solutions that try to avoid these situations. Smart Cities aim to use technology for providing useful services to their citizens and to solve urban problems [2], however, not much has been done in the context of avoiding problems caused by people crowds.

There are approaches that try to simulate people dynamics by modeling their behavior. For instance, the authors in [22] apply a multi-group microscopic model to generate real-time trajectories for all people moving in a virtual defined environment. The work in [3] combines modern game development technologies with traditional agent-based modeling methods for simulating people flow at an

airport, and there are other approaches that try to focus on the behavior in individuals within crowds [13]. However, all these works are about simulation, and they do not consider real people moving in a real environment.

A Digital Twin (DT) is a comprehensive digital representation of a system, employing models and data for representing its properties, conditions, and actions [24]. DTs are continuously updated with real-time information about their physical counterparts [14], facilitating two-way feedback between their digital model and the physical entities represented in it, allowing real-time monitoring, adopting adaptation policies, and enhancing decision-making processes. The use of DTs in Smart Cities, conforming the so-called Urban Digital Twins or Digital Twin City, has already been proposed [16,5]. The problem, again, is that current works do not focus on human crowds.

This paper presents ongoing work around a framework, named *Artax*, with the ultimate goal of addressing dangerous situations caused by crowds. We define an architecture that considers real people movements gathered by tracking the positions of their smartphones. Our framework is able to gather together the position of many people through time by storing them in traces. Since an experiment with real people might take a considerable amount of time and resources, our framework implements a simulated scenario integrating state-of-the-art tools, and it is prepared to replace the simulated data with real data. Regarding the simulation part, our framework allows to configure different parameters so that people motion is simulated for our purposes.

The rest of this paper is structured as follows. After this introduction, in Section 2 we motivate our work. Then, Section 3 describes different aspects of our framework, while Section 4 gives some details of its implementation. After this, a couple of validating scenarios are described in Section 5, and conclusions and future work are summarized in Section 6.

2 Motivation and Background

Smart cities apply Information Technologies to solve specific urban problems, such as the identification of traffic patterns to improve urban mobility, or improving the accessibility of city resources and utilities in order to provide better services to their citizens. Digital Twins are being proved as a promising approach for the development of smart cities (see for instance [24] for a review of the literature). A Digital Twin City [5] —also known as Urban Digital Twin (UDT)— models specific city aspects as an effective way to make a city smart. Similarly to other digital twins, UDTs facilitate two-way feedback between the model and the physical entities represented in it, enabling real-time monitoring for cities, proposing adaptation policies, and enhancing decision-making processes.

Just to mention a few initiatives within the research field of UDTs, Ruiz et al. proposed BODIT [21], a UDT of the public transportation system in the city of Badalona (Spain). They use a traffic simulator and a genetic algorithm to reproduce the city's traffic and adapt to different situations. Bus schedules are used to predict and detect a lack of punctuality at bus stops, enabling informed

decision-making as a response to unusual situations such as accidents. Another interesting related work is that of Lehtola et al. [12] studied the impact of digital twins in smart cities. In their work, the authors emphasize the necessity of taking humans into consideration to ensure successful implementation in order to improve decision-making, which is a concern that we also share.

Indeed, the interest in the social aspects of smart cities is growing fast. For this reason, citizens must be considered as first-class entities of the UDT, since they are the fundamental key to this ecosystem. However, the role of people has often been neglected: smart city applications typically focus on collecting and analyzing data coming from all kinds of sensors, including some related to human activities, but without really integrating individual users into the loop, just considering them collectively, i.e., as a crowd, not as distinct individuals.

In previous works, we presented the concepts of *Human Microservices* [11], and *Digital Avatars* (DAs) [18,19]. These models allow to incorporate individuals into UDT models. This way, we can achieve better planning of city infrastructures, adapting them to the context of the people who use them. Both initiatives derive from the People as a Service (PeaaS) [6] paradigm, which provides a conceptual model for application development focused on the smartphone as an interface to its owner.

In particular, the purpose of a DA is to serve as a digital representation of a person, facilitating their participation in collaborative and social computing systems. Thus, DAs are smartphone-based virtual representations of their users which act as virtual assistants in their interaction with the environment. The purpose of DAs is to exploit citizens' personal information, habits, and preferences for benefiting from better services, while the users keep full control of their data, including storage location and access control [17].

However, one of the problems of incorporating humans into UDT proposals is how to obtain a good dataset coming from a significant number of individuals. While there is a huge amount of open data available for any kind of UDT purposes, data sources coming from groups or real users are scarce[1], and people are reluctant to install apps in their smartphones and share their information even for research purposes. This is where synthetic data comes into the picture. As it will be shown in this paper, in our approach we employ The ONE tool [10] for generating individual path trajectories that although synthetic are realistic enough as input for Machine Learning techniques. Despite The ONE is in origin a tool for opportunistic network simulation [7,8], it has also been successfully used for simulating human path trajectories (see [9,1] as examples).

3 Designing the Artax Framework

In this section we explain how the Artax framework has been designed and modelled. We also present the main elements of its architecture.

[1] One of the rare publicly available examples can be found in `https://www.kaggle.com/datasets/chetanism/foursquare-nyc-and-tokyo-checkin-dataset/data`.

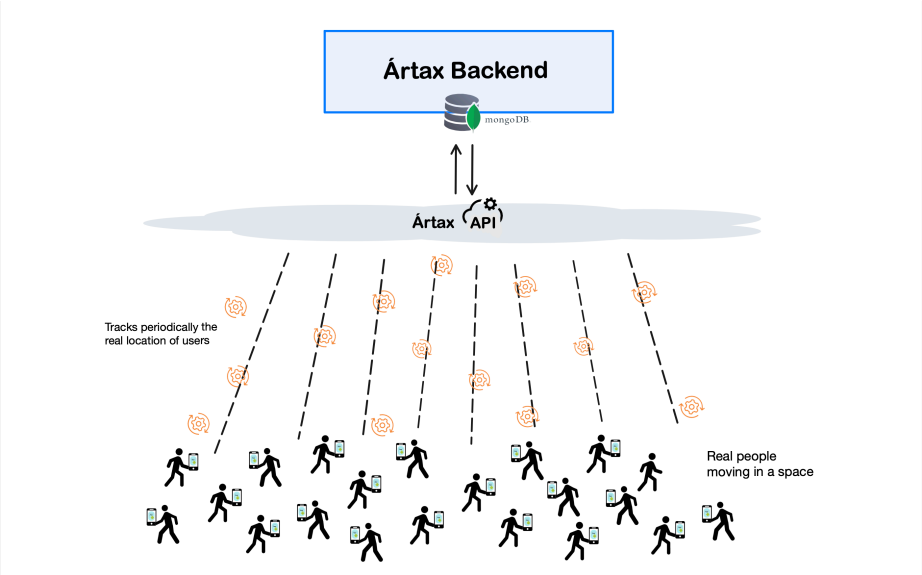

Fig. 1. High-level representation of the Artax framework, collecting traces of movements from its users.

3.1 Overview

The Artax framework collects information about people's movements through their smartphones (Figure 1), which constitutes a digital twin of people motion. Each user's device tracks the movements of its owner, building a trace consisting of a sequence of GPS positions and timestamps. This information is periodically sent to the backend through a REST API, where it is stored. How well the system works (how precise and accurate its predictions will be) depends largely on the quantity and quality of the available information about its users. However, before storing it, data is anonymized, and it can also be conveniently obfuscated so that the actual position of any user at a given time cannot be precisely determined.

Figure 1 portraits a real-life scenario where people are moving throughout a given open space. The system gets the location of each user from the GPS sensors of their smartphone, through a mobile application which constitutes the frontend of the system, as can be seen in the figure. The data collected is periodically sent to the backend where it is used to build and update a dataset of people's flow over the area, which can be used for inferring crowd movement patterns and detecting risk situations.

However, this scenario points out one of the main challenges of our approach: the problem of obtaining enough data for training the system. How to address this problem is described in the next section.

3.2 Dataset of people motion

In order to achieve a digital twin of people motion, we require traces of movements for a large number of users over a given area. With that, predictions of future paths and detection of risks can be inferred by the digital twin. Two alternatives can be considered.

One option consists in having a large amount of people willing to collaborate with our project, where each person installs the Artax mobile app in their smartphone, and gives it permissions to access their location while they spend a certain amount of time moving from one place to another in the area, and possibly producing crowd situations in particular zones. In terms of realism, the collected information would be notably valuable regarding how real people move and how different crowd situations occur. However, this option is hard to achieve in practice as it is difficult to recruit enough people to carry out the required experiments.

An alternative would consist in using a public dataset of people's movements. Some of these datasets exist, but there are also significant drawbacks in them. The main shortcoming is to find a dataset that adjusts to our needs. A good number of the datasets that can be found on the Internet are private, while others would simply not serve our purpose (the traces correspond to a small number of individuals, to people moving through a small enclosed space such as an office, or the movements refer to vehicles, instead of pedestrians, or people travelling from a country to another, to name a few).

As both options have their advantages and shortcomings, we decided to build a synthetic dataset specifically tailored for our needs. This way we can have enough data to train the system and we can precisely define the movement and crowd scenarios that interest us. For this purpose, we set up a simulation environment which is presented in the next section.

3.3 Simulating users and their smartphones

The architecture of the Artax framework in Figure 1 has been extended for overcoming the lack of available data exposed in the previous section. The Artax simulation environment allows the creation of a dataset of people's paths in a realistic way. For that, we start with the generation of traces of movements of a number of simulated or synthetic users sharing a common space with The ONE tool [10], and we feed each of these traces into virtual smartphones in turn simulated with the Perses tool [15]. The virtual smartphones have installed the same Artax app used in the real-life scenario, and they are run in the Amazon AWS cloud platform, where they behave and communicate with the Artax backend exactly as real devices would do. The Artax architecture extended with simulation capabilities is shown in Figure 2, where the right part depicts the simulation environment, showing the analogy between both real-life and simulated scenarios. In the following we describe the main elements of the simulation environments: The ONE and Perses.

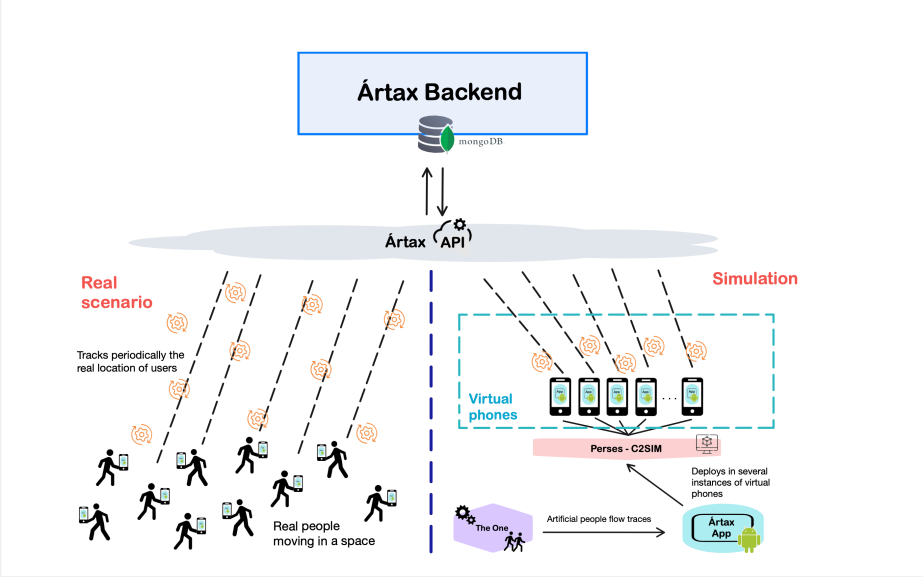

Fig. 2. Artax architecture considering the scenario with real users (on the left) and the simulation environment that reproduces it (on the right).

The ONE. The ONE (Opportunistic Network Environment) is a simulation tool originally developed for the research and evaluation of DTN (Delay Tolerant Network) routing protocols, modelling message routing and handling between nodes representing interconnected mobile devices such as smartphones. The tool is publicly available[2] and it provides information about the simulated scenarios, including all kind of measurements of performance and reports and visualization of the message routing and mobility of the devices, and it is commonly used for experimenting with opportunistic networks.

However, we are not interested in measuring and evaluating message routing, but in mobility. Thus, we use the tool for defining and simulating scenarios in which a certain number of nodes (representing people and their smartphones) move through a given area, specified by a map which describes their feasible paths. The result of the simulation is a sequence of snapshots with the location of all the nodes at each time step, from which we reconstruct each node's path, representing the trace followed by one of the simulated individuals. The number of nodes, the space and paths where they move, and their patterns of movement are some of the parameterized elements on the scenario. It is also possible to define groups of nodes each presenting different behaviors. In order to achieve our desired behavior, we needed to slightly modify The ONE's source code. In

[2] `https://akeranen.github.io/the-one`

particular, we made the positions (coordinates) publicly available and logged them during the simulation while the scenario is being executed.

Once the defined scenario is simulated and the traces of movements are generated, the results are fed into the *Artax* app. This is when Perses comes into picture.

Perses. Perses [15] is a virtualization framework for building scenarios involving virtual mobile devices of different characteristics. The virtual devices are run in the Amazon AWS platform. The tool is specifically designed for evaluating the behavior and performance of complex distributed mobile applications. Similar to The ONE in terms of flexibility, Perses scenarios can also be configured, indicating aspects as the simulation time, the number and type of the virtualized devices, and some of their specific features. The Perses virtualization tool is also publicly available [3].

For simulating a particular scenario we deploy a number of instances of virtual mobile devices, each endowed with the same Artax app already described for the real-life scenario, and the path followed by a given individual, as resulting from The ONE. Then, the virtualization framework starts execution, with the virtual smartphones behaving as in a real-life situation, and communicating with the Artax API for periodically sending their location to the backend.

Once we have described the Artax architecture, the experimental setup, and its main elements, we delve into the technical aspects of the Artax backend in the next section.

4 Implementation

In this section we discuss the technical aspects of the elements described in the previous section.

4.1 Artax Android App

This is the mobile application that tracks the people's locations and sends them to the backend. The main idea behind this app is to have a virtual profile for each user that stores useful data. It keeps some information such as persons' names and surnames, and other data like their age or whether they have any disability, which might be useful information when dealing with dangerous situations caused by crowds (cf. Section 4.3).

As we explained in the previous section and exemplified in Figure 2, our application considers both a real-life and a simulated scenarios. For this reason, the app presents two slightly different execution flows depending on the context where it is running, where the major difference is on the data processed and the resources used. Indeed, the app deploys into virtual smartphones when simulating, so it is better to free the software from some elements that may cause errors

[3] `https://github.com/perses-org/perses`

due to the limitations/restrictions of the platform, or simply just to be cautious. For example, one of the limitations that presents the virtualized smartphones is the lack of GPS access. Before digging into the specific differences of both execution modes, we detail some technical aspects of the app.

Artax has been developed in *Java* using *Android Studio*. The user interfaces, which in Android are layouts, consist in XML files where some components that Android offers are used. It also presents multi-language support for English and Spanish depending on the operative system's settings. The app's main components are described in the following.

- A **client** that communicates with the *Artax* API. This client builds the requests to be made, enqueues them and gets the corresponding API response.
- A **local database** to store the user's information. The app installs a **SQLite** local database the first time that it is executed on the smartphone, with the data being recovered on successive executions by querying the databse. To begin with, we have defined a **User Contract** where the *User* table structure is defined. The most important aspect of this element is the **helper** that we introduced in order to interact and operate over the database. This helper obtains the local database and allows the execution of SQL statements to query or manipulate the table.
  This is not present on the simulation scenario in order to free resources in the simulated smartphones. In addition, it does not make sense to install a local database and store data on those devices since these smartphones are shut down when the simulation is over.
- A Java **class** that models the user so it can be treated as an object.
- The app makes use of Google's **location services** to be able to track the device location while it is moving. For this to be correctly functional, a **Location Listener** has been included which listens to any event related to the device location update or the activation/deactivation of the provider. This is managed by the **Location Manager**, which requests location updates from the listener and indicates the provider that determines the position—this provider is the GPS. This is all encapsulated by a **Location Service**.
  An important aspect to consider is that, to be able to use the location services, first the user has to give the necessary permissions. For this purpose, we introduce a **Permission Manager** that manages which permissions are granted on our application, and can request them. In our case, the most useful permissions are the *COARSE LOCATION* and the *FINE LOCATION* permissions. Basically, the difference them lies in the accuracy in which the position will be determined. The apps gives users the option of choosing whatever choice they prefer.
  Needless to say, the simulated smpartphones do not make use of any GPS service.
- A **Location Monitor** that periodically interacts with the **location service** described above to get the current location of the device. Once the monitor gets the location, it generates the current timestamp and sends this data to the **API client** to forward it to the *backend*. It also updates the view to

display on screen the latest tracked location and logs the motion trace into an external text file.
The periodic task has been implemented using a *single thread executor service* which receives an Android **handler** that runs the task on said thread. The virtualized smartphones use a **Simulation Location Monitor** that implements the periodic task in the same way but does not intercede with any location service.

After summarizing the main components, we describe the execution flow in the two different modes. In the ***real-life functioning***, when users open the app for the first time (the local database is created, as we mentioned above), they are redirected to a form where they introduce their name, surname, birth date and whether they are affected by any disability. Next, the data is stored on the local database and sent to the API, then the user is redirected to the main activity of the application. If users' data have already been stored on any previous execution, when they open the app they are on the main activity. This is determined by querying the local database and checking whether there is any result. The interface of this main activity simply displays the information on the screen. Users can swap what is being shown alternating between the user data and the current location info. When the main activity executes and the user has been queried, then location permissions are asked to be granted by the user and, when said permissions are given, the location service and monitoring activate.

Regarding the ***simulation*** mode, when the simulation starts, the activity that contains the form executes but, since there is no interaction with the user, the data is randomly generated. The generated birth date lies within a period of time from 70 years ago to the present day, and to determine if the "user" is affected by any disability real statistics about disabled population in Spain are taken into account. This data is also sent to the API, but now it is not stored on any local database, since this is not even created. A User object is created with this information and passed onto the main activity. Unlike the other scenario, no permissions are asked to be granted. Instead, the simulation monitor just activates and the task begins to execute periodically.

4.2 Artax REST API

This software follows an architecture based on microservices to foster reliability and flexibility, so this REST API integrates the *frontend* of the system, which is the Android app, with the *backend*. There are two services: one for the users and another for the locations. Both services present basic CRUD functions.

Regarding the database (cf. Figure 2), the data is stored within a *Mongo DB* database that is hosted in a *Mongo DB Atlas* cluster.

The REST API has been implemented using the *Node.js* framework *Express*. We have also used the *mongoose* ODM (Object Document Mapping - analog to an ORM but for non-relational databases) to connect the API to the database and define simple schemas to carry out *Mongo DB*'s data validation. The API REST has been deployed on the *Vercel* cloud platform, so it is publicly available via *https*.

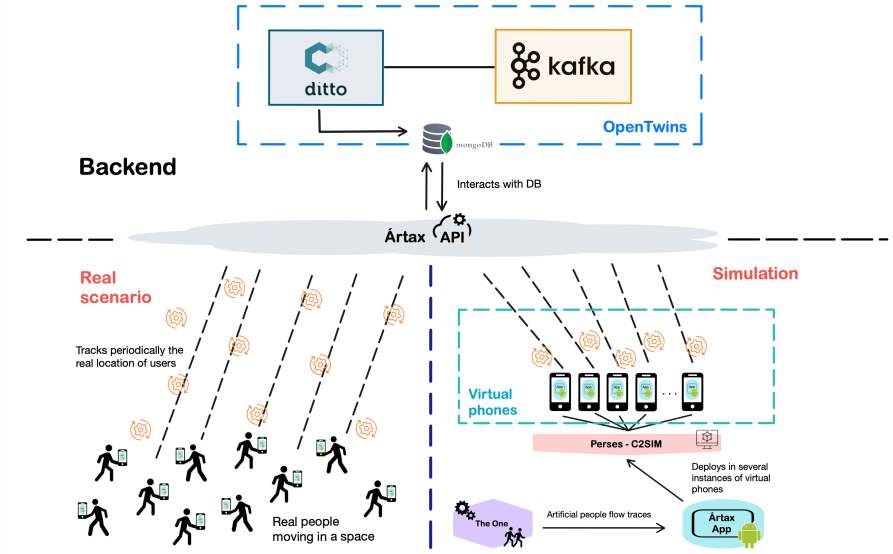

Fig. 3. Artax's full architecture.

4.3 Artax Backend

The Artax backend represents the part of the digital twin that offers the services of forecasting dangerous situations due to crowds and alerting the users. We are relying on the *OpenTwins*[4] framework [20] for materializing this backend, whose structure is displayed in Figure 3.

OpenTwins is an open-source framework, developed at University of Malaga, specifically designed for the development of digital twins. This platform allows the creation and management of digital twins, and it facilitates the orchestration of the different services that intertwine with each other. These services are all open-source and are integrated within the same framework. *OpenTwins* presents a variety of components used to offer different services: from basic digital twin functionality to 3D visualization of the system, IoT data streams, or machine learning aspects. Out of the components offered, Artax uses *Eclipse Ditto* and *Kafka ML*. *Ditto* is one of the most extended solutions for digital twin implementation, whereas *Kafka ML* is the main component in *OpenTwins* that is responsible for providing machine learning features to the system.

In order to make predictions and forecast dangerous situations, we need to train a machine learning (ML) model. We do this on an external platform, namely

[4] `https://github.com/ertis-research/opentwins`

Jupyter Notebook. Kafka receives the ML model once that it has been already defined and trained, and connects it to Ditto.

5 Experimentation

The current state of our Artax framework allows to define scenarios of people motion. So far, we have defined two scenarios that will be used for validating the forecasting functionality once it is finished. Right now, we are analyzing how people movements are performed in these simulations, so that they can be effectively used to simulate real people.

The first scenario consists of 400 nodes moving freely throughout the streets of a predefined urban area. This would represent an ordinary motion situation, meaning that large crowds might take place or not, as the nodes are moving randomly. A graphical visualization of this scenario is displayed in Figure 4, where we can see the map with streets through which people, represented by their ID, are moving.

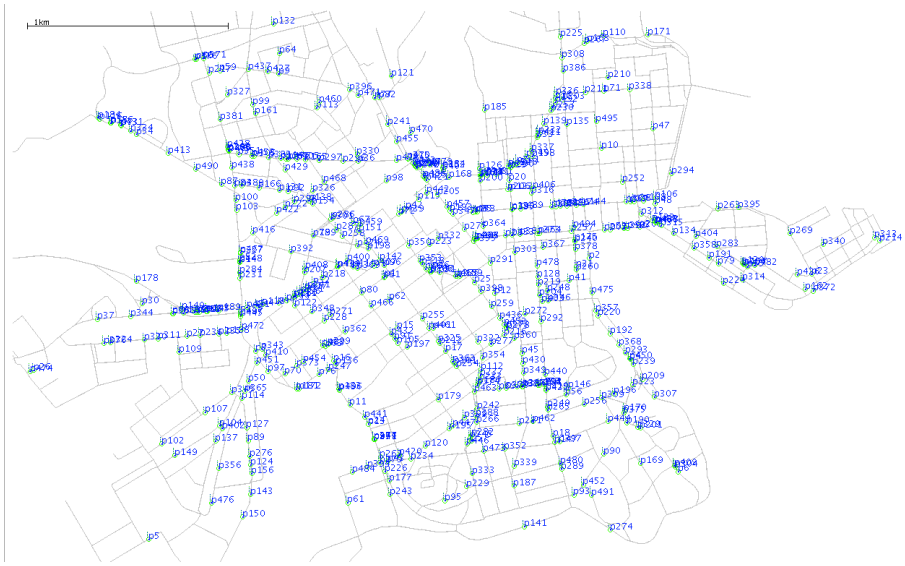

Fig. 4. Screenshot of the execution of the general test scenario.

The second scenario consists of 200 nodes moving in an extremely reduced space. Its main goal is to generate crowds. The nodes move along a triangle-shaped map, whose feasible paths are the edges of the figure—it is actually a

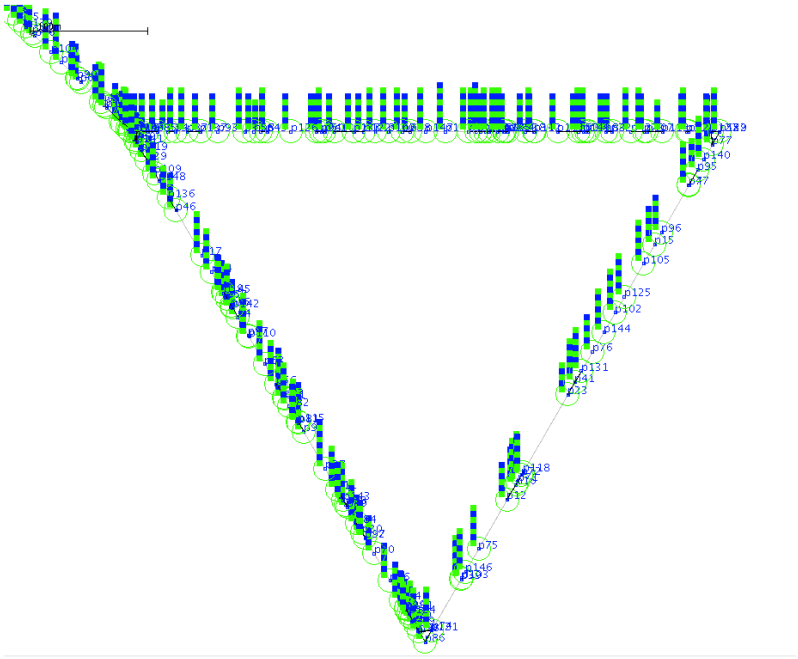

Fig. 5. Screenshot of the execution of the crowded test scenario.

rather simple scenario. This scenario can be seen in Figure 5, where we can see that crowds are already taking place.

6 Conclusions

This paper has presented the ongoing work centered on the *Artax* framework, which is built as a Urban Digital Twin that focuses on people motion. We have described the architecture and implementation of this framework, whose ultimate goal is to monitor people motion and detect potentially dangerous situations before they take place, so that people can be warned. The interface with which people interact with our framework is their own smartphones. We have described an architecture that considers both real people being tracked through their smartphones and a simulated environment where artificial nodes represent people movements.

As future work, we want to keep working on the backed part of our framework. In particular, we aim to focus on the machine learning component that will predict dangerous situations caused by crowds, and we will develop an alerting system to notify users when they might be in danger and to give them recommendations on how to act (such as taking an alternative path).

Finally, although challenging to put into practice, we would like to perform experiments with real people moving in a real space. For this, we might think of

some reward to give to the people who will be participating in the experiments. Their task is actually simple: they would simply need to install our Artax app in their smartphones and walk in an open space following some movement patterns that we will dictate.

Acknowledgments. This work has been partially funded by the Spanish Ministry of Science, Innovation, and Universities (projects PID2021-125527NB-I00, TED2021-130666B-I00, and TED2021-130523B-I00).

References

1. Azabal, M.J., Berrocal, J., Soares, V.N.G.J., García-Alonso, J., Galán-Jiménez, J.: A self-sustainable opportunistic solution for emergency detection in ageing people living in rural areas. Wirel. Networks **29**(5), 2353–2370 (2023). https://doi.org/10.1007/S11276-023-03294-9, https://doi.org/10.1007/s11276-023-03294-9
2. Batty, M., Axhausen, K., Giannotti, F., et al.: Smart cities of the future. The European Physical Journal Special Topics **214**, 481—518 (2012). https://doi.org/10.1140/epjst/e2012-01703-3
3. Bein Fahlander, L., Mossberg, M.: Simulating people flow at an airport: Case study: Arlanda airport (2020)
4. Bunde, A., Kropp, J., Schellnhuber, H.J., Helbing, D., Farkas, I.J., Vicsek, T.: Crowd disasters and simulation of panic situations. The science of disasters: Climate disruptions, heart attacks, and market crashes pp. 330–350 (2002)
5. Deng, T., Zhang, K., Shen, Z.J.M.: A systematic review of a digital twin city: A new pattern of urban governance toward smart cities. MSC Journal **6**(2), 125–134 (2021)
6. Guillén, J., Miranda, J., Berrocal, J., García-Alonso, J., Murillo, J.M., Canal, C.: People as a Service: A Mobile-centric Model for Providing Collective Sociological Profiles. IEEE Softw. **31**(2), 48–53 (2014). https://doi.org/10.1109/MS.2013.140
7. Herrera, J.L., Chen, H.Y., Berrocal, J., Murillo, J.M., Julien, C.: Context-aware privacy-preserving access control for mobile computing. Pervasive and Mobile Computing **87**, 101725 (2022). https://doi.org/https://doi.org/10.1016/j.pmcj.2022.101725, https://www.sciencedirect.com/science/article/pii/S1574119222001389
8. Jesús-Azabal, M., Herrera, J.L., Laso, S., Galán-Jiménez, J.: Oppnets and rural areas: An opportunistic solution for remote communications. Wireless Communications and Mobile Computing **2021**(1), 8883501 (2021). https://doi.org/https://doi.org/10.1155/2021/8883501, https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/8883501
9. Keränen, A., Kärkkäinen, T., Ott, J.: Simulating mobility and dtns with the ONE (invited paper). J. Commun. **5**(2), 92–105 (2010). https://doi.org/10.4304/JCM.5.2.92-105, https://doi.org/10.4304/jcm.5.2.92-105
10. Keränen, A., Ott, J., Kärkkäinen, T.: The ONE Simulator for DTN Protocol Evaluation. In: SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques. ICST, New York, NY, USA (2009)
11. Laso, S., Berrocal, J., García-Alonso, J., Canal, C., Manuel Murillo, J.: Human microservices: A framework for turning humans into service providers. Software: Practice and Experience **51**(9), 1910–1935 (2021)

12. Lehtola, V.V., Koeva, M., Elberink, S.O., Raposo, P., Virtanen, J.P., Vahdatikhaki, F., Borsci, S.: Digital twin of a city: Review of technology serving city needs. International Journal of Applied Earth Observation and Geoinformation **114**, 102915 (2022)
13. Lemercier, S., Jelic, A., Kulpa, R., Hua, J., Fehrenbach, J., Degond, P., Appert-Rolland, C., Donikian, S., Pettré, J.: Realistic following behaviors for crowd simulation. In: Computer Graphics Forum. vol. 31, pp. 489–498. Wiley Online Library (2012)
14. Madni, A.M., Madni, C.C., Lucero, S.D.: Leveraging Digital Twin Technology in Model-Based Systems Engineering. Systems **7**(1), 7 (2019)
15. Mangas, S.L., Olmeda, J.J.B., Fernández, P., Cortés, A.R., Rodríguez, J.M.M.: Perses: A framework for the continuous evaluation of the QoS of distributed mobile applications. In: Pervasive and Mobile Computing (2022)
16. Moreno, N., Toro-Gálvez, L., Troya-Castilla, J., Canal-Velasco, J.C., et al.: Modeling urban digital twins over the cloud-to-thing continuum. In: Mess Workshop @ STAF 2023 (2023)
17. Pérez-Vereda, A., Cabañero, L., Moreno, N., Hervás, R., Canal, C.: Distributed crowdsensing based on mobile personal data stores. In: Bravo, J., Urzáiz, G. (eds.) Proceedings of the 15th International Conference on Ubiquitous Computing & Ambient Intelligence (UCAmI 2023) - Volume 3, Riviera Maya, Mexico, 28-29 November, 2023. Lecture Notes in Networks and Systems, vol. 841, pp. 3–15. Springer (2023). https://doi.org/10.1007/978-3-031-48590-9_1, https://doi.org/10.1007/978-3-031-48590-9_1
18. Pérez-Vereda, A., Hervás, R., Canal, C.: Digital avatars: A programming framework for personalized human interactions through virtual profiles. Pervasive Mob. Comput. **87**, 101718 (2022). https://doi.org/10.1016/J.PMCJ.2022.101718, https://doi.org/10.1016/j.pmcj.2022.101718
19. Pérez-Vereda, A., Murillo, J.M., Canal, C.: Dynamically programmable virtual profiles as a service. In: 2019 IEEE SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI. pp. 1789–1794 (2019)
20. Robles, J., Martín, C., Díaz, M.: OpenTwins: An open-source framework for the development of next-gen compositional digital twins. In: Computers in Industry (2023)
21. Ruiz, P., Seredynski, M., Torné, Á., Dorronsoro, B.: A Digital Twin for Bus Operation in Public Urban Transportation Systems. In: Big Data Intelligence and Computing. pp. 40–52 (2023)
22. Saeed, R.A., Recupero, D.R., Remagnino, P.: Simulating people dynamics. In: 2021 17th International Conference on Intelligent Environments (IE). pp. 1–8 (2021). https://doi.org/10.1109/IE51775.2021.9486478
23. Soomaroo, L., Murray, V.: Disasters at mass gatherings: lessons from history. PLoS currents **4** (2012)
24. Wang, H., Chen, X., Jia, F., Cheng, X.: Digital twin-supported smart city: Status, challenges and future research directions. Expert Systems with Applications **217**, 119531 (2023). https://doi.org/https://doi.org/10.1016/j.eswa.2023.119531, https://www.sciencedirect.com/science/article/pii/S0957417423000325