ICEB 2017 Proceedings | International Conference on Electronic Business

Winter 12-4-2017

Emergency Management System Design for Accurate Data: A Case Study

Abbas Tarhini
Department of Information Technology and Operations Management, Adnan Kassar School of Business, Lebanese American University, Beirut, Lebanon, abbas.tarhini@lau.edu.lb

Jordan F. Srour
Department of Information Technology and Operations Management, Adnan Kassar School of Business, Lebanese American University, Beirut, Lebanon, jordan.srour@lau.edu.lb

Nabil Georges Badr
Grenoble Graduate School of Business, Grenoble, France, nabil.badr@alumni.grenoble-em.com

Tarhini, A. & Srour, J.F. & Badr, N.G. (2016). Emergency management system design for accurate data: A case study. In *Proceedings of The 17th International Conference on Electronic Business* (pp. 93-98). ICEB, Dubai, UAE, December 4-8.

Emergency Management System Design for Accurate Data: A Case Study

Abbas Tarhini*, Department of Information Technology and Operations Management, Adnan Kassar School of Business, Lebanese American University, Beirut, Lebanon, abbas.tarhini@lau.edu.lb
F. Jordan Srour, Department of Information Technology and Operations Management, Adnan Kassar School of Business, Lebanese American University, Beirut, Lebanon, jordan.srour@lau.edu.lb
Nabil Georges Badr, Grenoble Graduate School of Business, Grenoble, France, nabil.badr@alumni.grenoble-em.com

ABSTRACT

In any city – smart or not – emergency response is a critical service. In smart cities, the use of technology to manage access to and dispatching of emergency vehicles is particularly important. However, when a system must manage processes spanning multiple computers, clock drift becomes a prominent issue. We show the impact that clock drift can have in a prototype emergency management dispatch system along with a case-study illustrating design techniques that can remedy this issue.

Keywords: Emergency management dispatch; time synchronization; clock drift.

*Corresponding author

INTRODUCTION

Currently, 54% of the world's population lives in cities. According to the UN's World Urbanization Prospects report, 66% of the world's population is expected to reside in urban areas, representing an additional 2.5 Billion residents of cities, by 2050 (United Nations, 2014). With numbers as staggering as these the need for smart cities that exploit IT services to connect communities with the goal of improving life is all too apparent (Washburn, 2010). A chief tenet of smart city design is the connection of citizens to city services and infrastructure. One such service is emergency response. With the growth of urban populations, the growth of emergency response must also keep pace.

Emergency medical services are about saving people's lives and reducing the rate of mortality and morbidity from accidents in civilian settings. The onus remains on Emergency Management Systems (EMS) to provide effective urgent medical care. An EMS is expected to observe each of the six "points" of "the star of life" (Zatz *et al.*, 1992). This requires capturing data from (1) the point of detection of an incident, (2) the report of the incident to professional dispatch; (3) the response of the first responders; (4) the on scene care; (5) both the transportation and care in transit; and finally (6) the transfer to definitive care, such as a hospital, clinic or other facility. If any one of the time stamps associated with the star of life is made incorrectly, the EMS loses visibility into the performance of this critical smart city service.

This paper highlights lessons learned from a case study within a densely populated urban environment. In this setting we identify the impact of time synchronization issues in the Emergency Management Dispatch (EMD) system design on the ability to accurately record time related data. Through this case study we show how specific changes to an EMD system correlate to improvements in capturing time related data.

BACKGROUND

Although there are a number of design principles guiding EMD system design, few highlight basic principles of software engineering and user interface design methodologies (Miller, 2017; Salasznyk *et al.*, 2006). In fact, several researchers argue that no universal guidelines on EMD System design exist as emergency management systems depend on the local situation (Chumer *et al.*, 2004). Nevertheless, a well-designed EMD system must coexist with smart city requirements and be capable of providing efficient and effective communication and coordination of emergency response actions (Alazawi *et al.*, 2014; Chen *et al.*, 2007).

In their work, Chumer *et al.* (2004), propose a framework for system design and development and present eight general design principles for emergency management systems that are essential for an effective EMD system design. However, taking into consideration the EMS stakeholders and the context of their operations, the dynamic development of such systems must be coupled with methodological approaches (Chen *et al.*, 2008; Van de Walle & Turoff, 2008; Voss & Wagner, 2010). These researchers specify a set of phases to be included in the design of any emergency management system. In these phases, that are much like the arms in the "star of life", the response phase is a critical phase where work may be delayed because of limited or contradictory information. Thus, the authors call for integrating information from different sources, using several types of media. Bringing together multiple sources heightens the need to accurately record time. This need is echoed by Salasznyk and Lee (2006), who were concerned with the time needed for each of the phases of the response trajectory as this affects the response time overall. Admittedly, there is controversy regarding the use of response time as a metric, nevertheless a recent study by White *et al.* (2011) indicates that reducing the response time by one minute can increase the number of lives saved by six percent.

As response time is ultimately specified as the difference between two time stamps, this calculation will only be as accurate as the time stamps themselves. Even when initially set accurately, real clocks will differ after some amount of time due to clock drift or skew, caused by clocks counting time at slightly different rates. Thus, clocks in a distributed system must synchronize to the same "global" time.

In general, there are two popular methods for physical clock synchronization (Barroso *et al.*, 2011). The first is the internal clock synchronization, where all computer clocks in a system are synchronized with one another away from the global reference clock time. The second method is the external clock synchronization, where all computer clocks in a system are synchronized with an external reference clock, like the Coordinated Universal Time (UTC) signal.

Furthermore, as physical time is not perfectly synchronized, the Lamport logical time clocks (Lamport, L. (1978) is another mechanism for synchronization based on the causal order of events occurring at different machines ("happened-before relation"). Two events occurring at the same machine happen in the order in which they are observed by the machine. A Lamport logical clock is a monotonically increasing software counter which is not referenced to any physical clock. Each process keeps its own logical clock which it uses to apply so-called Lamport timestamps to events. Finally, the oldest technique for time synchronization in a network is the Network Time Protocol (NTP) which has been under continual development and updating since 1979 (Mills et al., 2010). The NTP is suitable for externally synchronizing machines to UTC via the internet. It works based on a hierarchical, peer-to-peer structure of computers and servers organized into strata with the top level strata containing a set of high-precision reference clocks. Further, it enables clients to resynchronize sufficiently frequently to offset typical hardware drift rates and provides a reliable service tolerating lengthy losses of connectivity.

To ensure the integrity of time stamps within EMD systems, synchronizing all computers to the same clocks using the same synchronization protocols is a must. The remainder of this paper uses a prototype EMD system to understand the extent of time synchronization related data corruption in response time data under two different time synchronization protocols.

CASE STUDY SYSTEM

The ambulance response service at the heart of this case study was conceived to provide insight on vital system information including emergency response times by time of day, region, and type of emergency. Relying mostly on paper based record keeping, the EMD management team was not equipped to rapidly analyze the data for timely decisions. In 2011, the system was overhauled to include a centralized dispatch center functionality with an information system designed to provide data entry for call taking protocols, mission management and patient care records (Badr, 2016). One of the objectives of this solution was to build a source of reliable data for performance reporting and decision making for operations improvement.

As implemented, the component of mission management comprises a dashboard for tracking mission progress. The dashboard represents data input by the dispatcher showing mission progress timing milestones that are loosely mapped to the stages of the blue star of life. Those milestones are: record creation, scheduling of ambulance services, notification to the ambulance station, departure of the ambulance, arrival at the scene, departure from the scene, arrival at the hospital or care facility, patient hand over, and ambulance available again.

The basic architecture of the system consists of a web-based interface that emphasizes principles of access security, resiliency and ease of use expected to support input, reporting and user interface in both English and the native language of the system locale. The call center infrastructure consists of 12-14 workstations taking calls and logging them into a system.

The workstations for data entry at the call center are physically in a different location than the corresponding application and database servers. The workstations and servers are separated by a set of firewalls and connected via a Virtual Private Network (VPN). These two end points may incur delay in synchronizing their time. To achieve time synchronization over the network, our case used the NTP. As a time source, the Global Positioning System (GPS) is used for central clock synchronization. Although GPS time signals are accurate, clock skew still introduces time stamp issues into database records. The more computers involved in a system, the more likely that skew will be reflected in the data.

When first established, the EMD system followed the process of time stamping and synchronization as depicted in Table 1. The table specifies the component, the time reference to which that component is synched, and a time stamp ID associated with the component. As shown in Table 1, at regular intervals, the application and database servers set their clock to the reference time tS1 of the NTP Time Server. When a user/agent logs into a workstation session, the time stamp is reset to the Application Server time, thus at the workstation t = tS3. When opening a new call on the system, a key database record is created with a time stamp of tR1 which is equal to the database clock time at that instance, tS2. In this case, if the server clock is skewed when the record is created, but gets reset subsequently, then a timing error could be introduced. As the call progresses, all statuses and subsequent records are stamped with the workstation time tRn as it is the workstation that initiates the data entry request, tRn = t*.

In this original implementation of the case study system, tRn could drift based on the internal workstation clock. If the workstation was not cycled at every shift, to synchronize its internal clock with the system, the absolute difference between tS3 and t could increase inducing errors in time measurements. For example, within 5 minutes a clock could drift 280 milliseconds. If the clock synchronization occurred on a 24 hour basis, then a total of 80 seconds of drift may occur between the system and workstation clock. These timestamp errors manifest themselves in "events happening in the past" (i.e. tRn < tR1 = tS2). Furthermore, due to the number of computers used in creating any one record the drift was not confined to that of one computer. As such, measuring and correcting drift from log files is an untenable task.

Table 1: Time Stamp and Synchronization Process Prior to Correction in November 2016.

Component	**Time Synch**	**Time Source**	**Time Stamp**
NTP Service (S1)	Polling interval = 8 sec	GPS	tS1
Domain Controller (DC)	Time synch based on VMs (Hyper-V)	NTP	tDC
Database Server (S2)	Time synch based on VMs (Hyper-V)	NTP	tS2
Application Server (S3)	Time synch based on VMs (Hyper-V)	NTP	tS3
Workstation Session	Time synch with Application Server Performed at user login	S3	t set to tDC
Key Record Time Stamp in DB	Time stamp applied by Database Server when the user creates the record (i.e. When call is received by dispatcher)	S2 (Unreliable time sync)	tR1 = tS2
Remaining Records Time Stamp	No Time Synch; Workstation current time (internal clock) is used tRn = t* based on WS current time	Internal WS time	tRn = t* based on WS current time

Recognizing the possibility for significant drift across the clocks, the system was modified in November of 2016. Specifically, the NTP Service (S1) polling interval was shortened to four seconds and synchronized with GPS. As for the Domain Controllers (DC), the polling interval was also shortened to four seconds, and the time synchronization based on the virtualization platform (Hyper-V) was disabled. All domain controllers were reconfigured to synchronize with NTP. Further, the Domain Controller became the source for time synchronization on all hyper servers (Database server-S2 and Application server -S3) with polling intervals set to four seconds.

Additionally, the system was set such that all workstations synchronize with the Domain Controller at user login. The Key Record Time Stamp workstation synchronizes with Database Server (S2) when a call is received by the dispatcher, and all other PCs in the system synchronize with the domain controller with a polling interval set to 20 seconds. Table 2 serves to summarize these system changes.Finally, on May 31, 2017, the Domain Controller was restarted forcing a resynchronization across the system.

Table 2: Time Stamp and Synchronization Process after Correction in November 2016.

Component	**Time Synch**	**Time Source**	**Time Stamp**
NTP Service (S1)	Polling interval = 4 sec	GPS	tS1
Domain Controller (DC)	Polling interval = 4 sec	NTP	tDC = tS1
Database Server (S2)	Polling interval = 4 sec	DC	tS2 = tDC
Application Server (S3)	Polling interval = 4 sec	DC	tS3 = tDC
Workstation Session	Performed at user login	DC	t set to tDC
Key Record Time Stamp in DB	Time stamp applied by Database Server when the user creates the record (i.e. When call is received by dispatcher)	S2	tR1 = tS2 =tDC (time synch with DC time polling every 4 sec)
Remaining Records Time Stamp	Synch with workstation time; Polling interval set to 20 sec to synch with DC time (central time keeper)	DC	tRn = tDC every 20 seconds

RESULTS

While the changes made to the time synchronization protocol for the EMD system under study may seem trivial, the impact on the data was tremendous. This impact is notable for two reasons (1) this dispatch center receives a high level of traffic and (2) the field of ambulance response revolves around short response times – thus, an error of only 30 seconds represents a 6% deviation relative to a hypothetical eight-minute response time. In order to appreciate these points of impact, the data reported here spans from 1 January 2016 to 30 June 2017 and includes 130,867 records. For each call a time stamp in yyyy-mmddThh: mm:ss+hh:mm format was recorded for the time the record was created (eg. call received), the time the mission was scheduled (e.g. assigned to an ambulance), the time the ambulance's station was notified (this stamp interval was added on 9 November 2017), the time the ambulance departed to the scene, the time the ambulance arrived to the case, the time the ambulance departed the case, the time the ambulance arrived to the transport destination, the time the mission was deemed complete, and the time the ambulance was available for service again. For the purpose of this study, we are interested only in the first four time stamps as these are the time stamps required to calculate ambulance response time following the commonly used definitions of response time – divided into the three intervals: record created to scheduled; call scheduled to ambulance departure; and time from departure to arrival at the case; subsequent to 9 November 2017 we also include the intervals from scheduled to site notified and from site notified to departed.

Within the fifteen month study period, time synchronization errors were identified on the basis of negative time progressions when moving logically through the data. For example, if the time stamp associated with the mission being scheduled was before the time stamp associated with the creation of the record, then the difference between these two time stamps would yield negative time. Negative time intervals were considered to be a function of time synchronization issues. Figure 2 illustrates the number of records having negative time in the interval between the creation and scheduling of the record across the full time period of this study. Of note in Figure 2 are the two major drops in number of erroneous records – the first coinciding with the system change in November, 2016 and the second with the DC restart on May 31, 2017.

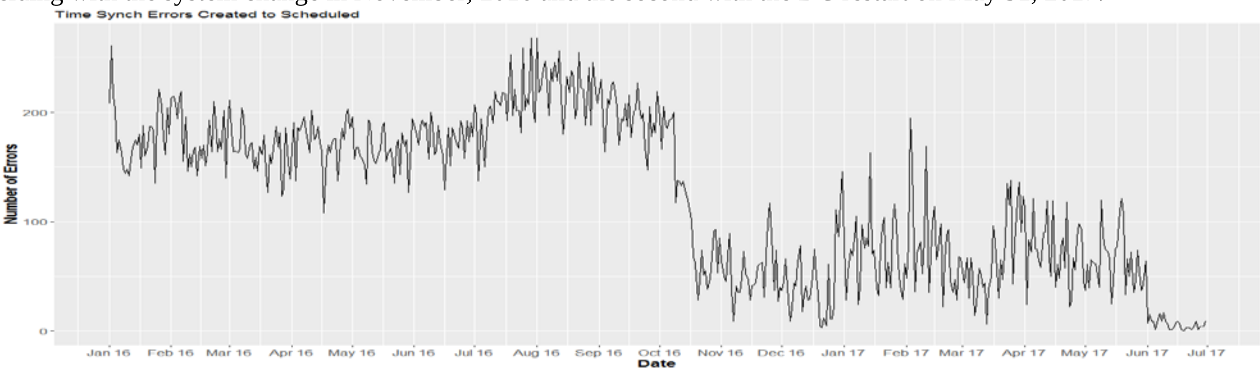


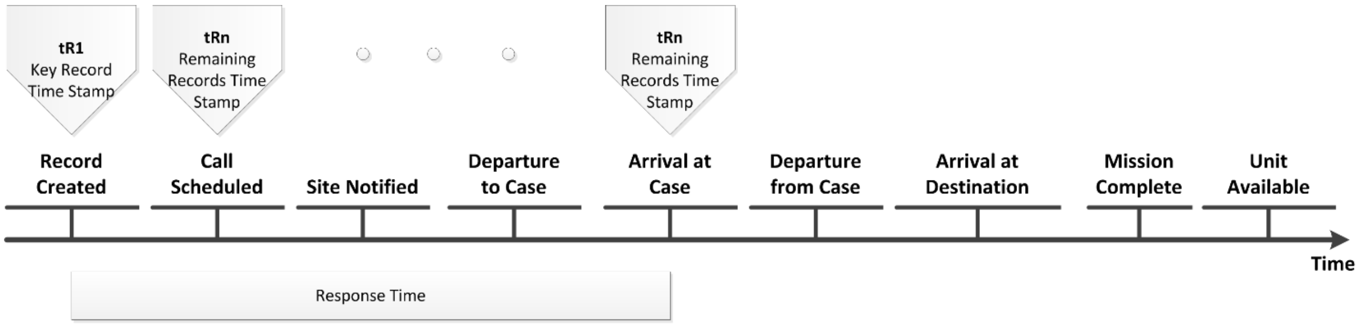
Figure 2: Number of synchronization errors for the interval between created and scheduled.

Of the 130,867 records, the vast majority of negative time intervals occurred relative to the mission scheduled and created on records. The percent of negative time interval records apportioned by response time interval type and divided across the three periods of pre-November changes, post-November, but pre-May and after May can be seen in Table 3.

Table 3: Percent of records with time sync errors in the three intervals prior to November 2016, between November 2016 and May 2017, and post-May 2017.

Time Interval	**% Time Sync Errors Through 8 Nov 2016**	**% Time Sync Errors 9 Nov 2016 Through May 2017**	**% Time Sync Errors June 2017**
Created To Scheduled	75.24	26.74	2.40
Scheduled to Site Notified	0.00	0.63	0.20
Site Notified to Departure	0.00	0.68	0.25
Scheduled to Departure	1.71	5.11	0.28
Departure to Arrival at Site	0.59	0.22	0.22

While a time synchronization error rate of 75% for the first interval of the response time period may seem shocking, it is logical as the created on time stamp was generated by the call center server while the remaining time stamps are generated on the workstation terminals. Prior to the change in November, it was these ICT component links that were most prone to time synchronization issues. As a further explanation for the drop in time synchronization issues, Figure 3 serves to highlight the response trajectory relative to the change made in November.



	Before Parameter Change	After Parameter Change
Key Record Time Stamp in DB	Time stamp applied by Database Server when the user creates the record (i.e. When call is received) **tR1 = tS2 (Ts2 with unreliable time sync)**	Time stamp applied by Database Server when the user creates the record (i.e. When call is received by dispatcher) **tR1 = tS2 = tDC every 4 sec**
Remaining Records Time Stamp	No Time Synch; Workstation current time (internal clock) is used **tRn = t* based on WS current time**	Synch with workstation time; Polling interval set to 20 sec to synch with DC time (central time keeper) **tRn = tDC every 20 sec**

Figure 3: Time trajectory for ambulance response mapped against changes in time synching protocol made in November 2016.

DISCUSSION AND CONCLUSION

A dependable time keeper is a key component of any system that relies on data for decision making. In our case the time keeper is the NTP server that keeps time with a GPS source. The system time synchronization agent has been designated as the “Domain Controller” that makes sure it is closely tracking to the NTP. The downstream servers participating in the domain are then updated through a polling mechanism initiated by each of the components.

Before the changes were made, the polling mechanism was not deterministic and was left up to each server to update their time based on how busy they were in processing other computing requests. Additionally, the virtual machines within the EMD system were synchronizing via software synchronization (Hyper-V) which is based on logical clock synchronization. As a result, we often saw ambiguous event ordering where time values assigned to events were not necessarily the same as the actual times at which they occurred. Since the actual time is important in EMD systems, the system components must synchronize with a physical clock. Accordingly, the system synchronization was based on the GPS timing through the NTP Service. Once the changes were performed and the server configuration changed to poll at a specific interval of 4 seconds, the errors reduced by 70%. Hence, a robust time keeping protocol is one that is definitive and based on an architecture that dependably polls the time source for time keeping that is as accurate as possible.In our case, once the time keeping was adjusted, the system did not converge to a low error state until the end of May when the Domain controller (system time synchronization agent) reset the system back to the same starting point.

The sharp decrease (from 75% to 2%) in erroneous records is critical when considering the impact of reporting data from an EMD system. There are two outcomes that such inaccurate data might yield: 1) an analyst reporting response time may choose to exclude all records that exhibit a non-logical negative time interval or 2) the analyst may unwittingly include all records with a negative time interval. If the first outcome is realized then the response times reported will be consistently biased and may skew the decision making of city managers. If the second outcome is realized then the reported average response times will be artificially low leading ambulance service providers to ignore potentially important response time improvements that may be necessary. As our case illustrates, the polling intervals for synchronization have a vital impact on keeping the correct clock time and reducing drift. We also show that distributed subsystems are more vulnerable to time keeping errors. The persistence of some errors in the system despite the changes made to the time synchronization protocols serves to highlight the scope of possible errors within a highly networked smart city system. Smart city ecosystems clearly require a robust architecture with a complex time keeping protocol to maintain synchronization between systems and subsystems to minimize the effect of clock drift. Hence, on a macro level, this paper supports an essential role for central time keeping in smart cities especially where systems and subsystems are to be integrated, exchanging vital, time sensitive data.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the Lebanese Red Cross for their interpretation of and comments on the findings in this paper

REFERENCES

[1] Alazawi, Z., Alani, O., Abdljabar, M. B., Altowaijri, S., & Mehmood, R. (2014). A smart disaster management system for future cities. In *Proceedings of the 2014 ACM international workshop on Wireless and Mobile Technologies for Smart Cities* (pp. 1-10). ACM.

[2] Badr, N. G. (2016). Could ICT be harnessed for prehospital emergency medical services? In *Proceedings of the International Joint Conference on Biomedical Engineering Systems and Technologies* (pp. 269-276). SCITEPRESS-Science and Technology Publications, Lda.

[3] Barroso, L.A., Clidaras, J., Hölzle, U, (2013). The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis Lectures on Computer Architecture*, *8*(3), 1-154.

[4] Chen, R., Sharman, R., Rao, H. & Upadhyaya, S. (2007). Design principles for critical incident response systems. *Information Systems and E-Business Management*, *5*(3), 201-227.

[5] Chen, R., Sharman, R., Rao, H. & Upadhyaya, S. (2008). Coordination in emergency response management. *Communications of the ACM, 51*(5), 66–73.

[6] Turoff, M., Chumer, M., van de Walle, B., & Yao, X. (2004). The design of a dynamic emergency response management information system (DERMIS). *Journal of Information Technology Theory and Applications*, *5*(4), 1.-35

[7] Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, *21*(7), 558–565. doi:10.1145/359545.359563.

[8] Miller, A. N., Sellnow, T., Neuberger, L., Todd, A., Freihaut, R., Noyes, J., & Gamhewage, G. (2017). A systematic review of literature on effectiveness of training in emergency risk communication. *Journal of Health Communication*, *22*(7), 612-629.

[9] Mills, D., Martin, J., Burbank, J., & Kasch, W. (2010). Network time protocol version 4: Protocol and algorithms specification (No. RFC 5905). DOI 10.17487/RFC5905, June 2010. Retrieved from https://www.rfc-editor.org/info/rfc5905 (October 8, 2017).

[10] Salasznyk, P. P. & Lee, E. E. (2006). A systems view of data integration for emergency response. *International Journal of Emergency Management*, *3*(4), 313-331.

[11] United Nations (2015). *World urbanization prospects: The 2014 revision*. Department of Economic and Social Affairs, Population Division (ST/ESA/SER.A/366). Retrieved from https://esa.un.org/unpd/wup/publications/files/wup2014-report.pdf (October 8, 2017).

[12] Van de Walle, B. & Turoff, M. (2008). Decision support for emergency situations. *Information Systems and E- Business Management*, *6*, 295-316.

[13] Voss, M., & Wagner, K. (2010). Learning from (small) disasters. *Natural Hazards*, *55*(3), 657-669.

[14] Washburn, D., Sindhu, U., Balaouras, S., Dines, R. A., Hayes, N. M., & Nelson, L. E. (2010). *Helping CIOs Understand “Smart City” Initiatives: Defining the Smart City, Its Drivers, and the Role of the CIO*. Cambridge, MA: Forrester Research. Inc.

[15] White, J., Thompson, C., Turner, H., Dougherty, B. & Schmidt, D.C. (2011). WreckWatch: Automatic Traffic accident detection and notification with smartphones. *Mobile Networks and Applications*, *16*, 285-303.

[16] Zatz, A. (1992). Star of life. *Rescue-EMS Magazine*, July/August. Retrieved from www.angelfire.com/co/fantasyfigures/staroflife.html (October 8, 2017).