World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

Design and implementation of an offline-first road crash data collection system

Awa Tiam, Ibrahima Gueye *, Ahmed-Mouhamadou Wade, Samba Sidibe and Oumar Niang

LTISI laboratory GIT, Polytechnic School of Thiès (EPT), Thiès, Sénégal.

World Journal of Advanced Research and Reviews, 2022, 13(02), 095–107

Publication history: Received on 01 January 2022; revised on 05 February 2022; accepted on 07 February 2022

Article DOI: https://doi.org/10.30574/wjarr.2022.13.2.0129

Abstract

Road traffic accidents (RTAs) are a global burden that particularly affects people in developing countries. According to the World Health Organization, having reliable and precise data is necessary to raise the alert on the scale of road accidents and convince decision-makers of the need to take action. Each year, nearly 700 people lose their lives due to accidents on Senegalese roads. In Senegal, dis-aggregated data related to accidents are rare due to very significant under-recording. Most of these data are in a non-computerized form and their collection methods differ from one side to the other among road safety actors. The task of collecting data is simply considered of secondary importance for certain road safety actors. The disparity between the collection methods as well as collected data and the non computerized collection processes limit the possibility of having a real knowledge of the phenomenon of road accidents.

In this work, we present a global solution of a road crash data collection and management system. In particular, we will focus here on the collection component with an application that aims to federate data collection processes of all road safety actors in Senegal. It uses a collection guide designed after a review of road accident data guides in Senegal and in other regions of the world. It also implements a parallel collection methodology, i.e the possibility of several agents working on collecting data for the same accident.

Keywords: Offline-first; NoSQL Databases; Crash Data Collection; Crash Data Stores; Big Data

1. Introduction

Nowadays, road crash data systems are crucial for many reasons. According to the World Health Organization, such a system is mandatory to produce correct and accurate data in order to advocate for road safety by making explicit the need and necessity for road safety actions. Data systems are also helpful for identifying the problems and risks inherent in road safety, setting road safety targets, formulating appropriate strategies to promote road safety and monitoring the impacts of road safety actions.

However, the data collected every day in most countries may not achieve these objectives unless they respect some important criteria. - Indeed, these data have to be properly coded : the coding having for objective to standardize the modalities of writing and reading of these data ; - they have to be entered in a system, preferably computerized one ; - properly processed : the processing having for objectives to decrease the noise and to standardize the formats of the data to make their analysis easier ; - these collected data has, finally, to be analyzed and disseminated ; the analysis step consists in extracting knowledge from the data and dissemination is for informative purpose and helping for decision making. The costs of road traffic accidents are estimated at nearly 1% of Senegal's Gross Domestic Product. Each year, nearly 700 people die from road accidents in Senegal. The police, who are authorized to act in crashes occurring in urban areas, use the BAAC data system, whose collection module consists of a printed form that must be filled out manually by the officer at the accident site. The Firefighters, which are the actors that can act either in or out of urban

* Corresponding author: Ibrahima Gueye
LTISI laboratory GIT, Polytechnic School of Thiès (EPT), Thiès, Sénégal.

Copyright © 2022 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution Liscense 4.0.

areas, also have a form called Assistance Report to be filled in manually after intervention on site. Beyond the fact that these data collection forms (from police and Firefighters) are processed manually and stored in ad-hoc systems, their content differs from one source to another due to the lack of standardization. The most significant problem according to some safety actors in Senegal is related to the collection of accident locations.

In this work, we propose a unique collection form that unifies every existing form in the country and we provide a mobile application for digitized collection mode. The collection form we propose was designed after a review of road accident data guides in Senegal and in other regions of the world. Furthermore, we propose an entire system that allows collecting data from road accidents through mobile devices and automatically stored those collected data on a remote and available storage infrastructure. The collection system can work autonomously and synchronize to the remote storage system as soon as the Internet connection becomes available. Our solution also implements the notion of parallel collection to accelerate the on-site data entry. In fact, when many agents have to collect data from the same site accident, they can do it in parallel, and our algorithms handle the synchronization and consistency of data.

2. Background

In Senegal, the Police and the Firefighters are the actors authorized to respond to road accidents. In the course of our work, we have been in contact with these different actors. Data collection process is different on each side and remains manual for the different sides. For the firefighters, a form called Assistance Report is used ; this form is filled in after intervention on site. The Police use a form adapted from the BAAC [1]. Data to be collected are different from one actor to another, they are in fact linked to the mission of the different actors. In addition to this disparity in collection methods and objectives, there is manual data entry. This manual entry poses a number of problems. First, there is time required to inform data elements. Second, errors due to human accuracy can affect data quality. Manual data collection also requires an additional step of data entry for those actors who have a computerized database to store the data, particularly the Police. Post-intervention data entry poses also some problems: very important information is only available on-site and for a limited time. Road conditions or traffic conditions are examples of point-in-time elements that need to be collected on-site in order to be accurate enough.

Data collection is of secondary importance for some actors such as the Firefighters because of their life-saving mission. The task of data collection is therefore put to a secondary level. The disparity between the collection methods, collected data, and the lack of computerization of the collection processes limit the possibility of having a real knowledge of the road accident phenomenon at a national level. A computerized tool to federate data collection needs of the various actors and to accelerate the data collection process would be of significant help.

In this collection of road crash data, the materialization of the on-site collected data in a storage center is managed in different ways. In the case of a manual collection, the actual situation in Senegal, data are moved as-is to a central area. At this point, data is entered into a computerized database in the case of Police, or compiled in paper form for Firefighters.

In the case of a computerized collection, the mobile terminals do not have enough space for long-term recording, so data cannot be stored indefinitely. Accident data collected from a computerized terminal is therefore commonly transferred to a central database not located on the mobile terminal. In this routing process, the issue of connectivity loss between the mobile device and the final database is critical. Mobile data synchronization refers to the materialization of data entered from the mobile terminal interfaces to the storage databases. The synchronization of mobile data presents challenges due to the risks associated with connectivity. These challenges include ensuring data integrity.

When storing crash data in the central database, data is commonly stored in relational databases. The relational model assumes an established, rigid schema whose modification requires changes that can lead to data corruption and database unavailability. A crash is the result of a combination of different events.

The description of a crash should therefore not be fixed. In other words, Data or information required to describe or understand a crash called incident1 may be different from that required to describe or understand a crash called incident2. Even though some information may be common and included in what are commonly called data collection guides (MMUCC, CADaS, BAAC, ...). Data to be considered to understand a crash can change. Therefore, the crash databases must be flexible and not provided with a rigid schema. They must be able to store and make available the largest kinds of data.

3. Related Work

3.1. Crash data collection

The reviewed papers [2] [3] [4] suggest the use of mobile applications to quickly collect as many elements as possible on site. Mobile applications have the advantage of simplifying the mobility of the officers in charge of collecting data. These applications are installed on devices outside the vehicles, most of the time on tablets or smartphones. The Traffic and Criminal Software (TraCS) system of Indiana [5] [6] includes an application installed on the internal equipment of the officers' vehicles. It recommends automatic filling of fields by using bar code readers or GPS to speed up data collection. Any field that can be filled can help speed up collection. The application uses field logic, certain data available on a specific field and required by other fields are made available automatically through the application.

The MMUCC [7] proposes to remove the notion of source. States are encouraged to retrieve data from a variety of possible sources. In 1999, the use of expert systems was experimented to help ensure the completeness and validation of the input data related to seat belt usage, vehicle damage assessment and deformation extent, and road barrier identification through the collection tools [8]. To handle errors and other anomalies inherent in data collection, TraCS [11] uses a system of business rules for validation of data entered through the collection interface. These business rules can include checks on other related databases such as vehicle identification databases, roads, etc. but also checks on the weather data side for example. The BAAC [1] is a software intended to manage crash data from collection to analysis phase. The collection step is handled by a standardized form filled by the officers. These filled forms are then used to upload data with an intermediate data validation step. This form is made up of two sections. The first section is related to crash characteristics (location, road characteristics, weather conditions, ...).The last section is linked to vehicle related information (vehicle make, person involved, ...). The location of the crash has been identified as an important piece of information aiming to refine analysis on road crash data as they allow to conduct geographical analysis on data. Montella et al. suggests the use of phone's geolocation sensors to get GPS data. In the BAAC software, locations are given by the road name, road identification number and the road milepost.

3.2. Crash data load

Given the possible loss of connectivity between the collection interfaces and the central storage, ensuring the integrity of data stored in the central database is mandatory. Data transmission from the collection interfaces to the central storage point is often managed through the use of web services. Most studied papers are working fully online. They often use PHP/JAVA scripts along with an HTTP server to materialize data collected from mobile interfaces to a central database [4] [9] [3]. Ozianzi and Kenga [4] have proposed a MySQL database as a central database server. Data entered from the mobile collection interface is stored in the database using Web Services from an HTTP Web server. Thus, the only point of backup for Data is the central database. This process is subject to the risks of internet connectivity loss, however there is no documented way to manage the case of a break in connectivity between the mobile terminal and the central database.

Montella et al. [9] and Campisi et al. [3] have proposed the same kind of architecture when designing a web application to collect crash data. The ACID properties of relational databases protect them from operations that can corrupt data. This means that unsuccessful transactions will not be materialized in the database. We can therefore deduce that for applications that do not have an intermediate storage capacity, the transactions in progress during a loss of connectivity will simply be abandoned. It is therefore logically up to the application to set up procedures for recovery after a failure.

The RADaR application [2] stores Data entered via the collection interfaces in the memory of the mobile terminal. There can be up to 10 instances of accidents recorded on the mobile terminal before being sent to the central server hosted on a Web server. While this technique allows keeping data in safety in case of connectivity loss, the memory usage can be a bottleneck when the amount or the kind of data become non trivial. The only way to manage this problem will be to increase the memory capacity of mobile terminals.

Data collected through the BAAC [1] standardized forms are simply sent as-is to a central office, often at a Police station, in order to be uploaded in a computerized database.

3.3. Crashes data stores

The studied papers [3] [4] [10] [6] [9] use relational databases to store crash related data. There used to be four or more entities in the database intended to store crash data. Generally, there are crash general characteristics(weather conditions, locations, ...), vehicle related information, person related information and road related information. In respect to the relational paradigm, the relations between these different entities are also stored in different tables.

Montella et al. [9] uses 19 tables with 280 variables to materialize data entered through the collection interface in an MySQL relational database.

Khan et al. [10] propose a GIS database combined with a relational database to store location-based information. While this technique allows a specialized database to store this kind of information, a more global database can effectively store this type of information and reduce the costs associated with managing two or more databases. The BAAC [1] software uses a relational database based on Microsoft Access.'

4. System Design and Specifications

This paper presents the design and implementation of a road crash data collection system. This is part of a global system that means to design and implement a road crash data system for Senegal. It implements an offline-first on-site road crash data collection application for Senegalese road safety actors. The objectives were to overcome traditional crash data collection systems issues : manual collection, manual data entry in crash databases, systematic use of the internet for on-site mobile crash data synchronization and rigid modeling of crash databases. The system explores the use of a distributed NoSQL database to collect and store road crash related data instead of the traditionally used relational databases. Given the nature of the accident phenomenon, NoSQL technologies can be brought in such a way to originally support the evolution of Data to be collected and to reduce the collection delays due to the relations and interconnections between the different entities defining an accident.

4.1. Terminology

4.1.1. Offline-first

The term offline-first is used in this article to describe those mobile applications or web applications that offer capabilities to be used without Internet connectivity by their users.

4.1.2. NoSQL database

The term NoSQL database is used here to refer to a database where stored data is not tied to a rigid schema, which means that evolution in data components is handled by the database without need of re-configuration.

4.1.3. Android ViewModel

On Android, the ViewModel mechanism allows to share elements between different fragments or activities and to ensure that even if the activity is stopped by another process (call, lack of resources, ...), the important elements of the workflow are not lost.

4.2. Software architecture

The architecture of the collection system (Fig. 1) is composed of three parts : — a mobile crash data collection application (*JAR*) — a synchronization component — a storage database. The mobile application allows to collect data on-site even without the Internet connectivity. Collected data is first stored in a local database. With Internet connectivity, data is synchronized to a remote database. This remote database offers high availability and fault tolerance capabilities.

4.3. Just Another Recorder

JAR, or Just Another Recorder, is a mobile application running on phones emulating the Android operating system. It has been coded entirely using the Kotlin language; Its purpose is to allow the collection of road accident data as described in the collection guide. The location is very significant in this collection system. Indeed, its information allows us to refine the analyses made on the road accident data. The architecture of the mobile application is made to allow the use of the application in offline mode. Actually, Data entered in the different sections are recorded locally in a database embedded on the mobile terminal. When internet connectivity is restored, the replication mechanism between the embedded database and the central database takes place, and the local data is replicated on the central database. The replication in distributed systems allows us to ensure high availability of data. In the event that one of the cluster nodes becomes unavailable, Data is still accessible due to the replication. To speed up the collection process, multiple agents can work on collecting data for the same accidents.

System requirements:

- The application is intended to be used by officers (Police, Firefighters, ...)
- Android operating system based smartphone
- GPS enabled: for location retrieval purposes. No internet access required.
- Bluetooth enabled : for parallel collection

4.4. Implementation details

There are four views on the collection form corresponding to the four fields of the referential : Accident, Vehicle, Person and Road. In code logic, these fields are managed by the Android fragment mechanism. The accident identifier is deduced directly from the fields including the location, the date, the time and the place (region, department). By the mechanism of the Android ViewModel, this identifier is made available to the other fragment, making it possible to link the various entries of the rubrics.

An accident instance occurs on a road and may or not involve one to several vehicles and one to several persons. In the application, the possibility is given to translate this situation by adding vehicles and persons if necessary. In the logic, this is translated by the concept of Android template.A template is a sequence of reusable code. This fragment template is added on demand to the existing one at the activity level. The saving of the different views of the application is made independently. This operation is justified by the fact that different people can fill in the different views of the same accident in parallel.

4.4.1. Crash location

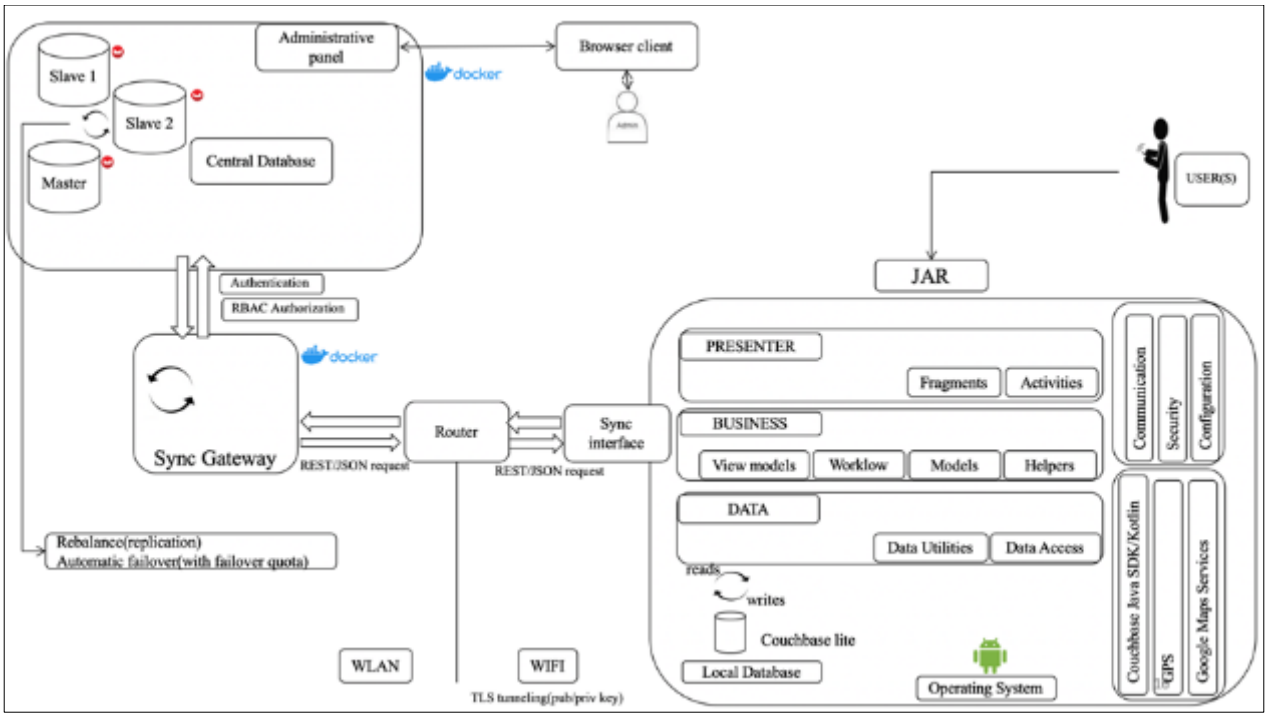

Figure 1 Architecture of collection module

Within the mobile application, location acquisition is based on the notion of providers. There are mainly two providers: Internet and GPS. The GPS gives the possibility to retrieve the location even in places without internet connectivity. The application has the possibility to switch providers for location retrieval. But the application retrieves the location preferably through GPS. Location updates are requests made by the application to get the location changes of the cell phone at regular times. Due to the importance of this location, the update requests are made every two seconds. In addition to GPS coordinates, the road name in respect to Senegalese road classification [12] is also added.

4.4.2. Local database

The local database is an instance of Couchbase Lite embedded in the mobile device. An instance is created locally when the mobile application is launched on a smartphone. There is no notion of a connection or user session embedded in the mobile application. Because of the replication, Data located in the central database is also present in the local database.

4.5. Parallel collection

The collection guide elements were selected following a comparative methodology between different collection guides in use and taking into account the recommendations of international organizations, in particular the WHO, concerning the road accident data collection. The collection interface *JAR* offers a visualization of this guide. But it should be noted that there is a significant number of data elements to collect. This can be a time-consuming task for a single officer, which is not aligned with our idea of using the application. Indeed, the application aims in part to alleviate the manual collection process and time spent to collect data. To overcome this, we have introduced the notion of parallel collection. As its name indicates, this collection aims at paralyzing the process of accident data collection so that several agents can work on this process.

From a technical point of view, the following considerations are required:

- The crash identifier is deduced by the application : in the crash tab, the accident identifier is automatically deducted from data entered by the user. This process is done when the user validates the crash form tab.
- Thanks to the structure of the database, the saving of the different sections of the form can be performed individually except for the crash des- ription tab which need to be filled in order to get crash identifier. This gives the ability to several persons to work on the collection of data elements of the same crash.
- The crash identifier must be unique in the database as it is used as a key for all crash documents.
- The vehicles and persons involved in the crash as well as the road characteristics are identified and linked to the crash instance by the identifier. This identifier deduced at the level of the first field of the form is made available to the other fields by the ViewModel mechanism.

It appears that having the crash identifier between several officers can ensure parallelism.

4.5.1. Data sharing technologies

In Android, different technologies exist to ensure data transfer between mobile phones. WIFI-Direct [13] or WIFI P2P is a technology that allows data to be transmitted between terminals via WiFi without an intermediate access point such as a modem. It does not require internet connectivity. However, this system requires authentication between the different terminals with a group owner. The latter is chosen from among the terminals after creating a group. The Bluetooth [14] technology is commonly used to share data between multiple mobile devices. It is also very widespread among cell phone brands. The WIFI-Aware is a technology similar to WIFI-P2P except that it does not require the authentication protocol of the group owner which does not exist in this implementation. However, Data that can be transferred between the terminals is less voluminous. For the transfer of heavy data, it is possible to initiate a network with the notion of group owner. The speed of information transfer is faster via WIFI than Bluetooth, which makes it effective for the transfer of heavy information. The energy consumption of WIFI-Direct technology is 40 times higher than the Bluetooth one. Both technologies support service discovery.

4.5.2. Implementation of the concept of parallelism:

The following considerations were taken into account when selecting technologies for data transfer

Data to be transferred is small, less than 255 characters. It consists of the crash identifier only. — Data transfer must be able to be done offline — The members of a team of collectors are theoretically located on the same site, the sharing must not take a long time, the notion of coverage is neglected in the process.

4.5.3. Actions sequence

Team leader informs data elements of the tab "Accident". — The team leader validates data elements of the tab "Accident". — The application generates the accident identifier from the location, the names of the cities and departments and the time. — The accident identifier is recorded in the ViewModel of the collection page — The team leader initiates the sharing of the crash identifier by clicking the corresponding button in the Accident section. This button is only used in the case of parallel collection. — The application displays a list of terminals (name, MAC address) recognized by the terminal used by the team leader and sharing the same UUID. The latter is specific to the software.

This means that all the terminals running the application, recognized by the terminal and in the Bluetooth coverage area are displayed on the team leader's screen. The concept of recognition refers to the fact that there has already been a connection between the displayed terminals and the team leader's terminal. The process of discovering new terminals is not implemented because we would end up with a potentially large number of terminals that are not interesting to display — The team leader selects the terminal(s) individually. The accident identifier is sent to each terminal — The accident identifier is automatically filled in the appropriate field on the receiving terminal.

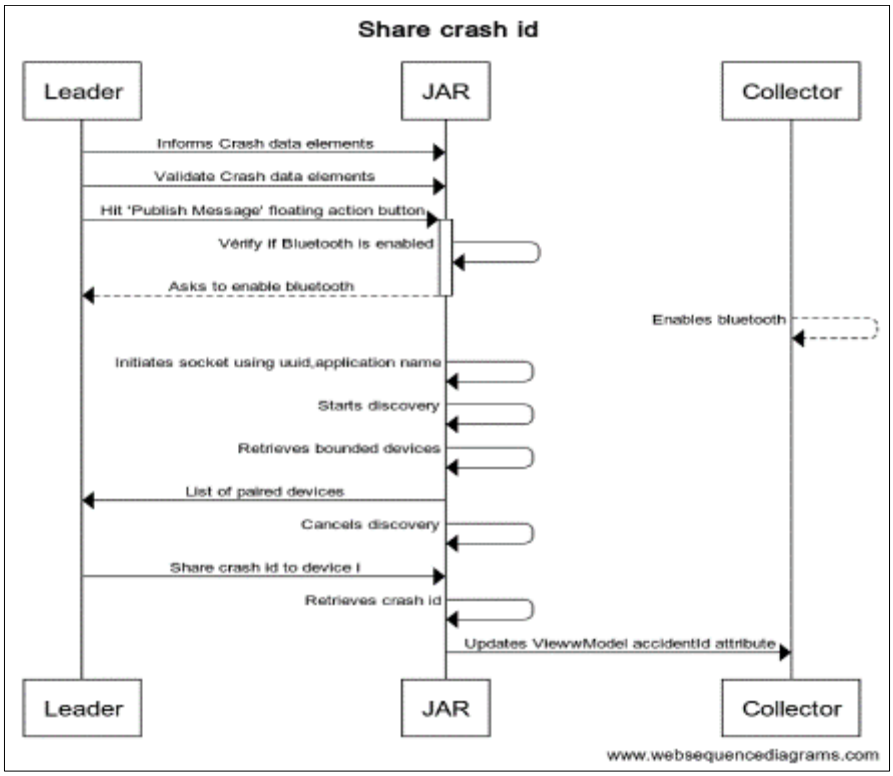

Figure 2 Sharing crash identifier

4.6. Synchronization

Synchronization in this system is simply a way to save entered data through the mobile application in the central database, taking into account the risks associated with connectivity. Synchronization is triggered by Internet connectivity and allows us to ensure offline-first methodology in this system.

4.6.1. Replication protocol

The replication protocol is based on web sockets. Replication is an asynchronous operation and its purpose is to send documents from a source to a destination. The destination of any replication action at the system level is the synchronization server Sync Gateway. The source is the local database embedded in the mobile device. The communication at the replication component is secured by using the TLS protocol with private key sharing. The synchronization server is configured with public and private key files. The client, here the mobile application, authenticates itself by:

- the private key integrated to the mobile application
- the user: mobile application authenticates itself to the synchronization server with a username and password. At the first connection, the synchronization server creates a cookie that will be used in all subsequent requests during the entire replication procedure.

Replication offers two types of services:

- push: send non-replicated documents from the local database to the central database.

- pull: recover the documents from the central database not replicated at this instance of the local database. This happens in the case where other clients send data to the central database in the same bucket. As we don't need to retrieve the documents located in the central database at mobile level, the replication is just configured with push service.

4.6.2. Connection to database:

The synchronization component authenticates itself to the database by using credentials (username, password) created at the database setup.

4.7. Data storage

When dealing with crash data stores, one has to keep in mind the nature of the phenomenon of road crash. It often depends on multiple factors. These factors can be different from a crash to another. Urban mobility and vehicle characteristics are changing very quickly. It is mandatory to have the possibility to add additional fields that would be necessary for future analysis without having to re-develop the whole database afterwards.

4.7.1. Database infrastructure

The database (Fig.3) is a cluster built with three node machines, each emulating an instance of Couchbase server. Each machine runs as a docker container. The latest Couchbase server Community edition docker image is used ; The three machines are linked in a network. Data is stored in a bucket ; each bucket contains virtual buckets named vbuckets. Those physical buckets exist on memory and disk. For testing purposes, all three nodes run in one physical host. In a production environment, however, each of these nodes need to run in a separate physical host to ensure high availability and fault tolerance semantics. The database supports re-balancing of data. In our three nodes cluster, data is stored in one node namely active node and copied in the two others nodes called replicas. The objective of re-balancing is to ensure the availability of data in case of node(s) unavailability. The system can support up to two nodes being down at a time, this ensures fault tolerance. When a node becomes unavailable, cluster manager (here master node) is informed to read/write data on those nodes that are up.

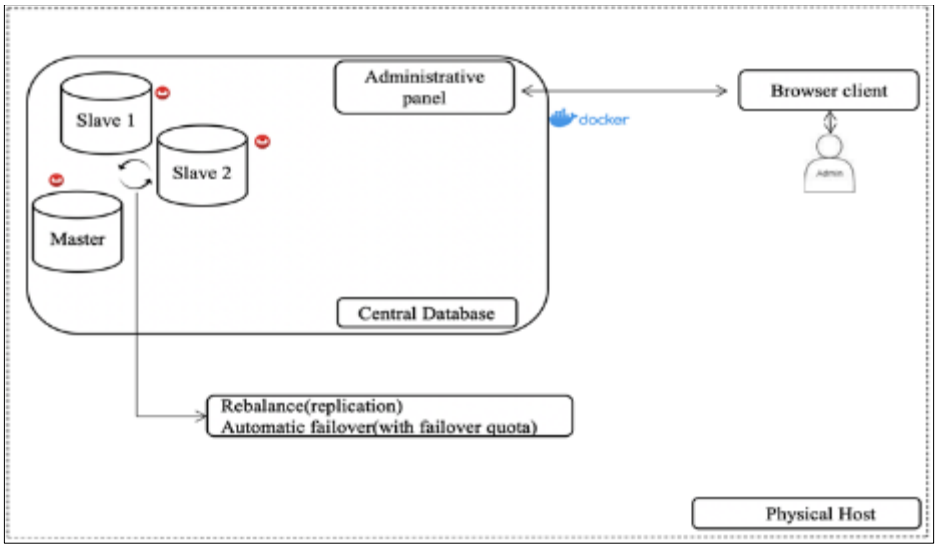

Figure 3 Central database

4.7.2. Data model

Each document has a key used to uniquely identify it in the database. We have four types of documents describing the four major entities of our data model. That model was built using a comparative study on data elements between the Senegalese version of BAAC and other collection guides data elements: Model Minimum Uniform Crash Criteria(MMUCC), Common Accident Data Set(CADaS), WHO minimum data elements. Data model is based on a JSON format. We have four main entities in the database namely : - Crash ; - Vehicle ; - Road and - Person. Figure 4 shows a simplified view of the database. This visualization shows relations between different document types in the database.

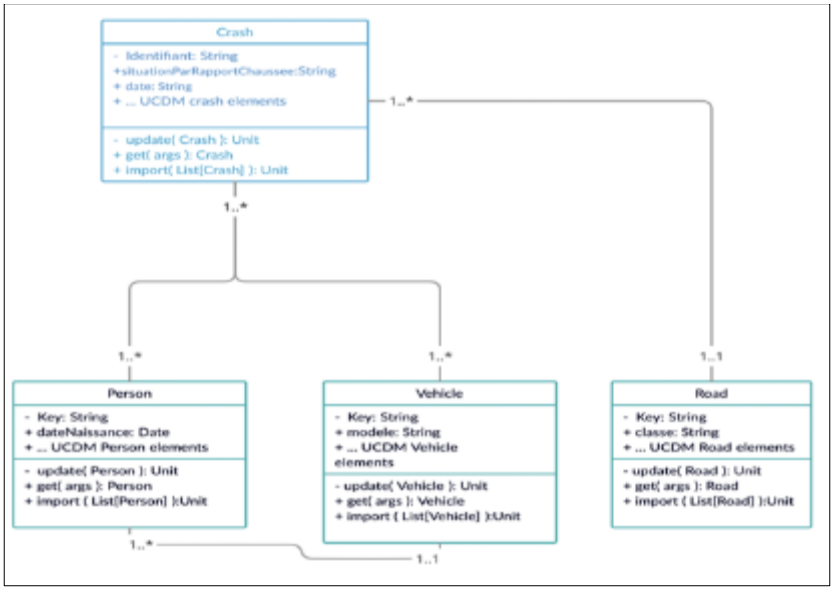

Figure 4 Relations between documents

Document modeling

The table 1 details the different document identifiers. Each entity of the database is translated into a JSON document. Each document is uniquely identified in the database and includes a key that provides information on its typology. The accident is identified by its unique identifier which is built from the location, date and time of the accident. The road is identified by the identifier of the accident to which we prefix the string ROAD. Since there can be several vehicles, each of them is identified by the accident identifier as well as the sequential vehicle number used to uniquely identify a vehicle involved in a given accident. The same principle of vehicle identification is also applied to persons. Every person document also contains a key identifying the vehicle involved in the case of a driver and passenger or pedestrian hit by a vehicle. Each person has also a sequential number identifying it uniquely.

Table 1 Documents identifiers

Key	Document key	Typology
CrashID	ACC_region_department_date_timestamp	Crash
VehicleID	Vehicle_CrashID_sequentialNumbe	Vehicle
RoadID	ROAD_CrashID	Road
PersonID	Person_CrashID_vehicleSequentialNumber_sequentialNumbe	Person

Durability

It is managed at the client level. It refers to the fact that a document write operation is considered successful only when this specific document is written at a certain number of cluster nodes in memory and/or on disk. The durability of data written through JAR has a majority of two nodes. Data is considered written only when it is available on two nodes of the cluster and on disk.

Conflicts resolution

The application is intended to be used such that updates and replication conflicts are minimized. The replication process concerns *push* only replication i.e data is replicated from mobile to central server only. The mobile application is not intended to handle updates or deletes on existing data. In fact, the application is used only to collect new data. In a parallel collection logic, each officer has a specific role and each of them must be given a specific data collection task to manage. Although, in certain scenarios, we could end up managing updates when:

the officer saves unintentionally tab information more than one time. In this case, the two documents to be written have the same key and the same data. Here we have the same instance of the application and the same instance of the mobile database. In this case, the *Last Writes Wins* algorithm is used. The document is updated in local and remote databases if already synchronized; — the user resubmits a new version of the same document with modifications on certain fields. This is intentional and means to correct already entered data. Here also, the *Last Writes Wins* is applied on the local database. The document is updated in a remote database if already synchronized; — due to a coordination problem, two different officers end up collecting the same information in a parallel collection logic. First, data is stored in their respective local databases. Here we have different instances of the mobile application, thus different replicators, and the same keys. An error is displayed with a *document that already exists exception*. The last writes will not be considered; — an officer tries to submit data for an already collected crash data. Given the way that keys are constructed; they include locations, date and timestamp; the documents will not have the same key and so, there are no conflicts to resolve but we will logically end up with duplicates entries in the database.

The way to handle those conflicts are depicted on Figure 5.

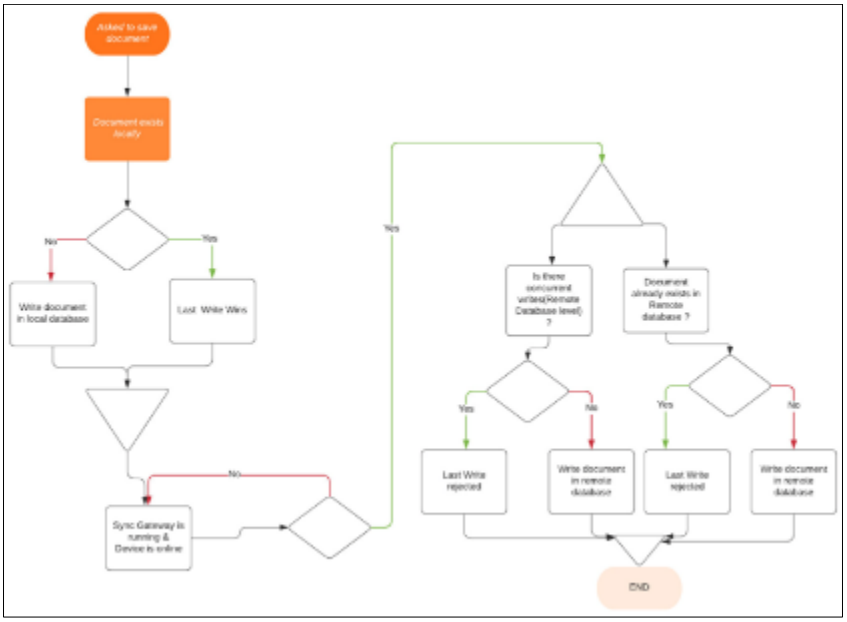

Figure 5 Conflicts resolution

5. Testing

The testing of the collection system was carried out in the form of a Proof Of Concept. Table 2 details hardwares and softwares used for testing purposes. Table 3 details the steps taken to test the collection system along with obtained results. All specified tests stages have been conducted using the specified hardwares and softwares. However, the test on the reception of the message in the case of a parallel collection is not yet effective; that's why the test has been marked *In progress*.

Table 2 Hardwares and softwares

Usage	Version	Properties
Mobile application deployment	Huawei Nova 3i	Android 8.1 (Oreo) upgradable to Android 9
Synchronization server	macOS Mojave 10.14.6	Processor 2,2 GHz Intel core i7 Memory 16 Go 2400 MHz DDR4
Central Database	macOS Mojave 10.14.6	Processor 2,2 GHz Intel core i7 Memory 16 Go 2400 MHz DDR4
Containerization	Docker Desktop	Engine 19.03.13 Community Edition

Table 3 Test phases

Test stage	User story	Status	Result
Mobile application deployment		Done	OK
Visualizing crash, vehicle, person and road	User must be able to visualize the tabs	Done	OK
Scrolling	User must be able to scroll in tab and to scroll across tabs	Done	OK
Pre-filled values(Date, hour, locations)	Date, time, locations must be filled at tab visualization	Done	OK
Modification of pre-filled values	User must be able to modify pre-filled values	Done	OK
Departments	The department drop down must be filled depending on value in region drop down	Done	OK
Add persons and vehicles	User must be able to add persons or vehicles as needed	Done	OK
Drop down list choices	User must be able to choose in already filled list of values	Done	OK
Enabling services	User must be asked to enable Bluetooth and GPS	Done	OK
Display list of discovered devices	User must be able to see a list of discovered devices over Bluetooth	Done	OK
Choose a device and send message	The leader must be able to select a specific device and send a message	Done	OK
Receive message from leader	A collector has capability to receive crash identifier from the leader	In progress	No OK
Automatic filling of crash identifier	The crash identifier has to be filled automatically by the application	Done	OK
Get crash identifier	The crash identifier need to be available on other tabs after saving of crash	Done	OK
Save data offline	User must be able to save data to database without an internet connectivity	Done	OK
Save data online	User must be able to save data to database with an internet connectivity	Done	OK

6. Results and discussion

The entire process of collection and storage in diverse Internet situations was successfully tested. We were able, in our test process, to bring data from the mobile application to the database and finally to the analysis portal. Fig.6 shows the

main collection interfaces of *JAR*. Data collected through *JAR* and stored are lately used for analysis purposes. Fig.7 shows a spatial view of a crash (right) and an ad hoc analysis (left) of the data recorded through *JAR*.

For further improvements;

- It would be very interesting to be able to test the overall performance of this collection system against existing data collection systems in Senegal.
- The mobile application is intended to be used in smart phones with Android operating system. But this application would benefit from being cross-platform.
- The carried out tests were in the form of proof of concept in order to show the feasibility and the utility of such a system for the collection and the storage of crash data. To evaluate the system more effectively, tests in a real context are needed.

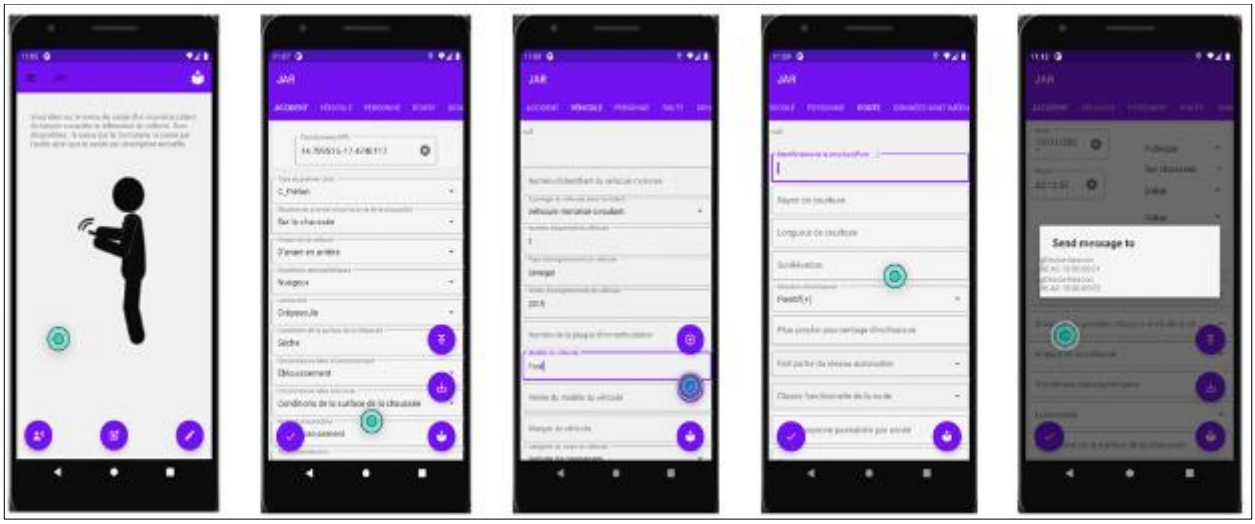

Figure 6 Collection interfaces

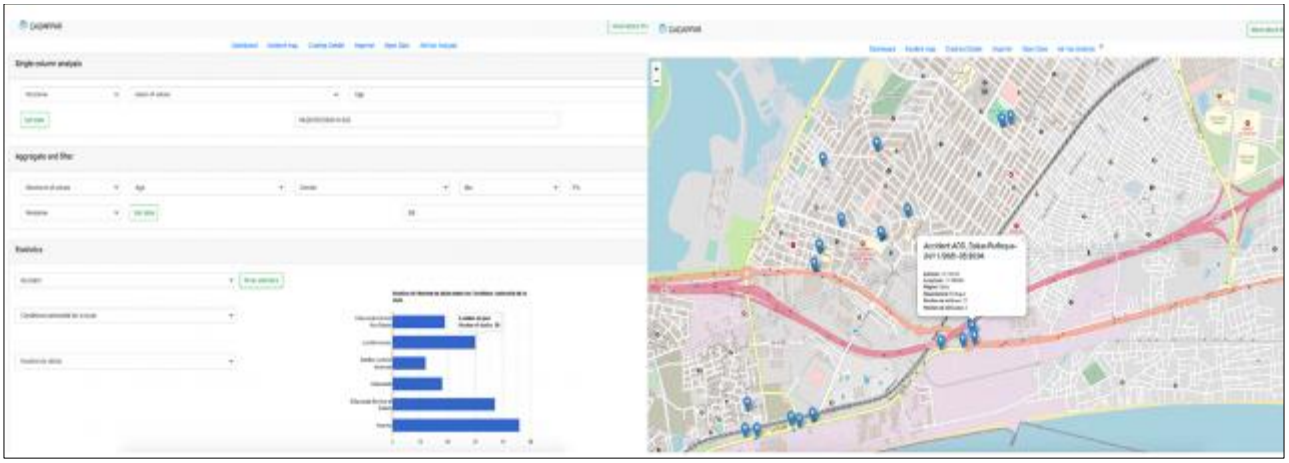

Figure 7 Spatial view of a crash(right) and ad hoc analysis (left) applied on collected data

7. Conclusion

The purpose of this work was to design and implement a road crash data collection system. It consists of a mobile application for on-site data collection and a NoSQL distributed database for data storage. The objectives were to overcome traditional crash data collection system issues : manual collection, manual data entry in crash databases, systematic use of the Internet for on-site mobile crash data synchronization and rigid modeling of crash databases.

This collection system and more globally the crash collection and analysis system aims to build a robust, modern and unique crash database shared among road safety actors, researchers and start-ups in Senegal.

Compliance with ethical standards

Disclosure of conflict of interest

The authors declare that there is no conflict of interests regarding the publication of this paper,

References

[1] WHO: Data systems – A road safety manual for decision-makers and practitioners. 2010; 95-96.

[2] Irf. global, International Road Federation. 2022.

[3] Campisi T, Galatioto F, Franco P, Barone R. A new approach for road accident data acquisition : The K_Road App. 2013.

[4] KM Derdus, VG Ozianyi. A mobile solution for road accident data collection. In: Proceedings of the 2nd Pan African International Conference on Science, Computing and Telecommunications. 2014; 115-120.

[5] Transportation Research Board of the National Academies,NCHRP Synthesis Technologies for improving safety data. A synthesis of highway practice. Transportation Research Board Publishing, Washington D.C. 2007.

[6] Ed Cherry, Rob Floyd, Tyson Graves, Steve Martin, David Ward, Crash Data Collection and Analysis System, Final Report 537; p. 1-91

[7] transportation.gov, Model Minimum Uniform Crash Criteria, Fourth Edition, 2012

[8] Carol Y. Thielman. Expert Systems for Crash Data Collection. Publication No.FHWA-RD-99-052. 1999.

[9] Montella A, Chiaradonna S, Criscuolo G, De Martino S. Development and evaluation of a web-based software for crash data collection, processing and analysis. Accident Analysis and Prevention. 2017.

[10] Khan M, Amer Al Kathairi, Abdulla Salim, Garib, Atef M. A GIS based traffic accident data collection, referencing and Analysis framework for Abu Dhabi. 2004.

[11] Miklos P, Gabor L. Mobile Data Synchronization methods. Hungarian Journal of Industry and Chemistry. 44(2): 93–98.

[12] Gouvernement du Senegal. Decret N°2012-1440 Portant classification du reseau routier national. 2012

[13] Google Developer Guide, Developer.android.com, Create P2P connections with Wi-Fi Direct. 2022

[14] Bluetooth.com, Introducing Bluetooth, 2022.