Natarajan Meghanathan
David C. Wyld (Eds)

Computer Science & Information Technology

9th International Conference on Networks & Communications
(NeCoM - 2017)
September 30~October 01, 2017, Dubai, UAE

AIRCC Publishing Corporation

Volume Editors

Natarajan Meghanathan,
Jackson State University, USA
E-mail: nmeghanathan@jsums.edu

David C. Wyld,
Southeastern Louisiana University, USA
E-mail: David.Wyld@selu.edu

ISSN: 2231 - 5403
ISBN: 978-1-921987-72-4
DOI : 10.5121/csit.2017.71201 - 10.5121/csit.2017.71207

Typesetting: Camera-ready by author, data conversion by NnN Net Solutions Private Ltd., Chennai, India

Preface

The 9th International Conference on Networks & Communications (NeCoM - 2017) was held in Dubai, UAE, during September 30~October 01, 2017. The 6th Conference on Software Engineering and Applications (SEAS - 2017), The 6th Conference on Control, Modelling, Computing and Applications (CMCA - 2017) and The 3rd International Conference on Computer Science, Information Technology (CSITEC - 2017) was collocated with The 9th International Conference on Networks & Communications (NeCoM - 2017). The conferences attracted many local and international delegates, presenting a balanced mixture of intellect from the East and from the West.

The goal of this conference series is to bring together researchers and practitioners from academia and industry to focus on understanding computer science and information technology and to establish new collaborations in these areas. Authors are invited to contribute to the conference by submitting articles that illustrate research results, projects, survey work and industrial experiences describing significant advances in all areas of computer science and information technology.

The NeCoM-2017, SEAS-2017, CMCA-2017, CSITEC-2017 Committees rigorously invited submissions for many months from researchers, scientists, engineers, students and practitioners related to the relevant themes and tracks of the workshop. This effort guaranteed submissions from an unparalleled number of internationally recognized top-level researchers. All the submissions underwent a strenuous peer review process which comprised expert reviewers. These reviewers were selected from a talented pool of Technical Committee members and external reviewers on the basis of their expertise. The papers were then reviewed based on their contributions, technical content, originality and clarity. The entire process, which includes the submission, review and acceptance processes, was done electronically. All these efforts undertaken by the Organizing and Technical Committees led to an exciting, rich and a high quality technical conference program, which featured high-impact presentations for all attendees to enjoy, appreciate and expand their expertise in the latest developments in computer network and communications research.

In closing, NeCoM-2017, SEAS-2017, CMCA-2017, CSITEC-2017 brought together researchers, scientists, engineers, students and practitioners to exchange and share their experiences, new ideas and research results in all aspects of the main workshop themes and tracks, and to discuss the practical challenges encountered and the solutions adopted. The book is organized as a collection of papers from the NeCoM-2017, SEAS-2017, CMCA-2017, CSITEC-2017.

We would like to thank the General and Program Chairs, organization staff, the members of the Technical Program Committees and external reviewers for their excellent and tireless work. We sincerely wish that all attendees benefited scientifically from the conference and wish them every success in their research. It is the humble wish of the conference organizers that the professional dialogue among the researchers, scientists, engineers, students and educators continues beyond the event and that the friendships and collaborations forged will linger and prosper for many years to come.

Natarajan Meghanathan
David C. Wyld

Organization

General Chair

David C. Wyld	Southeastern Louisisna University, USA
Jan Zizka	Mendel University in Brno, Czech Republic

Program Committee Members

Aalya Alajaji	Prince Sultan University, Saudi Arabia
Adnan A. Rawashdeh	Yarmouk University, Jordan.
Amizah Malip	University of Malaya, Malaysia
Anas Shatnawi	University of Quebec at Montreal, Canada
Artyom Grigoryan	UTSA,United States
Ashwath Rao B	Manipal Institute of Technology, India
Azam Khalili	University of Malayer, Iran
Belal M. Abuata	Yarmouk University, Jordan
Belkacem bekhiti	Boumerdes University, Algeria.
Bing Li	Xi'An Technological University, China
Bingwen Feng	Jinan University, China
Busyairah Syd Ali	University of Malaya, Malaysia
Christophe Claramunt	Naval Academy Research Institute, France
Cristina-Loredana Duta	University Politehnica of Bucharest, Romania
Dhguo	Xiamen University, China
Gurkana	Yildiz Technical University, Turkey
Haipeng Cai	Washington State University, USA
Hamid Ali Abed AL-Asadi	Basra University, Basra, Iraq
Hamid Mcheick	Université du Québec à Chicoutimi, Canada
Hani Bani-Salameh	The Hashemite University, Jordan
Héctor Migallón	Miguel Hernández University, Spain
Hu, Yu-Chen	Providence University,Taiwan
Isa Maleki	Islamic Azad University, Iran
Issac Niwas Swamidoss	Nanyang Technological University, Singapore
Janes Andrea	Free University of Bozen-Bolzano, Italy
Jichiang Tsai	National Chung Hsing University, Taiwan
Jose Gonzalez	University of A Coruña, Spain
Jun Liu	University of Michigan Dearborn, USA
Kevin Gary	Arizona State University, USA
Khaled Almakadmeh	Hashemite University, Jordan
Kheireddine abainia	USTHB university, Algeria
Koh You Beng	University of Malaya, Malaysia
Lark Kwon Choi	The University of Texas at Austin, USA
Lee Beng Yong	Universiti Teknologi MARA, Malaysia
Limiao Deng	China University of Petroleum, China
Liyakathunisa Syed	Prince Sultan University, Saudi Arabia
Mohammad Zarour	Prince Sultan University, Kingdom of Saudi Arabia

Mohammed Nabil El Korso	Paris Nanterre University, France
Mohanned Akour	Yarmouk University, Jordan
Nadjia Benblidia	Saad Dahlab University -Blida1-, Algeria
Nahlah M. Ameen Shatnawi	Yarmouk University, Jordan
Pelin Angin	Purdue University, USA
Pengfei Wu	Sichuan University, China
Prasan Kumar Sahoo	Chang Gung University, Taiwan
Ramayah A/L Thurasamy	Universiti Sains Malaysia, Malaysia
Razieh Saremi	Stevens Institute of Technology, USA
Res. See.GÜRKAN ALPASLAN	Yildiz Technical University, Turkey
Rocio Maciel Arellano	University of Guadalajara, México
Saber Elsayed	University of New South Wales, Australia
Samer Zain	Birzeit University, United Kingdom
Samia Nefti-Meziani	University of Salford, UK.
Sandra D'Souza	Manipal University, India
Selahattin KOSUNALP	University of Bayburt, Turkey
Shah J Miah	Victoria University, Australia
Son Nguyen Thai	Tra Vinh University, Vietnam
Sos Agaian	The University of Texas at San Antonio, USA
Tanmoy Sarkar	Microsoft Corporation, USA
Tanzila Saba	Prince Sultan University, Kingdom of Saudi Arabia
Tonghan Wang	East China University of Technology, China
Tzung-Pei Hong	National University of Kaohsiung, Taiwan
Wan Shuai	Northwestern Polytechnical University, China
Xianglei Xing	Harbin Engineering University, China.

Technically Sponsored by

Computer Science & Information Technology Community (CSITC)

Networks & Communications Community (NCC)

Organized By

Academy & Industry Research Collaboration Center (AIRCC)

TABLE OF CONTENTS

9th International Conference on Networks & Communications (NeCoM - 2017)

6th International Conference on Software Engineering and Applications (SEAS - 2017)

6th International Conference on Control, Modelling, Computing and Applications (CMCA - 2017)

3rd International Conference on Computer Science, Information Technology (CSITEC - 2017)

FAULT DIAGNOSIS IN BRAKING SYSTEM OF MINE HOIST BASED ON THE MOMENT CHARACTERISTICS

Li Juanjuan[1], Wang Aiming[1], Meng Guoying[1], Xie Guangming[2], Wang Shuai[1], Hu Liang[1], Jia Yifan[1]

[1]School of Mechanical Electronic &Information Engineering, China University of Mining and Technology (Beijing)
[2]College of Engineering, Peking University

ABSTRACT

Fault diagnosis method based on the moment characteristic of system pressure data is presented in this paper. Using AMESim simulation technology, three typical faults of the brake system are studied. After the moment and the percentile characteristics of the pressure curve of the hydraulic system are extracted and used as characteristic parameter, fault information is diagnosed effectively using the BP neural network. The problem that the signal is not synchronized and the characteristic parameters can not be obtained accurately are overcome. It provides some theoretical basis for the intelligent diagnosis and predictive maintenance of the high speed deep well hoist.

KEYWORDS

Moment characteristics; Fault diagnosis; Braking system; Performance degradation; Deep mine hoist

1. INTRODUCTION

Ultra deep mine hoist has large inertia with complex operation conditions and a complex system of electromechanical coupling. Safe and stable operation is related to staff's life and property safety [1-3]. Brake is the last safeguard for safe operation of mine hoisting. It has very important significance to research the fault and performance and to provide the basis theory for intelligent diagnosis and performance evaluation of the braking system.

Nowadays there are a lot of research focusing on disc brake fault diagnosis. literature [4-9] mainly detected the oil pressure on the hydraulic station, opening brake pressure, closing brake pressure，residual pressure, brake clearance, friction coefficient, brake disc runout, and etc. parameters to make fault diagnosis, the literature [10,11] measured the spring pressure, the spring pressure variation, hydraulic pressure, hydraulic pressure variation and brake state detecting switch to make fault diagnosis, and the literature [12,13] detected brake clearance, decomposition and reconstruction of wavelet energy entropy as feature parameters, combined with neural

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 01– 11, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71201

network to make fault diagnosis. All of the above methods can get effective fault diagnosis, but there are still some problems: 1) characteristic parameters are not accurate, 2) characteristic parameters can not represent the overall characteristics of the braking system, and 3) other issues. For example, because of the long-term load, the linearity of the signal and the zero drift problems are difficult to solve, regardless of choosing strain gauge or ballast sensors to measure of braking positive pressure; there is no feasible scheme for on-line monitoring of the friction coefficient [14], and the detection of brake clearance is influenced by brake shoe's wear and other factors. To overcome the shortcomings above, this paper presents a new fault diagnosis method. The method extracts moment and percentile characteristics from the brake pressure-time curve as characteristic parameters, then makes fault diagnosis combined with BP neural network.

2. BRAKING SYSTEM PERFORMANCE AND CALCULATION OF MINIMUM PRESSURE VALUE OF BRAKING SYSTEM FOR SAFETY BRAKING

2.1 Brake System Performance Requirements

Coal Mine Safety Regulations and the standard of coal mine mechanical and Electrical Equipment stipulate the following requirements to improve the working performance and working state of the hoist [15].

(1) Idle time of disc brake should not exceed 0.3 s.

(2) The gap between brake shoe and brake disc should not be greater than 2mm.

(3) Braking momentum requirement, i.e. the braking momentum generated by brakes during the work and safety brake should be at least three times the maximum static load momentum.

(4) When safety brake in vertical well, the safety braking deceleration must be less than or equal to 5m/s2 during vessel's ascending with heavy-load, and must be greater than or equal to 1.5 m/s2 during vessel's descending with heavy-load.

2.2 Calculating the Minimum Pressure Value of Braking System when Safety Braking

Coal Mine Safety Regulations stipulates that brake torque generated by hoist during the work and safety brake shall not be less than 3 times the maximum static load torque, i.e.

$$M_Z \geq 3M_j \qquad (1)$$

Where M_Z is brake torque, M_j is static load torque.

When a heavy lifting hoist is safely braking, calculated brake pressure value is 4.36MPa [16]. After the performance of brake parts is decreased, in order to meet the requirement that the braking torque is not less than 3 times the maximum static load torque, the minimum pressure value of the braking system is calculated as follows:

According to the principle of Darren Bell, when a heavy lifting hoist is safely braking, the torque balance equation to the hoist drum is [17,18]

$$M_Z + M_J = M_d \tag{2}$$

Where M_J is static and resistance torque, Nm. M_d hoist inertia torque, Nm.

.Where:

$$M_Z = 2(K \cdot x - P_Z \cdot A) \cdot n \cdot \mu \cdot R_Z \tag{3}$$

$$M_J = k \cdot M_j = k \cdot m \cdot g \cdot R_J \tag{4}$$

$$M_d = \sum m \cdot a_3 \cdot R_J \tag{5}$$

Where K is spring stiffness, N/mm. x spring preloading length, mm. $P_Z^{'}$ brake system pressure during safety braking, MPa. μ brake shoe friction coefficient. R_Z brake radius, m. k mine resistance coefficient. If the hoisting container is skip, k=1.15. If the hoisting container is cage, k=1.20. m load mass, kg. g gravitational acceleration, m/s^2. R_J drum radius, m. $\sum m$ mass of hoist equivalent, kg. a_3 lifting load deceleration, m/s^2.

Take k=1.15, combining Eqs. (1),(4) yields

$$M_J = \frac{1.15}{3} M_Z \tag{6}$$

Substituting Eqs. (3),(5)and (6) into Eq. (2) gets

$$P_Z^{'} = \frac{K \cdot x - \dfrac{3\sum m \cdot a_3 \cdot R_J}{8.3n \cdot \mu \cdot R_Z}}{A} \tag{7}$$

Substituting the value of parameters in Tab. 1 into Eq. (7) gets

$$P_Z^{'} = \frac{10328 \times 8.38 - \dfrac{3}{8.3} \times \dfrac{238860 \times 3.5 \times 2.25}{16 \times 0.3 \times 2.45}}{84.2 \times 10^{-4}} = 3.41 MPa$$

When the hoist is safety braking, the braking system pressure value is between 4.36MPa and 3.41MPa, which can meet braking momentum requirement.

3. SIMULATION AND ANALYSIS TO THE MAIN COMPONENTS PERFORMANCE DECLINE

In the process of operation, because the brake spring, brake cylinder, piston, seals and other important components work long-term under the condition of high load, the brake performance will gradually degenerate until failure occurs. The following carry out performance simulation of three main components, i.e. the brake spring stiffness reduction, brake shoe friction coefficient decline and cylinder leakage. The simulation only takes into account the heavy lifting condition. And the simulation time is set to a fixed value, that is to set the brake system 1-2s for accumulator charging, 2-5s for the hoist operation, 5-8s for constant deceleration braking, 8-10s system pressure relief. Sampling frequency is set to 100HZ. In order to reduce the length of the article, building simulation platform, reliability verification and the typical fault simulation methods are detailed in the literature [16], which is no longer mentioned here.

3.1 Friction Coefficient Decreased

The normal friction coefficient is 0.3. The simulation results with different friction coefficients are shown in figure 1, 2 and 3:

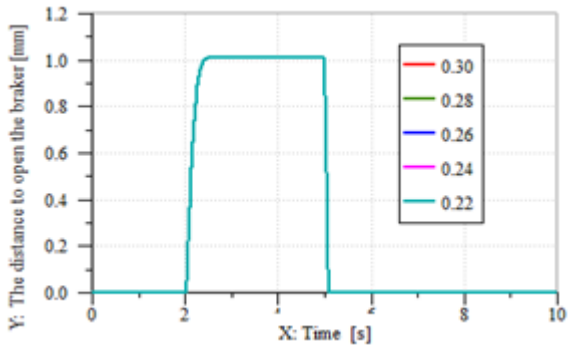


Figure 1 The brake clearance with different friction coefficients

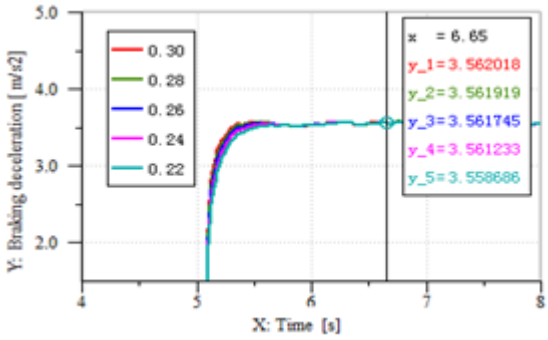


Figure2 Braking deceleration with different friction coefficients

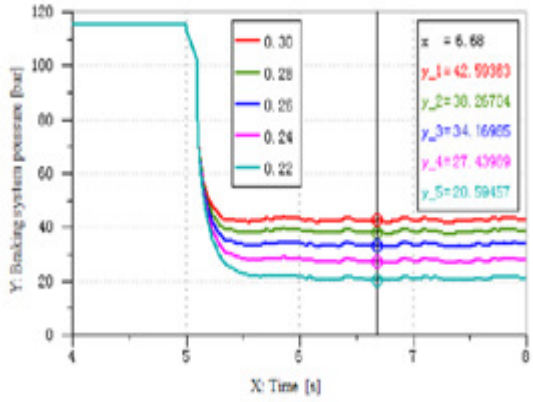


Figure 3 Braking system pressure with different friction coefficients

From the simulation diagram one can obtain that the brake clearance and the idle time of the brake do not change when the friction coefficient is reduced. This meets the requirements that the brake clearance is not more than 2mm and the idle time is not more than 0.3s. Braking deceleration meets the requirements of system deceleration. When the friction coefficient is greater than 0.26, the minimum pressure value of the braking system can meet the requirement of 3 times the maximum static load torque. From the analysis above we can conclude that: When the friction coefficient of braking system is between 0.26~0.3, brake performance degraded, but still meets the Coal Mine Safety Regulations requirements. When the friction coefficient is less than 0.26, the pressure of the braking system is reduced below 3.41MPa, which does not meet the Braking momentum requirements. The failure of brake system friction coefficient declined occurs.

3.2 Spring Stiffness Reduction

The simulation results in different spring stiffness are shown in figure 4, 5 and 6.

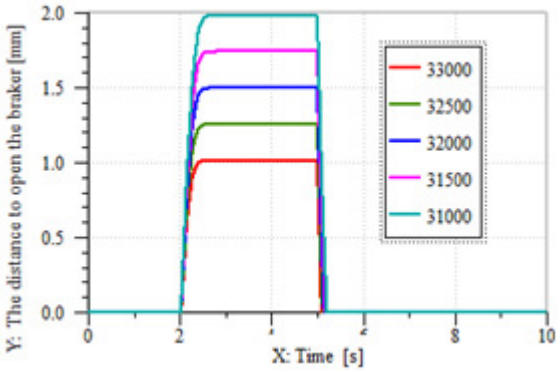


Figure 4 The brake clearance in different spring stiffness

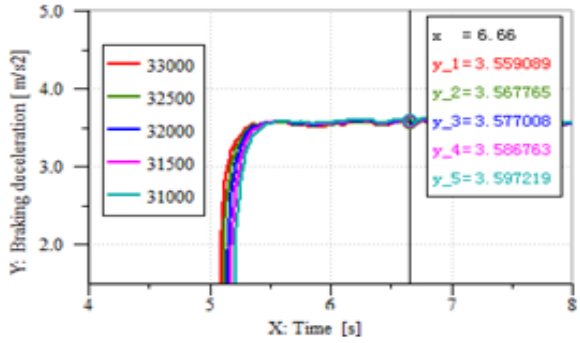


Figure 5 Braking deceleration in different spring stiffness

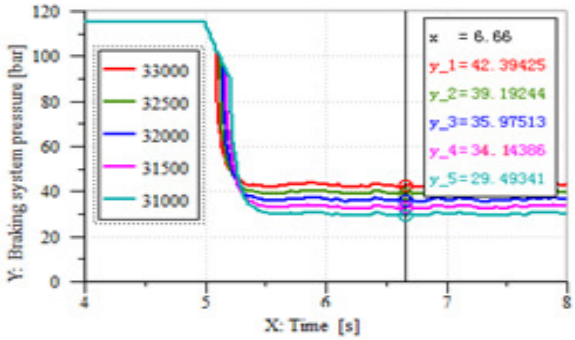


Figure 6 Braking system pressure in different spring stiffness

We can see from the simulation diagram. When the spring stiffness is reduced, the braking deceleration meets the requirements of system deceleration. The brake clearance and the idle time of the brake are gradually increased. When the spring stiffness is reduced to 31000, brake system meets the requirements that the idle time is not greater than 0.3s and the brake clearance is not greater than 2mm. When the spring stiffness is 31500, the pressure of the braking system is reduced to just meet the requirement of braking torque is not less than 3 times the maximum static load torque. The results show that when the spring stiffness is less than 31500, the failure of the spring stiffness reduced occurs.

3.3 Cylinder internal leakage

The simulation results in different cylinder internal leakage are shown in figure 7, 8 and 9:

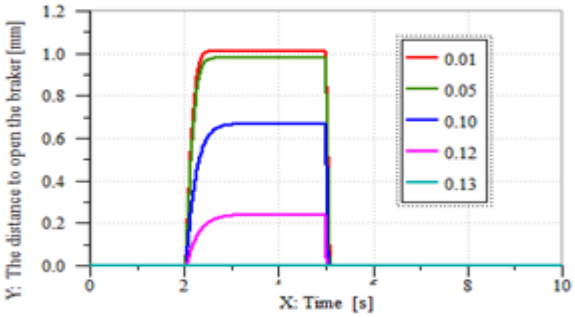


Figure 7 The brake clearance in different leakage

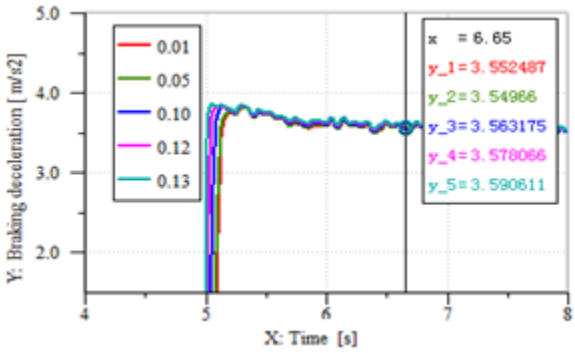


Figure 8 Braking deceleration in different leakage

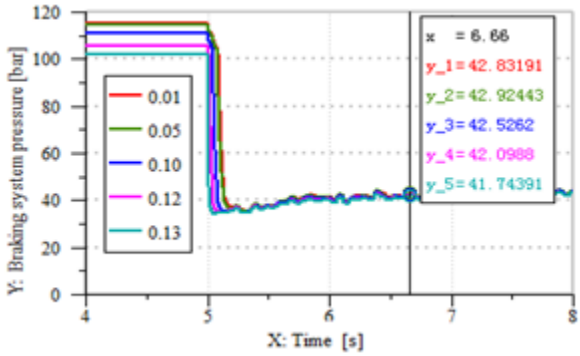


Figure 9 Braking system pressure in different leakage

We can see from the simulation diagram. When cylinder internal is leaked, brake deceleration meets the requirements of the system deceleration. The minimum pressure value of the braking system can meet braking momentum requirement. When the leakage clearance reaches 0.13 , the brake becomes closed, and the failure of cylinder internal leakage occurs.

4. FAULT DIAGNOSIS

We can obtain from the simulations of three main parts performance decline that the time - clearance curve and time-pressure curve of the brake system contain abundant fault information, from which feature parameters can be extracted for fault diagnosis. However, there are many pairs of brakes in the braking system, and the brake clearances are not same. The pressure of brake system can be measured by pressure sensor with highly measurement accuracy, so the pressure of brake system can be used to represent the overall performance of the braking system. In this paper, the time-pressure curve of the system is used to diagnose the fault of the braking system.

4.1 Generate Training Set / Testing set

The simulation platform is used to collect the brake system pressure data of 2-8.2 s as a set of fault data. Each fault is simulated by 30 sets of data, 25 of which are used as training samples, and the other 5 sets as test samples. Then there are 75 sets of training data of 620 dimensions, and the 15 sets of testing data of 620 dimensional. Use these data to extract its 2~7 order statistical

moments $m_2 \sim m_7$ and two percentile of p_{50} and p_{51} as characteristic parameter. After normalization, the BP neural network is used to analyze and identify the faults. The normalized training samples are shown in Tabel 1, and the normalized testing samples are shown in Tabel 2.

Table 1 The neural network training samples

Sample number	Network input								Network output	Corresponding fault
	p50	p51	m2	m3	m4	m5	m6	m7		
1	0.0532	0.3581	-0.7939	0.6595	-0.7736	0.5947	-0.6873	0.4829	001	Friction coefficient Reduce
2	0.0103	0.2926	-0.7756	0.6750	-0.7659	0.6140	-0.6910	0.5094	001	
3	-0.9712	-0.9716	-0.9255	0.9976	-0.9523	0.9965	-0.9681	0.9916	001	
…	…	…	…	…	…	…	…	…	…	
26	0.1871	0.8374	0.3677	0.7750	0.1999	0.8395	0.0402	0.8577	010	Spring fatigue
27	0.1890	0.8374	0.2703	0.7796	0.0911	0.8409	-0.0726	0.8545	010	
28	0.1833	0.8374	0.5708	0.7558	0.4428	0.8194	0.3121	0.8409	010	
…	…	…	…	…	…	…	…	…	…	
73	0.6109	0.6216	0.4149	-0.9577	0.6187	-0.9297	0.8125	-0.9043	100	Cylinder Leakage
74	0.5997	0.6058	0.3969	-1.0000	0.6204	-0.9913	0.8349	-0.9812	100	
75	0.5735	0.5830	0.3798	-0.9953	0.6160	-1.0000	0.8406	-1.0000	100	

Table 2 The neural network testing samples

Sample number	Network input								Corresponding fault
	p50	p51	m2	m3	m4	m5	m6	m7	
1	-0.7190	-0.4777	-0.8159	0.8410	-0.8425	0.8106	-0.8415	0.7538	friction coefficient Reduce
2	0.0419	0.4122	-1.0000	0.5504	-0.9016	0.4247	-0.7608	0.2017	
…	…	…	…	…	…	…	…	…	
6	0.1307	0.8294	1.0000	0.6699	1.0000	0.7380	1.0000	0.7594	Spring fatigue
7	0.1446	0.8336	0.7750	0.6538	0.7125	0.7163	0.6445	0.7380	
…	…	…	…	…	…	…	…	…	
11	0.8157	0.8221	-0.2086	0.1158	-0.3030	0.2941	-0.3876	0.3780	Cylinder body Leak
12	0.7398	0.7464	0.0962	-0.3557	0.0547	-0.1740	-0.0148	-0.0387	

4.2 Training and Testing BP Neural Network

The three-layer BP neural network with only one hidden layer can approach any nonlinear function. Therefore, the single hidden layer neural network is chosen in this paper. According to the number of input characteristic parameter, the number of input neurons is chosen to eight. According to the dimension of the output vector, the number of output neurons is chosen to three. And according to the experience, the number of neurons in the hidden layer is choosen to eighteen. The iterations is set to 1000 times, the training accuracy is set to 0.01, and the rest of the parameters use the default value. After finishing the network training, the test data is input for testing, test results are shown in Tabel 3

Table 3 The result of neural network diagnosis

Sample number	Network output			Diagnosis result
1	0.0001	-0.0004	1.0003	friction coefficient Reduce
2	0.0104	-0.0074	0.9970	
...	...	...	...	
6	-0.0005	1.0068	-0.0063	Spring fatigue
7	-0.0005	0.9935	0.0070	
...	...	...	...	
11	1.0005	-0.0034	0.0030	Cylinder body Leak
12	1.0009	-0.0173	0.0164	

It can be seen from the results that the first five fault samples were diagnosed as friction coefficient decreases, the middle five fault samples as spring stiffness decreases, the last five fault samples as cylinder internal leakage. The diagnosed fault type is consistent with the testing fault categories, indicating that the diagnosis method is accurate and effective.

5. CONCLUSION

Firstly, typical faults were simulated on the braking system simulation platform in this paper. It is concluded that the curve of the system oil pressure contains abundant fault information. Then a new method is proposed, in which hoist braking system fault characteristic parameters is extracted, namely statistical moment method. Using this method the characteristic parameters of system pressure-time curve were extracted. Finally, an accurate fault diagnosis of the braking system was obtained using BP neural network. The relationship between the braking system components performance degradation and the overall performance was investigated. It provides some theoretical basis for the establishment of the performance degradation model of the brake system of high speed deep well hoist, and for the intelligent diagnosis and operation maintenance of the hoist. The advantages of this method are as follows:

1) By using the continuous signal collected by the pressure sensor, the characteristic parameters of the fault diagnosis can be obtained accurately, and the problems that the signal is not synchronized and the characteristic parameters can not be obtained accurately is overcome.

2) According to the theory of invariants moment in image processing, the method of characteristic parameters extraction is proposed for the first time.

3) The whole performance of the braking system can be represented only by the signal collected by the pressure sensor, which can not only make fault diagnosis, but also the performance analysis of the braking system

4) The pressure sensor has the advantages of flexible installation position, convenient detection and replacement in time. The problem overcomes that the braking positive pressure sensor is installed inside the brake and is not easy to be replaced.

ACKNOWLEDGEMENTS

The authors would like to thank the National key research and development plan, "Basic theory and key technology of coal mine deep well construction and hoist" (Grant No.2016YFC0600900)

REFERENCES

[1] DU Bo, ZHANG Bu-bin & FENG Hai-ping, (2016) “Development status and tendency of mine hoisting equipmentst”, Journal of mining machinery, Vol. 44, No. 6, pp1-7.

[2] Alfred Carbogno, (2002) Mine hoisting in deep shafts in the 1st half of 21st Century . Acta Montanistica Slovaca.

[3] Ge Shi-rong et al., (1994) Reliability technology of mine hoist, Jiangsu: China University of Mining and Technology Press.

[4] Vilhelm Itu, Mihai Carmelo Ridze et al., (2008) “On diagnosis of brake mechanism of hoisting machines”, WSEAS International Conference on ENGINEERING MECHANICS, STRUCTURES, ENGNEERING GEOLOGY(EMESEG ‘08’) No. 7, pp:22-24.

[5] LIU Jing-yan, WANG Fu-zhong & LI Yu-dong, (2016) “Fault Diagnosis of Hoist Braking System Based on Neural Network Optimized by Particle Swarm”, Control Engineering of China, Vol.23, No. 2, pp294-298.

[6] Zhang Geng-yun, (2011) “Visual fault diagnosis of mine hoist brakes base on SOM network”, Journal of mining machinery, Vol. 39, No. 04, pp48-52.

[7] Li Juan-li & Yang zhao-jian, (2013) “Fault diagnosis method for mine hoist based on ontology”, Journal of vibration ,Measurement & Diagnosis, Vol. 33, No. 6, pp993-997.

[8] Li Juan-li & Yang Zhao-jian, (2014) “An uncertain reasoning method of hoist fault diagnosis”, Journal of China Coal Society, Vol. 39, No. 3, pp586-592.

[9] Zhang Qiang, Hu Nan & Li Hong-feng, (2016) “Fault diagnosis of the Mine hoist brake based on GA- BP neural network” Journal of Liaoning Technical University(Natural Science), Vol. 35, No. 2, pp155 -159.

[10] Wang Jian, Wang Shao-jin, LI Juan-li,et al., (2012) “Application of Information Fusion Technology in Fault Diagnosis for Mine Hoister Braking System”, Coal Mine Machinery, Vol. 33, No. 07, pp247-249.

[11] DONG Li-fang, SUN Wei, ZHAO Jun, et al., (2010) “Fault Diagnosis of Mine Hoist Braking System Based on Support Vector Machine”, Mechanical Engineering & Automation, Vol. 159, No. 2, pp124-126.

[12] GUO Xiao-hui & MA Xiao-ping, (2006) “Mine Hoist Braking System Fault Diagnosis Based on a Support Vector Maehine”, Journal of China University of Mining & Technology, Vol. 35, No. 6, pp813-817.

[13] Wang Zheng-you & Liu Ji-lin, (2003) “Information-fusion-based Fault Diagnosis of Hoist Braking System” Journal of China Coal Society, Vol. 28, No. 6, pp650-654.

[14] Lian Rui, Xu Zheng-guo & Lu Jian-gang, (2013) "Fault Monitoring Methods for Disc Brake System of Mine Hoist" Safety in Coal Mines, Vol. 44, No. 09, pp131-133.

[15] Fault treatment and technical reformation of mine hoist editorial board, Fault treatment and technical reformation of mine hoist, Beijing: Machinery Industry Press.

[16] Li Juan-juan, Hu Liang, Meng Guo-ying,Xie Guang-ming,et al., (2017) "Fault and Performance Research in Constant Deceleration Braking System of Mine Hoist Based on AMEsim" Industry and Mine Automation, Vol. 43, No. 8, pp53-60.

[17] Li Yu-jin & Kou Zi-ming, Basic theory of mine hoisting system, Beijing: Coal Industry Press.

[18] Rong Guan-ao, Computer image processing, Beijing: Coal Industry Press.

AUTHOR

Li juanjuan (1976 -), senior engineer, PhD candidate, major: mechanical electronics,

the research direction : failure diagnosis, E-mail: 673958678@qq.com

INTENTIONAL BLANK

CODEBOOK DESIGN STRATEGIES FOR MMO-OFDM DOWNLINK SYSTEMS WITH ZF-BF

Najoua ACHOURA[1] and Ridha BOUALLEGUE[2]

[1]Department National Engineering School of Tunis,
Innov'Com Laboratory, Tunisia
[2] High School of Communication of Tunis, SUP'COM,
Innov'Com Laboratory, Tunisia

ABSTRACT

Using the Channel State Information (CSI) at the transmitter is fundamental for the precoder design in Multi-user Multiple Input Single Output (MU-MISO-OFDM) systems. In Frequency Division Duplex (FDD) systems, CSI can be just available at the transmitter through a limited feedback channel [1], where we assume that each user quantizes its channel direction with a finite number of quantization bits. In this paper, we consider a scalar quantization (SQ) scheme of the Channel Direction Information (CDI). Although vector quantization (VQ) schemes [2], [3] still outperform this scalar scheme in terms of quantization error and Sum rate, the former scheme suffers from an exponential search complexity and high storage requirements at the receiver for high number of feedback bits.

KEYWORDS

MIMO-OFDM, zero-forcing beamforming, RVQ, Scalar Quantization

1. INTRODUCTION

Currently, higher data rate is preferred to supply high quality multi-media services. Multiple-input multiple-output (MIMO) technology has attracted much more attentions since MIMO wireless channels, created by exploiting antenna arrays at transmitter and receiver, promises high capacity and high quality wireless communication links. It is well-known that with full channel state information (CSI) at the transmitter (CSIT), employment of precoding techniques can improve its capacity [4], [5]. This implies that the transmitter requires some form of knowledge on the wireless channel conditions. In time division duplexing (TDD) systems, reciprocity can be employed to explore CSI and make CSI available at transmitter. However, employing reciprocity in frequency division duplexing (FDD) systems has been impossible since the forward and reverse links in FDD generally have highly uncorrelated channels. The adaptation of feedback makes instantaneous CSIT possible. In practical systems, the receiver estimates the channel conditions based on the pre-defined reference signals known by both transmitter and receiver. After channel estimation, the receiver sends the estimated channel information back to the transmitter side, and the transmitter uses this information to adapt the forward link transmission. The adoption of feedback can improve the system performance, such as increasing capacity and reducing the failure rate of data transmission, which cannot be handled by the receiver alone. However on the other hand, the feedback information itself occupies some frequency resource

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 13– 18, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71202

and decreases the spectrum efficiency of system. That's why limited feedback is necessary in practical systems over limited bandwidth feedback channels. There are many works dealing with limited feedback [6]. One of them is generating limited feedback based on the pre-defined codebook, which is known by both transmitter and receiver. Once the receiver obtained the CSI by channel estimation, it checks the codebook to figure out the quantization partitions, and represent all the channel information located in these partitions by the corresponding codewords. After finding those codewords from codebook, the receiver sends the codebook index corresponding to these codewords back to the transmitter instead of sending full CSI back. At the transmitter side, the transmitter chooses the codewords from the identical codebook based on the feedback information from receiver. Usually, the systems suffer lower burden to send codebook index rather than feedback full CSI.

In this situation, the strategies of generating codebook become the key issues. In this paper, we propose two codebook generation strategies for MISO beamforming systems. The first strategy is based on scalar quantization, which has lower computational complexity compared to high dimensional (more than 2 dimension) vector quantization strategies.

2. SYSTEM MODEL

The downlink or broadcast system is described as follows. The base station with N_t antennas transmits data simultaneously to N_t active users chosen from a total of U users, each with one receive antenna. The base station separates the multiuser data streams by beamforming, i.e. assigning a beamforming vector to each of the N_t active users. The beamforming vectors $\{w_n\}_{n=1}^{Nt}$ are selected from multiple sets of unitary orthogonal vectors following the beam and user selection algorithm. Equal power allocation over scheduled users is considered. The received signal of the u^{th} scheduled user is expressed as:

$$y_u = \sqrt{\frac{P}{Nt}} h_k^* \sum_{n \in A} w_n x_n + v_u \quad u \in A \tag{1}$$

Where we use the following notation:

N_t :number of transmit antennas and also number of scheduled users; h_u (N_t×1 vector) downlink channel; x_u transmitted symbol with $E[|xul^2] = 1$; y_u received symbol; w_u (N_t×1 vector) beamforming vector, A The index set of scheduled users, P :transmission power;

3. ZERO-FORCING BEAMFORMING

Zero-forcing (ZF) decomposes the channel into several parallel scalar channels with only additive noise, and the interference is removed completely by transmit beamforming techniques[12]. Suppose a base station with Nt transmit antennas transmits information to K users, each user is equipped with one single receive antenna. The received signal by the kth user is

$$y_k = h_k^H v_k x_k + h_k^H \sum_{j=1, j \neq k}^{K} v_j x_j + n_k \qquad 1 \leq k \leq K \tag{2}$$

ZF method transmits the signals towards the intended user with nulls steered in the direction of the other users, i.e., ($h_j^H v_k = 0 \ \forall j \neq k$) .The users will receive only the desired signal without any interference because of the perfect nulling. In this case, the received data at the k^{th} user can be written

$$y_k = h_k^H v_k x_k + n_k \tag{3}$$

The corresponding vector equation is

$$y = H^h V x + n \tag{4}$$

Therefore, if the normalized transmit beamforming vector of the k[th] user is selected

$$v_k' = \frac{h_k^{(\dagger)}}{\sqrt{\left\| h_k^{(\dagger)} \right\|_F^2}} \tag{5}$$

where h(y)k is the k[th] column of the pseudo inverse of H, denoted as H(y). Then it is shown that the interference can be canceled completely. In this case, the SNR of the kth user is

$$SNR_k = \frac{\left| h_k^H v_k' \right|^2 p_k}{\sigma^2} \tag{6}$$

4. SCALAR QUANTIZATION

We assume that our system is a multiuser MIMO (MU-MIMO) downlink. As shown in Figure 1, there is N_t transmit antennas at base station (BS) side, and Nr mobile stations (MS) communicate with BS simultaneously. Each mobile system is equipped with a single antenna, and the relationship $N_r \leq N_t$ holds in this system.

Since the base station only needs the spatial direction of the channel to eliminate the interference, the channel matrix H is normalized and then quantized using Scalar quantization at each user. The real and imaginary parts of each complex element h_{ij} in H are quantized to B_S bits, respectively

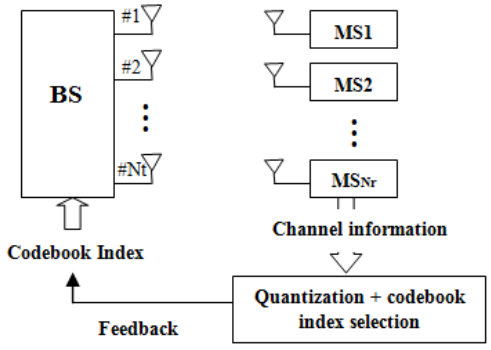


Figure 1. Block diagram of MU-MIMO system with limited feedback

Note that one bit is reserved for the sign of each of the real and imaginary parts. In [7], the quantized version of h_{ij} is $h_{ij}^{'} = \frac{1}{2^{B_S-1}}\left[\frac{h_{ij}}{m}(2^{B_S-1}-1)\right]$, where m is a scaling ratio which guarantees the real/imaginary element in the normalized channel matrix is always less than or equal to one. In particular, m can be chosen as the maximum value among all real and imaginary elements of the channel matrix H. Therefore, the number of feedback bits needed at each user is $B = 2N_t N_r B_s$.

A closed-form expression for the relationship between the rate loss and the number of feedback bits is usually intractable. A feasible approach is to approximate the quantization error as a

random variable with a given distribution, e.g., uniformly distributed in$[-2^{-B_s-1}, 2^{-B_s-1}]$. For the sake of analytical simplicity, we assume the quantization error is a Gaussian random variable with zero mean and variance $\sigma_s^2 = \frac{1}{12}2^{-2B_s+2}$ [8]. As the sum rate is a function of the variance [9], by applying a similar approach as in Theorem 2 in [10], an approximation for the number of feedback bits required to maintain a performance gap of no more than 3 dB with respect to the case with perfect CSI is derived to be

$$B \approx 2N_t N_r \left(\frac{P_{dB}}{3} - \frac{1}{2} log \frac{N_t}{12} \left(2^{\frac{N_t}{N_r}} - 1 \right) \right) \quad (7)$$

Where $P_{dB} = 10log_{10}P$ is the normalized transmit power in units of dB.

5. CHANNEL VECTOR QUANTIZATION

In [11], authors analyzed the channel capacity with perfect channel knowledge at the receiver, but with limited channel knowledge at the transmitter. Specifically, the optimal beamformer is quantized at the receiver, and the quantized version is relayed back to the transmitter. Given the quantization codebook $C = \{w_1, \ldots., w_{2^B}\}$, which is also known a priori at the transmitter, and the channel H, the receiver selects the quantized beamforming vector to maximize the instantaneos rate, [11]

$$w(H) = \arg\max_{w_j \in V} \left\{ \log(1 + \rho \left\| H w_j \right\|^2) \right\}$$

Where $\rho = 1/\sigma_n^2$ is the background signal-to-noise ratio (SNR). The (uncoded) index for the rate-maximizing beamforming vector is relayed to the transmitter via an error-free feedback link. The capacity depends on the beamforming codebook V and B. With unlimited feedback (B→ ∞) the w(H) that maximizes the capacity is the eigenvector of H*H, which corresponds to the maximum eigenvalue.

We will assume that the codebook vectors are independent and isotropically distributed over the unit sphere. It is shown in [12], that this RVQ scheme is optimal (i.e., maximizes the achievable rate) in the large system limit in which (B, N_t, N_r) →∞ with fixed normalized feedback B = B/N_t and $N_r = N_r/N_t$. (For the MISO channel $N_r = 1$). Furthermore, the corresponding capacity grows as log(ρN_t), which is the same order-growth as with perfect channel knowledge at the transmitter. Although strictly speaking, RVQ is suboptimal for a finite size system, numerical results indicate that the average performance is often indistinguishable from the performance with optimized codebooks.

6. SIMULATIONS RESULTS

In Figure 2, the sum rate is plotted versus SNR for the ZF-BF technique by considering the case of a perfect CSIT in comparison with the use of vector quantization RVQ for B=8, 16 and 32 bits, the number of transmit antennas is N_t=4. When the SNR is low, the limited feedback reacts almost as well as the zero-forcing with full CSI. However, when the SNR increases, the feedback system is limited by interference and the rate converges to a threshold limit based on the amount of the multi-user interference.

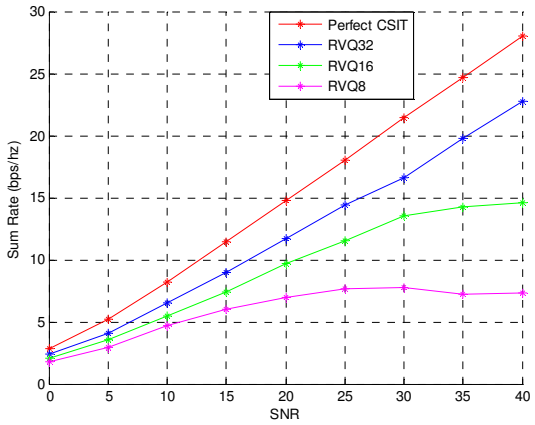


Figure 2. Comparison of the ZF-BF method using RVQ for different sizes of feedback

In Figure 3 the sum rate with RVQ quantization, as well as perfect CSI case for the same number of users (i.e. users T_{fb}/B) are plotted as a function of B for a system with $N_t = 4$, $T_{fb} = 300$ bits and SNR=15dB. We note that for $B \geq 25$ the RVQ perfectly approaches the case of perfect CSIT. As a result, increase the size of feedback beyond 25 bits no sense obviously because it reduces the number of users, but does not provide a measurable benefit. In the same figure, and considering the case of a scalar quantization, although we note that it provides a sum slightly below the RVQ. However, it is still interesting to work with a large B, which is generally the preferred operating point, even with a dictionary suboptimal quantization, and performance with scalar quantization is still competitive.

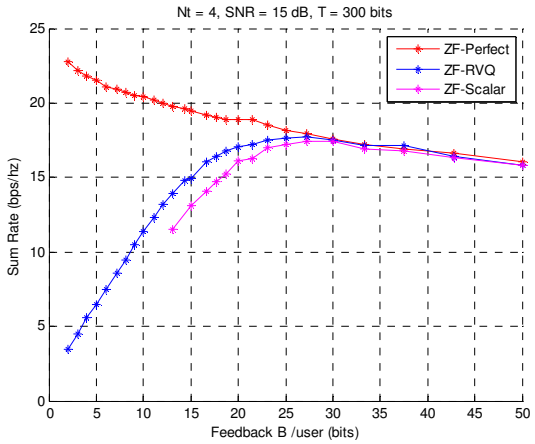


Figure 3. ZF-BF with RVQ and scalar quantization vs user feedback size

3. CONCLUSIONS

This paper presents two strategies for construction of codebook for the processes of quantization in MU-MIMO-OFDM systems. Using zero forcing as technique of beamforming, the simulation results show that the simple codebook generation strategies based on scalar quantization are efficient, and they require smaller computational burden than other codebook generation algorithms based on higher dimensional vector quantization. Additionally, these simple scalar quantization strategies, by choosing an appropriate number of feedback bits, are also efficient for MIMO systems employing precoding techniques.

REFERENCES

[1] D. J. Love, R. W. Health Jr., W. Santipach, and M. L. Honig, ‖What is the Value of Limited Feedback for MIMO Channels?‖, in IEEE Commun. Mag., vol. 42, no. 10, pp. 54-59, Oct. 2004

[2] C. K. Au-Yeung and D. J. Love, ‖On the Performance of Random Vector Quantization Limited Feedback Beamforming in a MISO System‖, in IEEE Trans. Wireless Commun., vol. 6, no. 2, Feb. 2007.

[3] J. C. Roh, B. D. Rao, ‖Transmit Beamforming in Multiple-Antenna Systems With Finite Rate Feedback: A VQ-Based Approach ‖, in IEEE Trans. on Information Theory, vol. 52, no. 3, March 2006.

[4] B. Khoshnevis and W. Yu, ―Limited Feedback Multi-Antenna Quantization Codebook Design–Part I: Single-User Channels,‖Submitted to IEEE Transactions on Signal Processing, March2010.

[5] B. Khoshnevis and W. Yu, ―Limited Feedback Multi-Antenna Quantization Codebook Design–Part II: Multiuser Channels,‖Submitted to IEEE Transactions on Signal Processing, March 2010.

[6] D. J. Love, and R. W. Heath, ―Limited feedback diversity techniques for correlated channels,‖ IEEE Trans. Vehicular Tech., vol. 55, no. 2, pp. 718-722, Mar. 2006

[7] Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications,‖ IEEE Std 802.11ac/D4.0, Nov. 2012.

[8] Qi Wang, Hao Feng, Lenoard J. Cimini, Larry J. Greenstein, Douglas S. Chan and Ahmadreza Hedayat, "Comparison of Quantization Techniques for Downlink Multi-User MIMO Channels with Limited Feedback," IEEE Wireless Communications Letters, vol. 3, no. 1, pp. 165-168, Feb 2014.

[9] Q. Wang, H. Feng, L. J. Cimini, L. J. Greenstein, D. S. Chan, and A. Hedayat, ―Sparse coding quantization for downlink MU-MIMO with limited channel state information feedback,‖ in Proc. 2013 IEEE MILCOM.

[10] N. Ravindran and N. Jindal, ―Limited feedback-based block diagonalization for the MIMO broadcast channel,‖ IEEE J. Sel. Areas Commun., vol. 26, no. 8, pp. 1473–1482, Oct. 2008.

[11] W. Santipach, M.L. Honig ―Optimization of Training and Feedback Overhead for Beamforming over Block Fading Channels‖ IEEE Trans. Info. Theory, August 2009

[12] G. Caire, N. Jindal, M. Kobayashi, and N. Ravindran, Multiuser MIMO Achievable Rates with Downlink Training and Channel State Feedback, IEEE Trans. Information Theory, Vol. 56, No. 6, pp. 2845-2866, June 2010

TOWARDS ESTABLISHING A CATALOGUE OF PATTERNS FOR ARCHITECTING MOBILE CLOUD SOFTWARE

Aakash Ahmad, Ahmed B. Altamimi, Abdulrahman Alreshidi

College of Computer Science and Engineering,
University of Ha'il, Ha'il, Saudi Arabia

ABSTRACT

Mobile computing empowers its users to exploit portable computation and context-aware communication, however; a mobile device lacks energy and performance to execute computation and memory intensive tasks. On the contrary, cloud computing exploits the 'pay-per-use' software and hardware services to provide virtually unlimited processing and storage resources. The unification of mobile and cloud computing as Mobile Cloud Computing (MCC) enables mobility and context awareness with computation and storage services to provide systems that are portable, yet resource sufficient. In an architectural context for MCC systems that require context-awareness, mobility and scalability, etc., there is a need to capitalise on reusable solutions – utilising patterns and best practices – to architect and develop mobile cloud software. This research aims to build and exploit a catalogue of patterns that support reusable design knowledge for architecture-based development of the MCC systems. We have discovered some patterns as generic and reusable solutions and demonstrate their usage in the context of mobile cloud systems. The proposed research aims to establish the catalogue as patterns repository – facilitating a continuous discovery and documentation of new patterns overtime that support reusable knowledge and practices to develop MCC systems.

KEYWORDS

Patterns and Frameworks, Software Architecture, Software Reuse, Cloud Computing, Mobile Cloud Systems.

1. INTRODUCTION

Mobile computing has emerged as a pervasive technology by exploiting anywhere, anytime – portable, context-sensitive and connected – mobile devices [1]. Such devices with embedded sensors (hardware) and freely available mobile apps (software) have empowered users to perform a variety of tasks ranging from mobile commerce to health and fitness monitoring [2]. However, a mobile device is considered as a resource-constrained computer that lacks the energy, efficiency and quality of service for computation and memory-intensive tasks [3, 4]. Cloud computing model have proven to be a success for provisioning/de-provisioning of the 'pay-per-use', on-demand and virtually unlimited hardware and software resources [5]. The unification of mobile and cloud as Mobile Cloud Computing (MCC) represents the state-of-the-art mobile computing technology that aims to minimize the resource poverty of mobile devices by exploiting the

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 19– 33, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71203

resource sufficient cloud-based software and hardware services [6, 7]. Despite the benefits of MCC technology, a number of challenges must be addressed while architecting and developing MCC systems [1, 2, 6]. Moreover, a rapid demand for developing mobile cloud based software requires a number of highly knowledgeable and experienced architects who may not be widely available as MCC has recently emerged as an innovative technology [2]. Architectural styles and patterns have been used for providing packaged knowledge about well-known design solutions to both experienced and novice architects [8, 9]. Specifically, patterns embody proven solutions to recurring problems by capturing concentrated wisdom of many practitioners and consolidated design knowledge from multiple systems [10]. In recent years, pattern-based approaches resulted in (i) promoting reuse while (iii) decreasing the efforts required during architectural design and evolution processes [8, 11]. In addition, pattern-oriented solutions [10] enhance quality by applying the best practices and knowledge to resolve recurring problems of architectural design [4].

In this research, we aim to support pattern-based architecting for MCC software that can accelerate the process of gaining knowledge and experience in successfully modeling and evolving the system's structure and behavior at higher abstractions [3, 7]. Specifically, we focus on building and exploiting the catalogue as a collection of architectural patterns that promote reuse of design knowledge for architecting MCC systems that is currently lacking in existing research [3, 4, 12]. We model and utilise an MCC architectural pattern as a generic and repeatable solution to recurring problems of architectural design. While architecting MCC systems, one exploits dynamically composed services to develop systems that are portable, context-sensitive and efficient [4, 5, 7]. In comparison to the more traditional (object-oriented, component-based and service-driven) systems [8, 9], patterns for mobile MCC architectures are characterized by specific requirements such as mobility, context-sensitivity for (front-end) mobile computing with service composition, and scalability of (back-end) cloud services.

One of the key challenges in providing pattern-based architectural knowledge is systematic discovery and detailed documentation of patterns as reusable packages of known solutions to recurring problems [8, 10]. In this paper, we report (i) our effort towards establishing a catalogue of architectural patterns for MCC applications; and (ii) demonstrate how the discovered architectural patterns can be applied to architect a mobile cloud system. Pattern discovery is a continuous process that requires frequent mining of pattern sources to discover and document new patterns that emerge over-time [10, 18]. Our approach consisted of three simple steps including: pattern discovery, pattern documentation, and pattern application. With regards to the existing research in [4, 5, 6, 7], our research aims to contribute:

- Systematic discovery of patterns that address architectural aspects of mobile cloud software. The discovered patterns provide a foundation to establish a pattern catalogue as a repository of reusable solution and best practices to design and architect the MCC systems.

- Exploit the pattern catalogue with discovered patterns as elements of reuse knowledge that guides a step-wise process of pattern-driven and reuse-oriented architecting of MCC systems.

This research reports our preliminary efforts to discover and document patterns. The futuristic research aims to support a (semi-) automated discovery of patterns with user's decision support in

the pattern discovery process. In addition to the discovery of new patterns, case study based evaluation are planned to evaluate the applicability of pattern-based architecting.

The rest of this paper is organized as follows. Section 2 presents the related research. Section 3 discusses the research methodology and proposed solution. Section 4 highlights the reference architecture for mobile-cloud systems. Section 5 presents the discovered patterns and their documentation. Section 6 discusses some threats to validity and futuristic dimensions of the research. Section 7 concludes the paper.

2. RELATED RESEARCH

In this section, first we discuss the existing research on (i) patterns for mobile and cloud computing architectures, and then highlight some (ii) reference architecture and patterns for mobile cloud based systems. The discussion of the related research here highlights the research state-of-the-art and justifies the needs and scope of the proposed work.

2.1. Patterns[1] for Mobile Cloud Computing Architectures

Cloud Computing – One of the thorough work on cloud architecture patterns [13] reports best practices for scalability, big data, fault handling and distributed services on Windows Azure (Platform as a Service: PaaS). Patterns in [13] provide guidelines and practical solutions to address the scalability and elasticity in cloud-native applications for Windows Azure platform. Also, the work reported in [10] provides a collection of patterns for development of cloud service models (SaaS, PaaS, IaaS), and their deployment using cloud deployment models (private, public, hybrid, community). The patterns are organized in a framework to guide developers to systematically select and apply these patterns. The work in [15, 16] reports a community-driven development of patterns. Specifically, the *Cloud Computing Design Patterns* in [15] represent a collection of 39 patterns to address the scalability, reliability, security and monitoring issues of cloud applications. Also, the *Cloud Design Patterns* [16] present a collection of patterns created by various (cloud) architects based on the type of problems, and their generic design patterns as repeatable solutions.

Mobile Computing – Compared to the research on cloud computing patterns, there is less work on pattern-based designing and architecting of mobile computing solutions. In [4], three architectural patterns are presented for mobile computing in the context of tactical-edge resource-constrained environments that support first responders and military personnel operating in edge environments. The proposed architecture patterns namely *data source integration, group-context awareness,* and *cyber foraging* are driven by flexibility, resource efficiency, and usability, which are key quality attributes for systems at the tactical edge. The proposed patterns enable system architects to instantiate them using a variety of technologies that can meet functional and quality requirements while developing and operating mobile systems. In a similar work [17], a collection of architectural solutions namely mobility patterns are presented. The proposed mobility patterns

---

[1]The terms patterns and styles are often used interchangeably [10]. We focus on patterns that 'provide generic, reusable solution to recurring problems of architectural design' [8], while 'styles provide a constrained composition of elements for architectural organization and restructuring' [9].

have been derived from successful mobile applications and allow a designer to reuse design elements as building blocks for mobile applications.

In contrast to the existing research [10, 13, 4, 17], the proposed patterns in our catalogue are focused on architecture-centric development, execution, and management aspects of MCC systems. Moreover, our work aims to establish a pattern catalogue - continuously evolving pattern repository - based on an incremental discovery and specification of new architectural patterns guided by [8, 14, 18].

2.2. Reference Architectures and Patterns for MCC Systems

A number of reference architecture [3, 19] and a pattern-based solution for mobile service oriented architecture (SOA) [12] exist. The technical distinction between a reference architecture and architectural patterns are detailed in [20]. In [3], the authors have described a reference architecture that enables off-loading of (computation and memory-intensive) mobile code to cloud-based servers in the context of hostile environments. This paper also presents viable implementations scenarios along with architectural tradeoffs. A similar work [19] presents a highly-extensible reference architecture that enables group-context-aware mobile applications that integrates contextual information from individuals with that of nearby team members to enable context-driven team coordination. In the context of mobile SOA, a taxonomy of six architectural patterns namely*; standalone, full offloading, partial offloading, SaaS-based, CaaS-based*, and *offloaded CaaS-based* is presented in [12]. For each of the proposed architectural pattern, the authors have organized architectural patterns with key components and their interactions at runtime. Different architectural patterns exhibit various levels of qualities including the performance, efficiency and energy consumption for mobile applications.

In the sections above, we have presented an overview of the most relevant research to our proposed work. Specifically, the work [3, 12, 19] on reference architecture and patterns [20] is most relevant to our proposed solution. Based on an overview of the existing research and the current challenges for mobile computing [1, 2, 6], we claim that the proposed solution is a preliminary attempt to establish a catalogue that aims to support architecture-centric (reusable) development of MCC systems.

3. RESEARCH METHODOLOGY AND PROPOSED SOLUTION

In this section, first we discuss the methodology for discovering and documenting the patterns (Section 3.1). We than provide an overview of the solution for pattern-based process for architectural design (Section 3.2).

3.1. Methodology for Pattern Discovery and Documentation

Pattern discovery is based on design review method [21] by reviewing recurring design solutions to frequent problems of architecting MCC systems [6, 7]. The design review team was comprised of 3 members with experience of (a) conducting the systematic review [1, 15], (b) pattern mining [18], and (c) development of mobile and cloud systems. Our approach to discover and document the patterns consisted of the following steps.

Step I – *Conducting Systematic Review on Architectural Solutions for MCC Systems*: The review was conducted to investigate the recurring challenges, design problems and existing solutions to develop mobile cloud architectures [24]. A systematic review [22] is expected to minimise the potential bias in the review and has a protocol that guides the process. Based on the research questions (RQs) below and the review protocol, we selected 85 studies (problem-solution map) as primary sources of pattern discovery.

RQ1 – What methods/techniques/frameworks/solutions are provided in existing (research and practices) to model/develop/evolve MCC system architectures?

RQ2 – What are the patterns/styles/frameworks to support reusable design knowledge for architecting MCC systems?

Step II – *Identification of Pattern Data Sets*: Once the studies were identified, we extracted the data sets in Table 1 from selected studies. Datasets refers to mapping the existing architectural design problems and their solutions. For an objective evaluation, we derived 7 items in Table 1 (I1 to I7 – item collection is referred to as datasets) by following the guidelines in [21] and based on our experience with architecture pattern mining [18] and classification of reusable architectural solutions and patterns [23]. Items in Table 1 guided the pattern mining team to objectively review the problem (P) and solution (S) mapping, the attributes (A) that affect the solution and the occurrence frequency (T) of the repeatable solution by analyzing the pattern datasets. Once a decision (D) is reached, the results are documented as pattern elements (E) for a peer-review before finalization.

Step III - *Thematic Analysis to Investigate Pattern Datasets*: After identification of the datasets, thematic analysis as the final step helps to 'identify, analyse and report' patterns from datasets by following three steps. A theme is a possible solution, or method or a mechanism to resolve problems.

(1) Data Analysis process comprises of (a) analysing datasets, (b) to extract design attributes from problem-solution mapping (I5 in Table 1).

(2) Pattern Discovery process involves (a) searching of the recurring themes based on data analysis, and (b) reviewing the identified themes. To discover patterns, we reviewed studies and aimed at discovering design problems (I2) and their relate solutions (I3). We consider a recurring theme as a discovered pattern (I6).

(3) Pattern Documentation is the last process that includes (a) classification of related themes based on design attributes (I5) and documented them in a template (I7).

3.2. Solution Overview for Pattern-based Architecting

We propose pattern-driven architecting as a 3-step process with underlying activities and repositories in Figure 1. Pattern discovery involves *pattern discovery* and *pattern documentation* (in Section 5). Pattern documentation involves pattern classification (in Section 6). Pattern application involves *selection* and *instantiation* (in Section 7). If a designer finds suitable patterns from the catalogue then the first two steps are skipped

Table 1. Dataset Items for Pattern Discovery Process

ID	Items	Description
I1	Design Space (C)	All the available architectural design (C = 85 studies).
I2	Recurring Problem (P)	Repeatable problems existing in the design space (P ∈ C).
I3	Frequent Solution (S)	Solutions to repeatable problems in the design space (S ∈ C).
I4	Frequency Threshold (T)	Threshold for occurrence of S to be discovered as a pattern
I5	Design Attributes (A)	Attributes affecting S (A = 08) *Context Awareness, Mobility, Computational Efficiency, Energy Efficiency, Service Reliability, Service Availability, Data Storage, Data Processing*
I6	Discovered Pattern (N)	2 = *Yes*, 1 = *Not sure* (consensus required), 0 = *No*
I7	Pattern Elements (E)	Elements of Pattern Description (E = 9) *Name, Intent, Problem (P), Solution (S), Impact, Origin, Uses, Reference Diagram, Architecture Elements, Constraints*

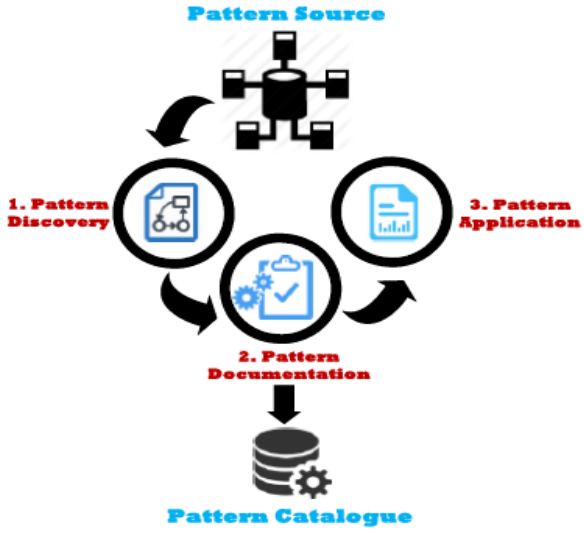


Figure 1. A Three Step Processes for Pattern-driven Architecting

4. REFERENCE ARCHITECTURE AND CHARACTERISTICS FOR MCC SYSTEMS

We now present the reference architecture for the MCC systems (Section 4.1) and also discuss the characteristic of the MCC architectures (Section 4.2)

4.1 A Reference Architecture for MCC Systems

In Figure 2 a) we illustrate the reference architecture for the mobile cloud systems that consists two distinct layers of architectural component, (i) Cloud Computing Layer (resource sufficient - back-end), and (ii) Mobile Computing Layer (portable and context-aware - front-end). The

discussion and presentation of the reference architecture is vital before presenting the architectural patterns and their presentation [20]. Specifically, the reference architecture in Figure 2 a) acts as a blueprint or simply a reference to derive advanced architectural solutions. Further details about the reference architectures, architectural solutions and architectural patterns are provided in [3, 4, 20].

(1) *Portable and Context-Aware and User Interface Layer* acts as a front-end that allows a mobile user to manipulate the data by exploiting the context and location information of the mobile device. However, the front-end mobile device lacks computation and storage-intensive resources.

(2) *Resource Sufficient and Elastic Cloud Layer* acts as a back-end to the off-loaded data by mobile device that allows scalable and virtually unlimited storage and processing resources [2, 7]. As in Figure 2, the mobile-cloud computing can empower its users by unifying the features such as context-sensitivity, location-awareness and mobility at the mobile computing layer, with virtually unlimited computation and storage resources of cloud computing. However, such system integration and operation requires a continuous network connectivity and involves latency along with security and privacy of data that communicates between mobile and cloud.

The intra-layer and inter-layer communication among the components is enabled by means of architectural connectors [10, 23]. For example, Figure 2 b) that represents a simplified (component and connector-based) architectural view of a system that invokes location-based services. In the component and connector based architectures the components acts as computational elements or data stores that are connected to each other using the connectors. Specifically, the *getLocationService* component (on a mobile device) requests for a location based service that is computed and provided by *sendLocationService* component on the (cloud server). Both these components are interconnected using *LocSrv* connector.

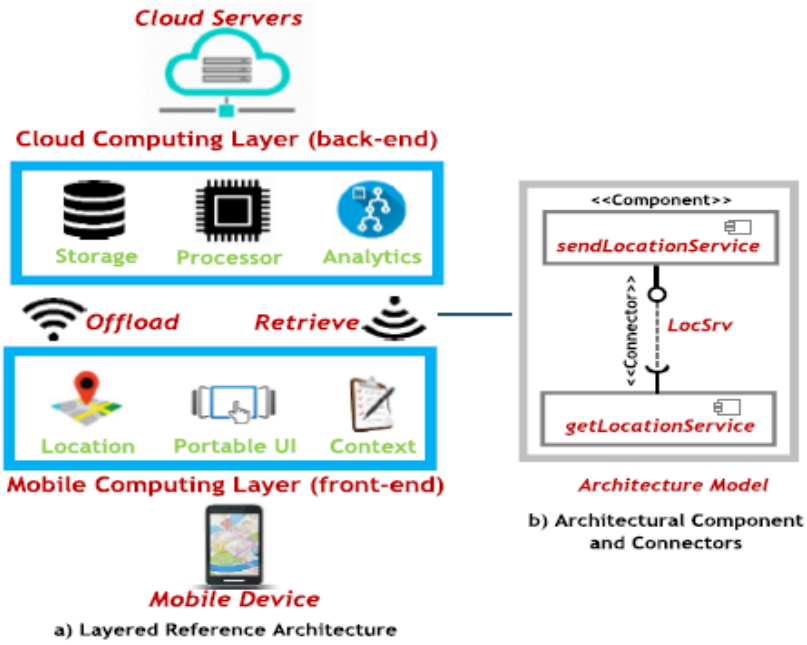


Figure 2. Reference Architecture for Mobile Cloud Systems.

4.2. Characteristics of Mobile Cloud Architectures

In the mobile cloud architectures, in addition to supporting the core functionality the characteristics of software quality are also vital for both the mobile computing and cloud computing. Specifically, for the mobile computing aspects of the MCC systems the main characteristics of the quality refer to the *context-awareness, mobility, efficiency* and *security of the mobile data* [1, 2]. In contrast to mobile, the cloud based systems exploit the principle of service-orientation that enables service composition as a foundation to develop cloud-based applications [15, 22]. In addition, central to cloud architectures are the quality of service (QoS) requirements that ensures composable services must satisfy the desired quality characteristics. These characteristics include but not limited to scalability, elasticity, multi-tenancy and virtualization of software services that distinguish the mobile cloud architectures [4, 7] from the traditional software architectures [9, 10]. Therefore, the patterns for traditional software development and architecture [9] cannot easily be applied to mobile cloud systems unless they support the above-mentioned characteristics specific to mobile cloud architectures. A single pattern may not ensure all these characteristics, however the pattern collection must try to address them all. For example, unlike the traditional architectures, mobile cloud based architectures are expected to serve context-aware multiple-tenants with each tenant having its own specific context and QoS requirement that can vary from performance and reliability to security aspects. Context-aware multi-tenant capabilities of MCC systems need to be considered not only at service but also at the platform and infrastructure [17, 3] level not addressed in existing patterns [15, 16].

5. CATALOGUE BASED DOCUMENTATION OF PATTERNS

In this section, we presents the pattern catalogue in terms of the individual patterns and their representation. A catalogue essentially documents and maintains the patterns as a repository of reusable architectural solution [10, 14, 15]. In the context of a catalogue (as a repository), the pattern template (structured representation) provides the necessary elements to capture and represent the individual patterns [10]. Due to space constraints, we only present the necessary elements of pattern template as per the guidelines from [8, 10] to document software patterns in a template. The pattern template provides a structured mechanism to capture individual elements of the patterns for storage, analysis and retrieval of the patterns effectively and efficiently. Our pattern template include:

- **Pattern Name** providing unique name for pattern,
- **Intent** describes the motivation or the known uses,
- **Design Problem and Solution** provides a mapping of the problem-solution view,
- **Architecture Elements** as component and connectors that are part of the pattern,
- **Reuse Design Knowledge** that is supported by the pattern,
- **Quality Characteristics** are the attributes of the quality that are affected by the pattern.
- **Reference Diagram** provides overview of the pattern also known as pattern thumbnail.

We now discuss the three discovered patterns namely Adaptive Mobile-Cloud Offloading, Mobile Cloudlets, Mobile Sensing and Cloud Analytics that are presented in a structured template in Table 2 – Table 4 respectively. We distinguish between pattern abstraction and pattern instantiation and explain the (i) applicability and uses of the pattern, and (ii) template-based documentation of patterns.

5.1. Pattern Abstraction and Pattern Instantiation

We use an example of one of the discovered patterns named Adaptive Mobile-Cloud Offloading to distinguish abstraction and instantiation of the pattern as in Figure 3.

(1). Abstraction for Pattern Modeling

Abstraction is vital to promote a pattern as a generic solution by abstracting the complex implementation specific details as in Figure 3 (a). As illustrated in Figure 3 (a), the abstraction of Adaptive Mobile-Cloud Offloading pattern is vital to help a pattern user (designer/architect) to analyze the high-level solution for off-loading computational and storage intensive data to the cloud-based servers. Pattern abstraction helps to analyze its impacts on architecture model before pattern application (preconditions) and architectural view when pattern is applied (post-conditions) to also promote pattern as an incremental design process.

(2). Instantiation for Pattern Application

Instantiation provides the necessary details in terms of concrete architectural elements to instantiate a pattern. This is also referred to as a pattern application by means of adding refinements – extending the abstract box and arrows with architectural components and connectors from Figure 3 (a) – to pattern abstraction in Figure 3 (b) [8, 10]. In Figure 3 (b), the instance of Adaptive Mobile-Cloud Offloading pattern utilizes the Data Access Bus to bind services (in Service Pool) to data collection (in Data Source) component.

Pattern I - Adaptive Mobile-Cloud Offloading

A) Pattern Intent: To enable a mobile device to dynamically determine what, how and where to off-load its data to enhance efficiency and of mobile computing

B) Design Problem: How to enable a (resource-constrained) mobile device to delegate its memory and computational-intensive data and tasks to (resource-sufficient) computers?

C) Solution: Integrate the off-loading logic between Mobile Computing and Cloud Computing Layers. Such an integrated logic enables a mobile device to exploit dynamic parameters – such as energy efficiency, computational overhead and storage requirements – to determine and off-load data and tasks to cloud computing servers.

D) Architecture Elements: Mobile Computing Layer with (resource constrained) mobile devices, Cloud Computing Layer (resource Sufficient) server are integrated with off-loading knowledge.

E) Reuse Design Knowledge: Integration of Off-loading logic/knowledge to delegate mobile computing data and tasks to cloud-based servers.

F) Quality Characteristics:

- Elasticity of cloud services (acquiring and releasing resources) based on dynamically determined off-loading.

- QoS-driven Offloading to ensure that dynamic parameters such as energy, storage and computational efficiency.

G) Reference Diagram: Figure 3 b) represents a concrete instance of the abstract pattern representation in 3 a). Specifically, Figure 3 b) illustrates a scenario of pattern application where mobiles devices are used as portable computers to capture contextual images that needs analytics and processing to gather information. The image processing must be delegated to the cloud servers that have image databases to match and process the image. The offloading logic is integrated between the mobile device and cloud server to enable a mobile device to selectively and dynamically off-load the images to the appropriate cloud-based server based on energy, computational or storage efficiency.

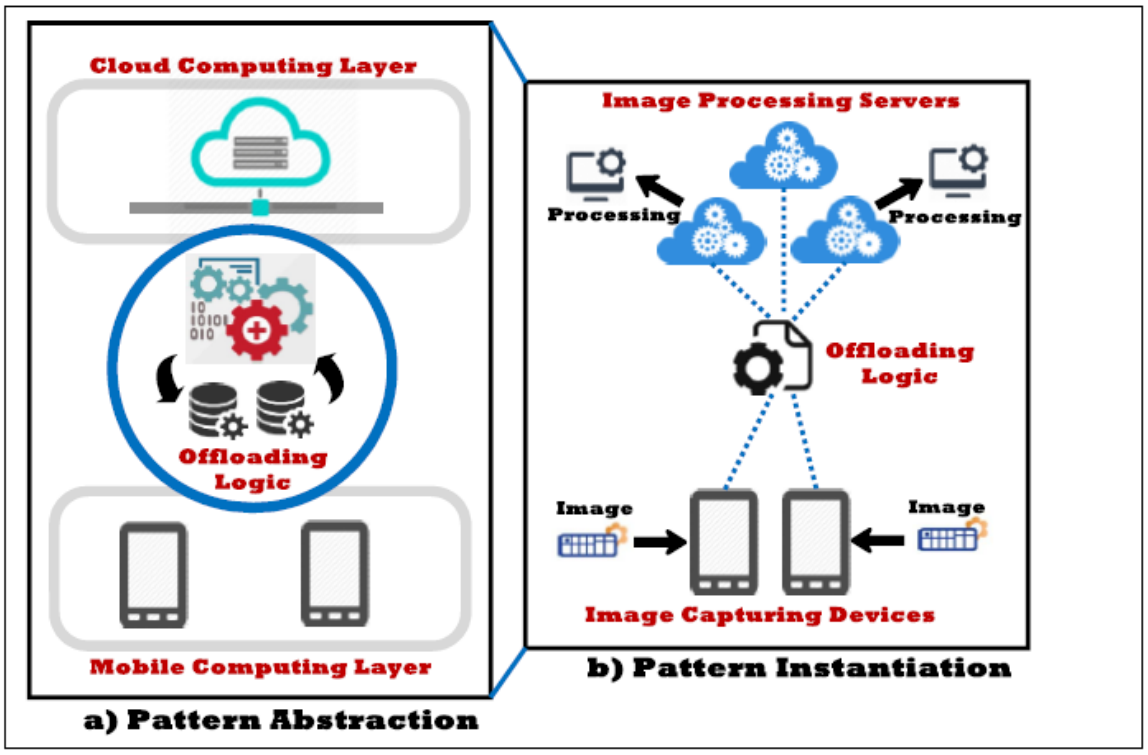


Figure 3. Adaptive Mobile Cloud Offloading Pattern
(Pattern Abstraction and Pattern Instantiation)

Pattern II – Mobile Cloudlets

A) Pattern Intent: To enable a mobile device in a hostile environment to frequently off-load data to servers that are in close proximity of mobile devices that they serve.

B) Design Problem: How to minimise the off-loading latency (on remote server) to a single-hop network while maximising the performance and QoS for mobile computing tasks?

C) Solution: The proposed architecture integrates an intermediate layer (based on localised cloud known as cloudlets) between the enterprise cloud and the mobile device. The solution assumes

that connectivity to the main cloud (enterprise) cloud is either not reliable or commonly un-available.

D) Reference Diagram: This architecture inserts an intermediate layer between the central core (i.e., enterprise cloud) and the mobile devices. At the heart of this architecture is a large centralized core that could be implemented as one of Amazon's data centers or a private enterprise cloud. At the edges of this architecture are offload elements for mobile devices. These elements, or cloudlets, are dispersed and located close to the mobile devices they serve [23]. This architecture decreases latency by using a single-hop network and potentially lowers battery consumption by using WiFi or short-range radio instead of broadband wireless which typically consumes more energy [24] [25].

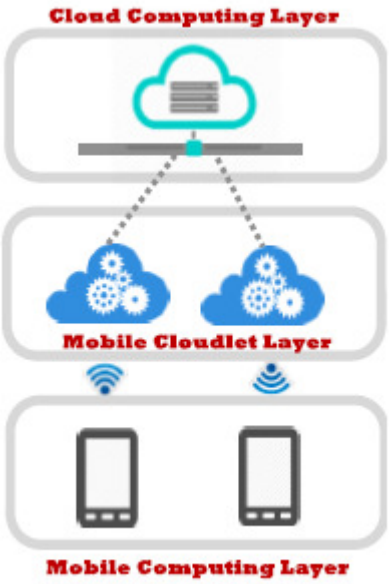


Figure 4. Mobile Cloudlets Pattern

Pattern III – Mobile Sensing and Cloud Analytics

A) Pattern Intent: To exploit the context-sensitive mobile device to capture contextual data that is sent to the cloud-server for data processing and analytics.

B) Design Problem: How to gather the contextual information that can be processed and analysed to perform decisions?

C) Solution: The proposed architecture presents a two layered architecture namely the mobile sensing and cloud analytics layer. Specifically, the mobile layers enables the capturing of the contextual information (e.g.; environmental condition, traffic congestion) and send it to the cloud based server. The cloud server runs the algorithms to analyse data and provide decision support based on processed data.

D) Reference Diagram: This architecture inserts an intermediate layer between the central core (i.e., enterprise cloud) and the mobile devices. At the heart of this architecture is a large centralized core that could be implemented as one of Amazon's data centers or a private

enterprise cloud. At the edges of this architecture are offload elements for mobile devices. These elements, or cloudlets, are dispersed and located close to the mobile devices they serve [23]. This architecture decreases latency by using a single-hop network and potentially lowers battery consumption by using WiFi or short-range radio instead of broadband wireless which typically consumes more energy [24] [25].

E) Reference Diagram

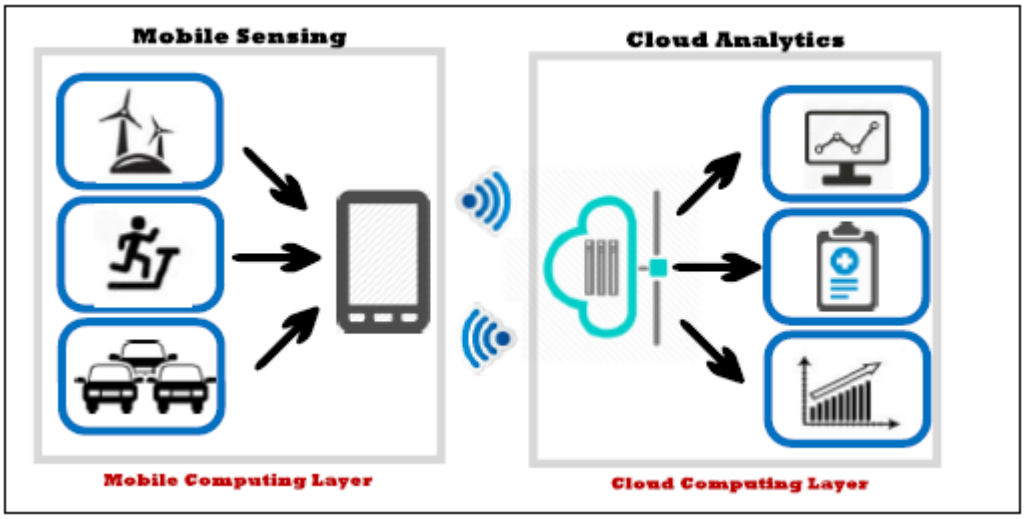


Figure 5. Mobile Sensing and Cloud Analytics

6. THREATS TO VALIDITY AND FUTURE RESEARCH

After presenting the patterns, we now discuss some validity threats and indicate some possible dimensions of the future research.

6.1. Threats to Validity

This paper reports our preliminary work on the discovery and documentation of the patterns to architect mobile cloud software systems. The number of patterns presented here is small and only aims to address and alleviate the resource poverty of the mobile devices by offloading storage and processing intensive data to the cloud-based servers.

(1) Threats to the Applicability of Patterns - We have only demonstrated the patterns and hypothetical scenarios for their application during the architectural design process. However, these patterns must be validated in a real context (case studies and scenarios) for the mobile cloud systems. Case study based validations also correspond to our possible future work. Specifically, we aim to evaluate the increased reusability and decreased efforts with pattern-based architecting. Moreover, research on design and architectural patterns have highlighted pattern-based software design as a continuous and intellectual process [9, 23]. Pattern-based architecting often requires human intervention, and collective decision before applying a specific pattern. Therefore, in addition to other types of validation, the feedback of the practitioner's is also vital to objectively evaluate the applicability and usefulness of the patterns.

(2) Threats to Continuous Discovery of Patterns - Design and architectural patterns are regarded as reusable solutions and best practices that have emerged overtime. For example, the concept of applying reuse-knowledge or best practices to design and develop software emerged from the Gang-Of-Four (GOF) design patterns over more than two decades ago [9]. With now more than two decades of research and practices, design patterns have matured to establish various catalogues and languages as an integral part of modern-day design-to-develop processes [8, 9, 10, 12]. Patterns and best practices cannot be invented, however; they can be (empirically) discovered, overtime with systematic investigation of the pattern sources.

Currently one of the main threats is the limited number of patterns as reported in this paper. However, this threat can be overcome by establish a process and identifying the sources of patterns that can be mined to continuously discover new pattern. Future research demands for a continuous and in an ideal scenario an automated discovery of patterns. Only a longterm approach to discover new patterns and extend the catalogue can prove effective. In comparison to the traditional software systems, mobile cloud systems are new and therefore need innovative patterns to-support their reusability [13, 17].

6.2. Dimensions of Future Research

In the future, our main work is to establish a catalogue of patterns that continuously grows by accommodating new patterns. However, a continuous discovery for new patterns poses two challenges. The first challenge is to identify the sources of patterns that can be empirically investigated to discover reusable knowledge and patterns from them. The second challenge requires the replacement the manual mechanism of pattern discover with a (semi-automated) discovery of the patterns.

Algorithms and Tool Support for Pattern Discovery - In future, we aim to focus on developing algorithms and tools that automate the process of pattern discovery. These algorithms and tools can support automation as well as appropriate user intervention to guided and execute the pattern discovery process. We plan to identify the logs that maintain a history of the steps/activates executed to architect the mobile cloud systems as guided by [11, 18]. The logs can provide the historical data to perform the post-mortem analysis of the architectural design. The algorithms needs to be designed to operate on these logs and to discover new patterns as best and repeatable practices [18].

7. CONCLUSIONS

Mobile cloud architectures rely on mobility and dynamically available software services that needs a recurring need for context-awareness, elasticity and scalability etc. that can be best supported by applying reusable practices and solutions. We proposed a 3-step; pattern-driven architecting process that exploits empirically discovered patterns to guide architectural design for MCC systems. A collection of patterns enhance reusability by abstracting (design primitives). Pattern discovery is a continuous process and provides a systemic approach to investigate emerging design problems and their recurring solutions.

We have highlighted some validity threats that include an objective assessment of the applicability, reusability and efficiency of the patterns based on more complex case studies. In

addition, to support the vision of pattern-based architecting for MCC, new patterns must be continuously discovered and also validated by the pattern users/designers.

REFERENCES

[1] M. Satyanarayanan. Mobile Computing: the Next Decade. ACM SIGMOBILE Mobile Computing and Communications Review. vol 15, no 2, pp: 2-10, 2011.

[2] G. P. Picco, C. Julien, A. L. Murphy, M. Musolesi, G.-C. Roman. Software Engineering for Mobility: Reflecting on the Past, Peering into the Future. In Proceedings of the on Future of Software Engineering (FOSE), pp: 13–28. ACM, 2014.

[3] S. Simanta, K. Ha, G. Lewis, E. Morris, and M. Satyanarayanan. A Reference Architecture for Mobile Code Offload in Hostile Environments. In Fourth International Conference on Mobile Computing, Applications and Services (MobiCASE),), pp: 274–293, 2013.

[4] G. Lewis, S. Simanta, M. Novakouski, G. Cahill, J. Boleng, E. J. Morris, J. Root. Architecture Patterns for Mobile Systems in Resource-Constrained Environments. In Military Communications Conference (MILCOM), pp: 680–685, 2013.

[5] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia. A View of Cloud Computing. In Communications of the ACM (CACM), vol 53, no 4, pp: 50 – 58, 2010.

[6] IBM Developers Work. Mobile Cloud Computing Devices, Trends, Issues, and the Enabling Technologies. [Online:]http://www.ibm.com/developerworks/cloud/library/cl-mobilecloudcomputing/ ,Accessed 02/12/2015.

[7] H.T. Dinh, C. Lee, D. Niyato, P. Wang. A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches. In Wireless Communications and Mobile Computing, vol. 13, no. pp: 1587 – 1611, 2013.

[8] F. Buschmann, K. Henney, D. C. Schmidt. Pattern Oriented Software Architecture vol 5: On Patterns and Pattern Languages. Wiley and Sons, ISBN-13: 978-0471486480, 2007.

[9] C. Pahl, S. Giesecke, W. Hasselbring. Ontology-based Modelling of Architectural Styles. In Information and Software Technology. vol. 51, no. 12, pp: 1739–1749, 2009.

[10] N. B. Harrison, P. Avgeriou, U. Zdun. Using Patterns to Capture Architectural Decisions. In IEEE Software, vol. 24, no. 4, pp: 38-45, 2007.

[11] J. Cámara, P. Correia, R. Lemos, D. Garlan, P. Gomes, B. Schmerl, R. Ventura. Evolving an Adaptive Industrial Software System to Use Architecture-based Self Adaptation. In International Symposium on Software Engineering for Adaptive and Self Managing Systems (SEAMS), pp: 13 – 22, 2013.

[12] J. Kim. Architectural Patterns for Service-based Mobile Applications. In IEEE International Conference on Service-Oriented Computing and Applications (SOCA), pp: 1-4, 2010.

[13] B. Wilder. Cloud Architecture Patterns. O'Reilly Media, Inc, ISBN 10: 1-4493-1977-7, 2012.

[14] C. Fehling, F. Leymann, R. Retter, D. Schumm, W. Schupeck. An Architectural Pattern Language of Cloud-based Applications. In Proceedings of the 18th Conference on Pattern Languages of Programs (PLoP), 2011.

[15] Cloud Computing Design Patterns: [online:] accessed on January 5, 2016. http://www.cloudpatterns.org/

[16] Cloud Design Patterns. [online:] accessed on January 6, 2016. http://en.clouddesignpattern.org/index.php/Main_Page

[17] J. Roth. Patterns of Mobile Interaction. In Personal and Ubiquitous Computing, vol 6, no 4, pp: 282 - 289, 2002.

[18] A. Ahmad, A., P. Jamshidi, C. Pahl. Graph-based Pattern Identification from Architecture Change Logs. In 10th Workshop on Systems/Software Arhitecture, pp. 200-213. Springer. 2012.

[19] G. Lewis, M. Novakouski, and E. Snchez. A Reference Architecture for Group-Context-Aware Mobile Applications. In Mobile Computing, Applications, and Services, volume 110 of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pp: 44–63. Springer, 2013.

[20] Architectural Patterns, Reference Models, and Reference Architectures, [online:] accessed on January 6, 2016. http://www.ece.ubc.ca/~matei/EECE417/BASS/ch02lev1sec3.html

[21] K. Z. Chen. Integration of Design Method Software for Concurrent Engineering using Axiomatic Design. In International Journal of Manufacturing Technology Management, vol 9, no 4, pp. 242–252, 1998

[22] P. Jamshidi, A. Ahmad, C. Pahl. Cloud Migration Research: A Systematic Review. In IEEE Transactions on Cloud Computing (IEEE TCC), vol 1, no. 2, pp: 142 – 157, 2013.

[23] A. Ahmad, P. Jamshidi, C. Pahl. Classification and Comparison of Architecture Evolution Reuse Knowledge – A Systematic Review. In Journal of Software Evolution and Process, vol 26, no 7, pp: 654-691, 2014.m.org/10.1145/161468.16147.

[24] A. Ahmad, A. B. Tamimi, N. Saeed, M. Hamayun, M. Fraz. Research Protocol of Software Architecture for Mobile Cloud Systems: A Mapping Study. Technical Report, College of Computer Science and Engineering, University of Ha'il, 2017. [Online] http://media.

INTENTIONAL BLANK

FRAMEWORK FOR WIRELESS SENSOR NETWORKS CODE GENERATION FROM FORMAL SPECIFICATION

Sara Houhou[1], Laid Kahloul[1], Saber Benharzallah[2] and Roufaida Bettira[1]

[1]LINFI laboratory, Computer Science Department, Biskra University, Algeria
[2]LINFI laboratory, Department of Computer Science, Batna 2 University, Batna, Algeria

ABSTRACT

The development of embedded applications (such as Wireless Sensor Network protocols) often requires a shift to formal specifications. To insure the reliability and the performance of the WSNs, such protocols must be designed following some methods reducing error rate. Formal methods (as Automata, Petri nets, algebra, logics, etc.) were largely used in the specification of these protocols, their analysis and their verification. After that, their implementation is an important phase to deploy, test and use those protocols in real environments. The main objective of the current paper is to formalize the transformation from high-level specification (in Timed Automata) to low-level implementation (in NesC language and TinyOs system) and to automate such transformation. The proposed transformation approach defines a set of rules that allow the passage between these two levels. We implemented our solution and we illustrated the proposed approach on a protocol case study for the "humidity" and "temperature" sensing in WSNs applications.

KEYWORDS

Formal Methods, Timed Automata, Wireless Sensor Network, Code Generation, Automatic Transformation

1. INTRODUCTION

Embedded systems are used in several domains: robotics, smart cars, wireless sensor networks (WSNs), etc. WSNs [1] are composed of tiny embedded devices. They can be defined as a distributed sensors system with an auto-configured infrastructure. To guarantee the reliability and performance in the development of these systems, the use of formal methods represents an ambitious issue. Formal methods allow high-level specification, avoid ambiguity and provide verification techniques. However, the passage from high-level description to the concrete implementation remains an informal step and error prone. The formalisation of this passage and its automation is an attractive research field.

In this paper, we propose a code generator tool that takes as input a formal specification of WSN protocols and generates implementation files in NesC language [2], which is a C dialect language, intended for programming structured component based applications for the TinyOs [2] operating system. admittedly, the approach takes as input a specification written in Timed Automata (TA)

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 35– 52, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71204

[3] provided and verified by UPPAAL [4] model-checker tool and generates a source code written in the NesC [2] language which is the most used in WSNs development. The choice of TA relapsed to time aspect, which is an important characteristic of WSNs. In fact, this paper makes the following contributions: (1) Proposes an extension for timed automata model, (2) Defines a set of transformation rules from TA elements to NesC code, and (3) Implements in Python the generator code tool that takes the TA model as input and generates the corresponding source code.

The reminder of this paper is organised as follows: Section 2 presents related work. Section 3 presents the proposed approach. Section 4 presents the proposed rules and restrictions for the transformation. Section 5 presents the algorithm of the code generation tool. Section 6 presents the application of the proposed approach on a WSN protocol, and finally Section 7 concludes the paper.

2. RELATED WORK

Different approaches to WSN application development can be found in the literature. The work [5] presents a model driven development (MDD) approach for the of WSN applications. The authors define three meta-model at different level of abstraction, with automatic model transformations between them. Moreover, the input of their approach is a domain specific language (DSL) [6]. At first, they define two model-to-model transformation successively which are: DSL model to a subset of UML [7] meta-model for describing a structure of the system and model resulting to NesC meta-model, then, they define model-to-code transformation from NesC meta-model to NesC code. This approach is based on semi-formal specification instead of formal language as proposed in our approach. In [8], the authors present another MDD process to convert a high-level abstract model to a concrete model. The input of their approach is Domain Specific Modelling Language (DSML) [9], and then it is transformed into an executable model TinyDB [10] by using the model transformation rule. The key limitation in their research is that the DSMLs have the capability to describe static applications, meanwhile not to describe mobility and adaptive behaviour of WSNs. In addition, the authors do not conserve the consistency between the models. In [11]–[13], the authors present an approach for developing the DMAMAC protocol proposed in [11]. In [12], the authors design the protocol with an abstract formal model (TA), and then they implement it in NesC language. This approach is based on a manual transformation, thus it may induce lot of errors, and consumes time and efforts.

Recently, several authors [14]–[17] have proposed model driven software engineering (MDSE) approach for WSNs code generation from Coloured Petri Nets (CPNs). Inge *et.al*. [14] describe an automatic code generation from a restricted class of CPNs, annotated with code generation pragmatics, called Pragmatics annotated Coloured Petri nets (PA-CPN) [15] models, to a set of target language (groovy [18], Java [19], Clojure [20], and Python [21]). The tool PetriCode described in [16] shows the implementation of this approach. The authors in [17] used this approach on an industrial sized protocol, which is IETF WebSocket, and they generated its implementation for the Groovy platform. However, the use of pragmatics approach to generate code for any target platform restrict the class of CPN models and make it not suitable for TinyOs. In [22], the authors expanded the PetriCode tool [16] to specific platform model to generate the implementation of sensor network protocols. The input of this approach is Coloured Petri Nets models of protocols, then, the resulting CPN model is given to PetriCode tool, which transformed it automatically to code for (MiXiM [23] simulator, and TinyOs platforms, C++, and NesC respectively. In [24], the authors completed the code generation for the two specific platforms and they tested it in real-world deployment Zolertia Z1 motes. However, this last approach ignores the stochastic nature and real time constraints of WSNs. In [25], another approach based on pragmatics is presented. The approach describes a transformation from CPNs to NesC code

running in TinyOs platform. The construction of CPNs models follows five stages of manual refinement. After the fifth stage, the source code of protocol is generated automatically for the TinyOs platform using a prototype software. The major drawback of this approach is that the refinement steps are specified informally. In addition, the approach is based on manual refinement of models, which may induce errors.

3. APPROACH DESCRIPTION

The main purpose of the paper is the automatic code generation to facilitate the development of WSN applications. The figure 1 represents the global architecture of the proposed approach, illustrating the use of two tools: UPPAAL and the "Code Generator tool". Firstly, the UPPAAL tool is used to model the application and to verify the WSNs model protocols. Secondly, the "Code Generator tool" which represents the implementation of our transformation approach. The input of the approach is a formal protocol description (i.e., a network of Timed Automata) and a set of modelling restrictions. The timed automata network is specified using UPPAAL tool and the designer of the system must respect the modelling restrictions (introduced in this paper) to assist the generation of a complete code. These restrictions oblige the designer to add specific keywords on timed automata (as procedures and synchronization variables). These later represent functional information linked with patterns of the target language. After the modelling, we check the properties to be verified, if they are satisfied, the "Code Generator tool" takes as input the resulted and verified model and transforms it into an implementation by applying the proposed mapping rules. The result of the mapping phase is an instance of the implementation. The generated implementation represents a large part of the real code of the protocol but it needs some additions to be emulated in real hardware or simulated in a simulator tools as TOSSIM [26].

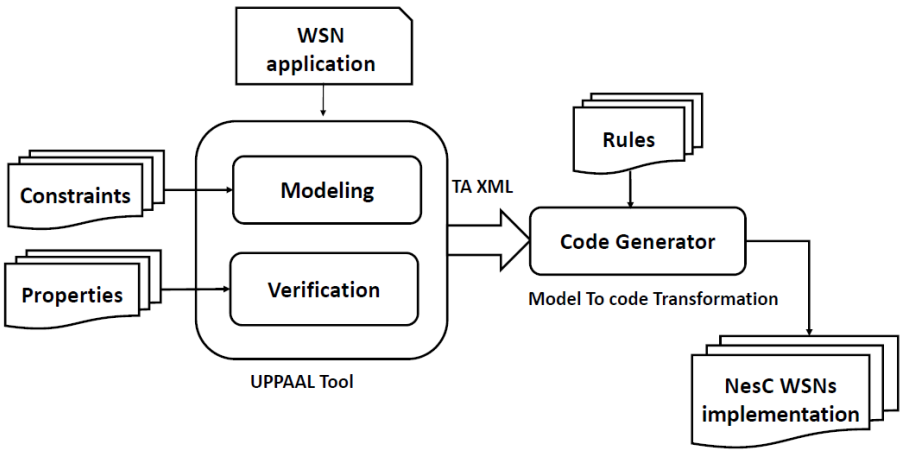


Figure 1: Approach Overview.

4. CODE DERIVATION

In this section, we present the basic step of our approach, which transforms high-level specifications written in TA to a NesC source code of a WSN protocol. In order to generate a source code using our tool, we apply the following informal restrictions to transform a given timed automata to a NesC code. These restrictions refer to the generic patterns that TinyOs defines (the use of hardware as Radio and LEDs, the software as the clock components, the sensing components, and the message communication). We quote these restrictions in the table 1:

Table 1: Restrictions.

Number	Restrictions
1	the user must define explicitly the timed automata model used to model the environment;
2	the user must use the procedure PreparePacket() to define the packet structure;
3	the user must use the procedure LediToggle to turn on and off LEDs;
4	the user must use the procedure LediOn to turn on LEDs;
5	the user must use the procedure LediOff to turn off LEDs;
6	the user must use the procedure Ack() to enforce acknowledgements for the transmitted messages in the protocol;
7	the user must use the procedure TempRead() to sense temperature measure from the environment;
8	the user must use the procedure HumidRead() to sense humidity measure from the environment;
9	the user must use the procedure LightRead() to sense light measure from the environment;
10	the user must use the procedure VoltRead() to sense the voltage measure from the environment;
11	the user must use the variable isSucceed to check whether the previous sensing procedures are executed successfully;
12	The user must use the synchronisation channel Comm to model the sending and receiving radio messages.

The generation of protocol implementation uses a set of rules that we have defined to map each TA element to a portion of source code. These rules are presented in the following items:

- **R1**. Each set of templates in the TA model will be used to generate a WSN application.
- **R2**. Each template is mapped to a ***module*** component and a ***configuration*** component unless if this TA is added for only simulation purposes (i.e., humans, environment, pre-exiting systems...etc.).
- **R3.** Each module component of a template uses the ***Boot*** interface and implements the initial point of each NesC program to be executed automatically by the processor, which is the "***Booted"*** event. The Boot event is signalled when the system has booted successfully.

- **R4.** Clocks in the TA trigger the use of an instance of the ***TimerC*** component. The interface ***Timer*** provided by this later is used in the module file. The application starts firstly the Timer by calling the command ***startPeriodic*** for many times or ***startOneShot*** for one single time. Such commands are often implemented in the booted event that we have specified in the previous rule. The clock variables values represent the Timer period. As a consequence of using the interface Timer, we should define the fired event, which is signalled when the Timer expires. In the configuration file, a component must be added for wiring the Timer interface to an instance of the ***TimerMilliC*** component.

- **R5.** The use of keywords: (***LediToggle (), LediOn (), and LediOff ()*** where i = 0...2 according to the manipulated LED) in TA. These keywords are mapped to: the use of the ***Leds*** interface in the module file, a call of ***the LediToggle***() (resp. ***LediOn()***, and ***LediOff()***) command, finally, the declaration of the component ***LedC*** in the configuration file, which provides the ***Leds*** interface and wires it to the module which uses this interface.

- **R6**. The use of the keyword ***Ac***k () in the TA is mapped to: the use of the ***PacketAcknowledgements*** interface in the module file. To call the ***RequestAck*** command defined in the ***PacketAcknowledgements*** interface before sending messages, to call the ***WasAcked*** command defined in the ***PacketAcknowledgements*** interface to ensure the arrival of the ***acknowledgement***, and to declare the component ***ActiveMessageC*** in configuration file, which provides the interface ***PacketAcknowledgements*** and wires it to the module which uses this interface.

- **R7.** The use of the keyword ***preparePacket(arg)*** in the TA is mapped to the: use the ***Packet*** interface in the module file, to define a structure of message in a header file, to import this later in the module, and to declare a function ***preparePacket(typedef arg)*** in the concerned module.

- **R8**. The use of the keyword ***comm*** refers to a communication, which involves : the manipulation of the radio and the use of the ***SplitContro***l interface in the module, the call to the start command for starting the radio in the booted event, and the implementation of the ***startDone()*** (resp. ***stopDone()***) events, which are defined in the ***SplitControl*** interface. In addition, the declaration of the component ***ActiveMessageC*** in the configuration file which provides the interface ***SplitControl*** and wires it to the module, which uses this interface.

- **R9**. As the communication contains two parts, the transformation will be as follows:

  - The sender part ***comm!*** is mapped to the: use of the ***AMsend*** interface, to call the function ***preparePacket(arg)***, to call the ***send()*** command in the module file, to implement the ***sendDone()*** event in the module, and to declare the component ***AMSenderC(AM RADIO)*** in the configuration file, which provides the interface ***AMSend*** and wires it to the module using this interface.

  - The receiver part ***comm?*** is mapped to the: use of the ***Receive*** interface, to implement the ***receive*** event. Then, to declare the component ***ReceiveC*** in the configuration file, which provides the interface ***Receive*** and wires it to the module, which uses this interface.

- **R10**. The use of the keyword ***TempRead ()*** in the TA are mapped to the : use the ***Read ()*** interface as ***TempRead***, to call the ***Read*** command in the module file, to implement the ***ReadDone ()*** event in the module, and to declare an instance of the component

SensirionSht11C () in configuration file. This component provides the interface temperature and wires it to the module, which uses the ***TempRead*** interface.

- **R11.** The use of the keyword ***HumidRead ()*** in the TA are mapped to: the use the ***Read ()*** interface as ***HumidRead***, to call the ***Read*** command in the module file, to implement the ***ReadDone ()*** event in the module, and to declare an instance of the component ***SensirionSht11C ()*** in configuration file. This component provides the ***humidity*** interface and wires it to the module, which uses the ***HumidRead*** interface.

- **R12**. The use of the keyword ***LightRead ()*** in the TA is mapped to: the use of the ***Read ()*** interface as ***LightRead,*** to call the ***Read*** command in the module file, to implement the ***ReadDone ()*** event in the module, and to declare an instance of the component ***HamamatsuS10871TsrC ()*** in configuration file, then wires it to the module which uses the ***LightRead*** interface.

- **R13**. The use of the keyword ***VoltRead ()*** in the TA is mapped to: the use of the ***Read ()*** interface as ***VoltRead***, to call the ***Read*** command in the module file, to implement the ***ReadDone ()*** event in the module, and to declare an instance of the component ***VoltageC ()*** in configuration file, then wires it to the module which uses the ***VoltRead*** interface.

- **R14.** Variables declared in a particular template are mapped to a declaration of variables in the module, which implements this template.

- **R15.** If the guard is built upon non-clock variables then it is mapped to a standard conditional branching (an If or If-Else block) in the implementation of the module file, according to the state source of the transition. If the guard is upon clock variables then it is mapped to arguments used in the starting commands of the timer. When using the variable isSucceeded, a particular if-else block will be generated.

5. CODE GENERATOR TOOL

In this subsection, we present the implemented tool that enables the use of formal model for a semi-automated code generation approach. As presented in Figure.2, the transformation can be applied on a given formal specification in timed automata to nesC language. In that case, the designer can design the TA using the UPPAAL tool [4]. UPPAAL is a model checker supporting the timed automata modelling. UPPAAL is based on timed automata concepts, which are a set of clocks, channels for the synchronized systems (automata), variables and additional elements. Each automaton has an initial state. The synchronization between the different automata can take place using channels. A channel may be an output channel (denoted by channel-name!), or an input channel (denoted by channel-name?). After the design of the model, the designer can use UPPAAL also to check proprieties, verify and validate the model. When the model is validated, the designer can import it to the implemented Code Generator tool in-order to generate the corresponding code implementation (See the Figure.3).

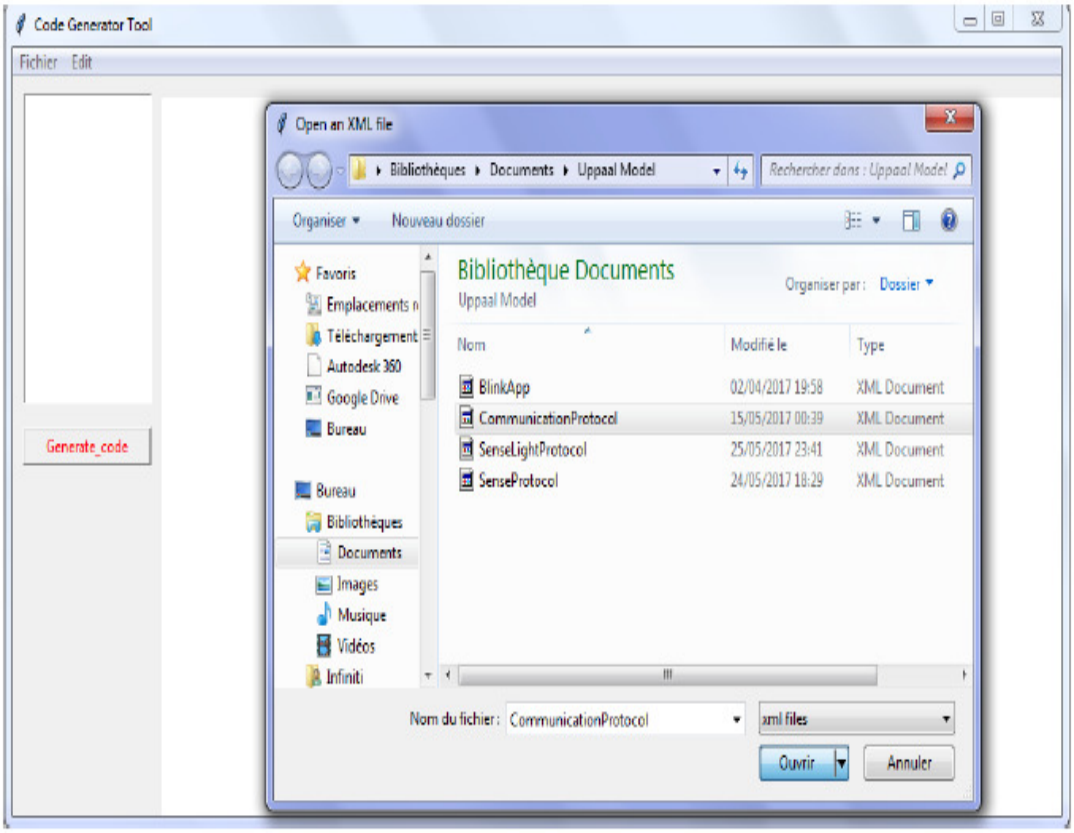


Figure 2: Generator tool interface (Importing XML Files from UPPAAL).

The implementation of the code generator framework and the translation library are written in Python language [21]. Python is an intelligent untyped programming language that lets you work more quickly and integrate your systems more effectively. It is an open source language, and it is available for several operating systems. Firstly, we use the PyXB Python library for parsing XML files of timed automata. XML [27] is used as the standard format to describe models, formats, and data types. The modelled specification in UPPAAL is XML files that can be opened and visualized graphically as template models. Secondly, we have developed a translation from the information resulting from the parsing step to NesC source code by implementing the transformation rules. The following algorithms present the main procedures used in the implementation of our generator tool. The Algorithm 1 introduces the procedures used to map the formal model specification to NesC code. In this algorithm, we follow the same logic of steps presented in Fig. 2. First, each element in the XML file (representing the formal specification) is statically parsed to identify the set of operations and variables declaration. The functional behaviour (operations) can be identified from global declaration part of the XML file and from the template part. Then, as represented in the procedure 1 and the procedure 2, for each checked operation, we apply the corresponding rule. At the same time, the generated files are filled.

Algorithm 1 Code Generator algorithm.

```
Data: XML file
Result: NesC implementation
Var : balise : String
while not at end of the XML file do
    Get(file_Content)
    Read content
    if balise=="Declaration" then
        ParsingDecalaration()
    end
    if balise=="Template" then
        Aply_rules(R2&R3)
        ParsingTemplate()
    end
    Apply_rule(02)
end
```

Procedure 1 ParsingDeclaration.

```
Data: Declaration content
Result: source Code
Var : chan_nam,variable_type, text : String ;
while text not null do
    Read content
    if variable_type= int then
        Add( file.h , "uint8_t " , var_name )
    end
    if variable_type= String then
        Add( file.h , "String " , var_name )
    end
    if variable_type= chan or variable_type= broadcast chan then
        if chan_name= comm then
            Apply_rule(R8&R9)
        end
        if chan_name= comm? then
            Apply_rule(R8&R9)
        end
    end
end
```

Procedure2 ParsingTemplate.

```
Var : balise_child, text : String ;
while not at end of the template do
    text== Get(Template_Content)
    if (text not null) then
        Apply_rule(R13) if ( balise_child == Declaration ) then
            if (( variable_type = int ) and ( variable_name= isSucceed )) then
                Apply_rule(R14) ;
            end
            if ( variable_type= clock ) then
                Apply_rule(R4) ;
            end
            if (declared_procedure ) then
                if ( declared_procedure= preparePacket( )) then
                    Apply_rule(R7);
                end
                if ( declared_procedure = Toggle( ) )or (declared procedure=
                lediOn()) or (declared procedure= lediOff())) then
                    Apply_rule(R5) ;
                end
                if (declared_procedure= TempRead ( )) or (
                declared_procedure= HumidRead ( )) then
                    Apply_rule(R10) ;
                end
                if (declared_procedure= LightRead( )) then
                    Apply_rule(R11) ;
                end
                if (declared_procedure= VoltRead( )) then
                    Apply_rule(R12) ;
                end
            end
        end
        if (balise_child == Transition) then
            Kind==Get_Label_Kind() if (Kind=="Guard") then
                Check_Source_transition()
                Apply_rule(R14)
            end
        end
    end
end
```

6. APPLICATION ON A CASE STUDY

Our example is an illustrative system that offers to the user the opportunity to measure the humidity and the temperature from the environment. This system presents an Air temperature and Humidity monitoring used in temperature-controlled rooms. This system contains all the elements

of sensor network application such as communication (sending and receiving), sensing, and processing packets. The system describes the communication between two Telosb nodes in a sensor network. The system uses two clocks. An alarm system should be linked to the monitoring system, which presented in our case by the use of two LEDs. The system manages the sending and receiving cycle using two clocks. The two nodes are specified as Sense_Sender node, which has two types of sensing (temperature and humidity), and the receiver node. The communication between these two nodes is based on the temperature, and humidity measurements. Each time c1 (2000 milliseconds), the Sense_Sender node detects the humidity and the temperature values from the environment and uses these values to calculate the humidity (resp. the temperature). If any of these values are upper a certain threshold, the corresponding LED turns on. After 4000 milliseconds for the second clock, the receiver board will light up.

6.1. Timed Automata Models of the System

We modelled the example with three timed automata. The first and second automaton represent the behaviour of the temperature (resp. humidity) sensing operations as shown in the Figure 4 (resp. 5). The third TA depicted in Figure 6 represents the receiver.

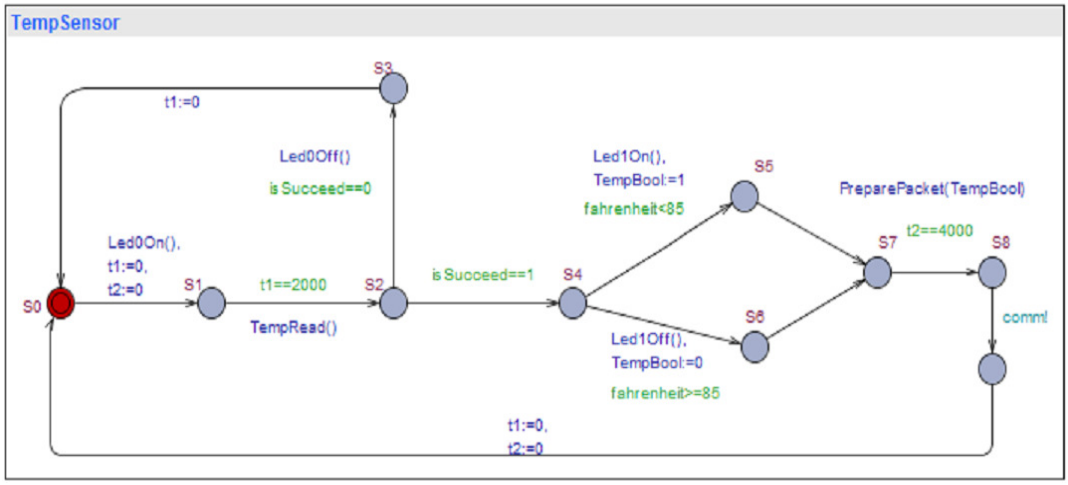


Figure 3: TA Model for Temperature Sensing.

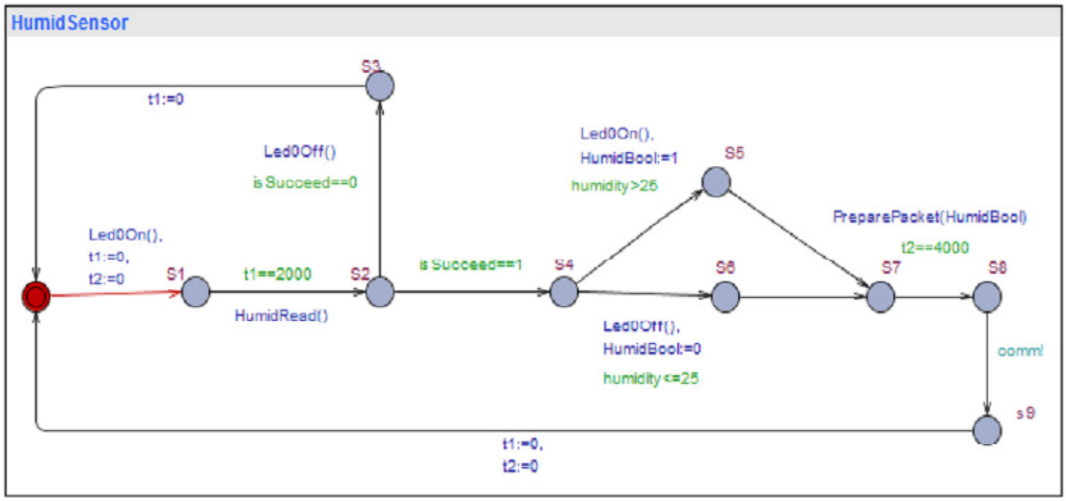


Figure 4: TA Model for Humidity Sensing.

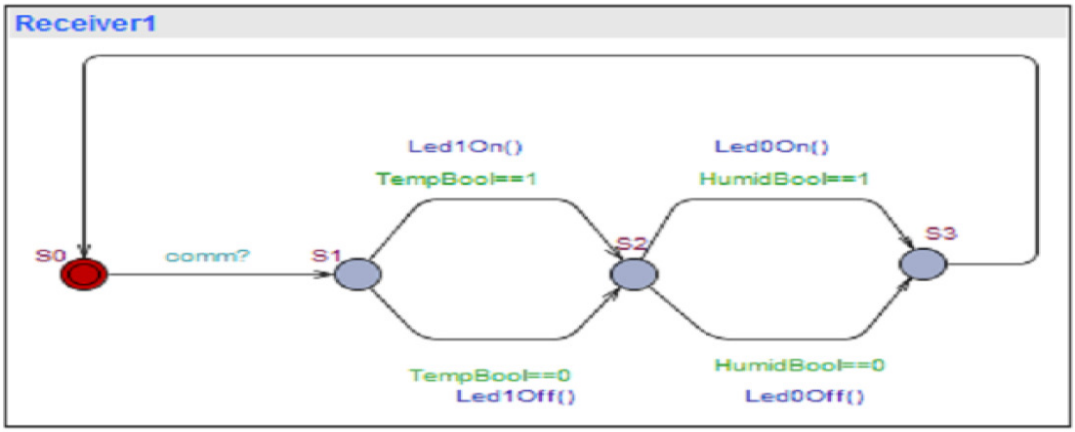


Figure 5: TA Model for the Receiver part.

6.2. Timed Automata Models of the System

In this subsection, we present the principal files, which contain the implementation of the given model. The code presented in the following listings (see listings 1, 2) show the generated code by our code generator. Due to space reason, we have explained a part of the generated code from the model presented in 6.1. The Figure.4 presents the timed automata of the temperature sensing. Firstly, the code generator applies the rules R1 and R2 to create a module and a configuration files. Then, it applies the rule R3 to implement the first event to start the program execution. The figure.3 shows a started transition (S0, S1) labelled by two resets (t1=:0, t2:=0) of the clocks t1, t2 and one procedure "Led0On()". During the parsing of the model, the appearance of a clock in a guard or a reset on a transition triggers the application of the rule R4 that consists to a call for the Timer component. The procedure call "Led0On()" on the transition (S0,S1) is mapped to the use of the LEDs component in the code, using the rule R5. The listings 1, 2 present the module and the configuration files. The transition (S1, S2) is labelled by the guard (t1==2000) and the procedure call (TempRead()). During the parsing of the model, the guard upon the clock t1 is used to define the argument value used in a starting time command as mentioned in the rule R14. The call of the procedure TempRead() is mapped to the use of the "temperature sensing component" as described in the rule R10. In addition, if the procedure TempRead defines and uses variables, the code generator applies the rule R13. The following listings 1 and 2 show the generated code corresponding to the part of the model linking state S0 to state S2. We put the complete generated code in the appendix A.

```
configuration SenseC{
}
implementation{
        components MainC, LedsC;
        components TempM;
        components new TimerMilliC() as T1;
        components new TimerMilliC() as T2;
        TempM.Boot -> MainC;
        TempM.Leds -> LedsC;
        TempM.T1 -> T1;
        TempM.T2 -> T2;
}
```

Listing 1: Skeleton of the configuration file.

```
#include "packets.h"
// application of the rule R1
module TempM{
        uses {
                // application of the rule R3
                interface Boot;
                // application of the rule R5
                interface Leds;
                // application of the rule R4
                interface Timer<TMilli> as T1;
                interface Timer<TMilli> as T2;
        }
}
implementation{
        // application of the rule R3
        event void Boot.booted(){
                // application of the rule R4
                call T1.startPeriodic(...);
                call T2.startPeriodic(...);
                // application of the rule R5
                Leds.led0On;
        }
   // application of the rule R4
        event void T1.fired(){
                                }

        event void T2.fired(){

                                }
}
```

Listing 2: Skeleton of the module file.

7. CONCLUSION AND PERSPECTIVES

Wireless sensor networks are of considerable interest and a new stage in the evolution of information and communication technologies. This new technology attracts increasing interest because of the diversity of its applications: health, environment, industry and even sports. In the current paper, an automated transformation approach was presented which intended to generate source code of specific target language (NesC programming language) from high-level formal specification (Timed Automata model). This approach is divided into two parts: the first part is dedicated to the modelling of the WSNs system respecting specific constraints model, and a second part dedicated to code generation. In fact, the proposed approach is intuitive, it defines a set of transformation rules that a generator code takes as input as well as the XML description of a TA specification, and then it generates the source code. In order to automate this transformation, we have implemented these rules in a framework. In addition, we have demonstrated the application of the approach on a case study of air monitoring system. During the realisation of the current work, we have faced several difficulties in the definition of rules. This is due to the high level abstraction of TA whereas NesC is closer to hardware devices. Therefore, it is not evident to extract enough information from TA, that help in the implementation of a WSN system. The current work has defined a set of not complete rules. These rules can be used to generate code for several case studies. The work will be improved by completing the set of rules that can be used to generate code for more complex formal models and to consider other aspect of the formal model (i.e., probabilistic parameters).

REFERENCES

[1] Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer networks*, *52*(12), 2292-2330.

[2] Levis, P., & Gay, D. (2009). *TinyOS programming*. Cambridge University Press.

[3] Bengtsson, J., & Yi, W. (2004). Timed automata: Semantics, algorithms and tools. *Lecture Notes in Computer Science*, *3098*, 87-124.

[4] J. B. Gerd Behrmann, Tobias Amnell, (2015). "UPPAAL", URL: http://www.uppaal.org/. [accessed: 2017-05-08].

[5] Losilla, F., Vicente-Chicote, C., Álvarez, B., Iborra, A., & Sánchez, P. (2007, September). Wireless sensor network application development: An architecture-centric mde approach. In European Conference on Software Architecture (pp. 179-194). Springer, Berlin, Heidelberg.

[6] D. Demange, (2006), "Domain specific language", URL: http://igm.univmlv.fr/~dr/XPOSE2006 /LOPDEMANGE/dsl.html [Accessed: 2017-07-14].

[7] O. M. Group, (2017), "Unified modeling language", URL: http://www.uml.org/ [Accessed: 2017-03-15].

[8] Shimizu, R., Tei, K., Fukazawa, Y., & Honiden, S. (2011, May). Model driven development for rapid prototyping and optimization of wireless sensor network applications. In Proceedings of the 2nd Workshop on Software Engineering for Sensor Network Applications (pp. 31-36). ACM.

[9] S. I. J. R. B. S. Grady Booch, Alan Brown, (2004), "Domain-specific modelling", URL: http:// www.dsmforum.org/ [accessed: 2017-05-15].

[10] Madden, S. R., Franklin, M. J., Heller stein, J. M., & Hong, W. (2005). TinyDB: an acquisitional query processing system for sensor networks. ACM Transactions on database systems (TODS), 30(1), 122-173.

[11] Somappa, A. A. K., Øvsthus, K., & Kristensen, L. M. (2014). Towards a dual-mode adaptive MAC protocol (DMA-MAC) for feedback-based networked control systems. Procedia Computer Science, 34, 505-510.

[12] Somappa, A. A. K., Prinz, A., & Kristensen, L. M. (2015). Model-Based Verification of the DMAMAC Protocol for Real-time Process Control. In VECoS (pp. 81-96).

[13] Somappa, A. A. K., Øvsthus, K., & Kristensen, L. M. (2016). Implementation and Deployment Evaluation of the DMAMAC Protocol for Wireless Sensor Actuator Networks. Procedia Computer Science, 83, 329-336.

[14] Simonsen, K. I. F., Kristensen, L. M., & Kindler, E. (2013, September). Generating protocol software from cpn models annotated with pragmatics. In Brazilian Symposium on Formal Methods (pp. 227-242). Springer, Berlin, Heidelberg.

[15] Simonsen, K. I. F., Kristensen, L. M., & Kindler, E. (2016). Pragmatics annotated coloured petri nets for protocol software generation and verification. In Transactions on Petri Nets and Other Models of Concurrency XI (pp. 1-27). Springer Berlin Heidelberg.

[16] Simonsen, K. I. F. (2013, September). PetriCode: a tool for template-based code generation from CPN models. In International Conference on Software Engineering and Formal Methods (pp. 151-163). Springer, Cham.

[17] Simonsen, K. I. F., & Kristensen, L. M. (2014, June). Implementing the websocket protocol based on formal modelling and automated code generation. In IFIP International Conference on Distributed Applications and Interoperable Systems (pp. 104-118). Springer, Berlin, Heidelberg.

[18] The Apache Groovy project, (2003-2017), "Apache groovy". [Online]. Available: http: //www. groovy-lang.org/

[19] Oracle, (2003-2017), "Java", URL : https://www.java.com/fr/ [Accessed : 2017-07-20].

[20] R. Hickey, (2017), "The clojure programming language", URL: https://clojure.org/ [accessed: 2017-02-20].

[21] Python, J. (2009). Python (programming language). Python (programming Language) 1 CPython 13 Python Software Foundation 15, 1.

[22] Kumar, S. A., & Simonsen, K. I. (2014, April). Towards a model-based development approach for wireless sensor-actuator network protocols. In Proceedings of the 4th ACM SIGBED International Workshop on Design, Modeling, and Evaluation of Cyber-Physical Systems (pp. 35-39). ACM.

[23] A. Viklund, (2011-2017), "Mixim", URL: http://mixim.sourceforge.net/ [accessed: 2017-07-20].

[24] Somappa, A. A. K., & Simonsen, K. I. F. (2016). Model-based Development for MAC Protocols in Industrial Wireless Sensor Networks. In PNSE@ Petri Nets (pp. 193-212).

[25] Kristensen, L. M., & Veiset, V. (2016, June). Transforming CPN Models into Code for TinyOS: A Case Study of the RPL Protocol. In International Conference on Applications and Theory of Petri Nets and Concurrency (pp. 135-154). Springer International Publishing.

[26] Levis, P., Lee, N., Welsh, M., & Culler, D. (2003, November). TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In Proceedings of the 1st international conference on Embedded networked sensor systems (pp. 126-137). ACM.

[27] Tim Bray, Jean Paoli, C Michael Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (xml). World Wide Web Journal, 2(4):27–66, 1997.

APPENDIX A

```
#ifndef STRUCT_MSG_H
#define STRUCT_MSG_H
   typedef nx_struct my_msg_t {
       nx_uint16_t msg_type;
       nx_uint16_t msg_id;
       typedef msg_value;
       } my_msg_t;
#define REQ 1
#define RESP 2
   enum { AM_RADIO =6 };
#endif
```

Listing 3: Skeleton of the Packet Structure.

```
configuration SenseC{
}
implementation{
       components MainC, LedsC;
       components TempM;
       components ActiveMessageC;
       components new AMSenderC(AM_RADIO);
       components new TimerMilliC() as TempTimer;
       components new TimerMilliC() as NetworkTimer;
       components SerialPrintfC;
       components new SensirionSht11C() as TempRead;
       TempM.Boot -> MainC;
       TempM.Leds -> LedsC;
       TempM.T1 -> TempTimer;
       TempM.T2 -> NetworkTimer;
       TempM.TempRead -> TempRead.Temperature;
       TempM.Packet -> AMSenderC;
       TempM.AMSend -> AMSenderC;
       TempM.AMControl -> ActiveMessageC;
}
```

Listing 4: Skeleton of the Temperature Configuration

```
#include "packets.h"
module TempM{
        uses {
                interface Boot;
                interface Leds;
                interface Timer<TMilli> as T1;
                interface Timer<TMilli> as T2;
                interface Read<uint16_t> as TempRead;
                interface Packet;
                interface AMSend;
                interface SplitControl;
        }
}
implementation{
        uint16_t centigrade, fahrenheit;
        uint8_t tempBool;
        message_t _packet;

        event void Boot.booted(){
                call T1.startPeriodic(2000);
                call T2.startPeriodic(4000);
                call SplitControl.start();
                Leds.led0On;
        }
        event void SplitControl.startDone(error_t error) {
                if (error == SUCCESS) {
                        call Leds.led0On();
                }
                else {
                        call SplitControl.start();
                }
        }
        event void SplitControl.stopDone(error_t error) {
                                }
        event void T1.fired(){
                if (!(call TempRead.read() == SUCCESS))
                   call Leds.led0Off();
        }
        //construct the packet:
        void PreparePacket(uint16_t val){
        //obtain the packet pointer
            my_msg_t* msg = call Packet.getPayload(& _packet, sizeof(my_msg_t));
        // Specify the sequence number of the message
                msg->msg_id= TOS_NODE_ID;
        //affect the sensed information to the message value
                msg->msg_value=tempBool;
        }
        event void T2.fired(){
           PreparePacket(tempBool);
           call AMSend.send(AM_BROADCAST_ADDR, &_packet, sizeof(my_msg_t));
                                        }
        event void AMSend.sendDone(message_t *msg, error_t error) {
        ...
        }
        event void TempRead.readDone(error_t result, uint16_t val){
                        centigrade = -39.6 + .01*val;
                        fahrenheit = (9/5)*centigrade + 32;
                        if (fahrenheit > 85){
                                call Leds.led1On();
                                tempBool = 1;
                        } else {
                                call Leds.led1Off();
                                tempBool = 0;
                        }}}
```

Listing 5: Skeleton of Temperature Sensing Module

```
#include "MsgStruct.h"
module ReceiveC{
  uses {
       interface Boot;
       interface Leds;
       interface Packet;
       interface AMPacket;
       interface SplitControl as AMControl;
       interface Receive;
       }
}
implementation{
  uint8_t tempBool;
  uint8_t humidBool;
  event void Boot.booted(){
       call SplitControl.start();
       }
  event void SplitControl.startDone(error_t error) {
       if (!(error == SUCCESS))
              call SplitControl.start();
    }

  event void SplitControl.stopDone(error_t error) {
       }
  event message_t * Receive.receive(message_t *msg,void *payload,uint8_t
len) {
              if (len == sizeof(my_msg_t)) {
                     my_msg_t * incomingpacket = (my_msg_t*) payload;
                     tempBool = incomingpacket->tempBool;
                     humidBool = incomingpacket->humidBool;
                     if (humidBool == 1) {
                            call Leds.led0On();
                            }
                     else {
                            call Leds.led0Off();
                            }
                     if (tempBool == 1) {
                            call Leds.led1On();
                     }
                     else {
                            call Leds.led1Off();
                            }
              }
              return msg;
       }
}
```

Listing 6: Skeleton of Receiver Module.

```
#include "packets.h"

module HumidityM{
        uses {
                interface Boot;
                interface Leds;
                interface Timer<TMilli> as T1;
                interface Timer<TMilli> as T2;
                interface Read<uint16_t> as HumidRead;
                interface Packet;
                interface AMSend;
                interface SplitControl;
        }          }
implementation{
        //variables declaration
        uint16_t humidity;
    uint8_t HumidBool;
        message_t _packet;
        event void Boot.booted(){
                call T1.startPeriodic(2000);
                call T2.startPeriodic(4000);
                call SplitControl.start();
                Leds.led0On;
        }
        event void SplitControl.startDone(error_t error) {
                if (error == SUCCESS) {
                        call Leds.led0On();
                }
                else {
                        call SplitControl.start();
                }}
        event void SplitControl.stopDone(error_t error) {
        }
        //start the system, radio..
        event void TempTimer.fired(){
                if (!(call HumidRead.read() == SUCCESS))
                        call Leds.led0Off();
        }
         //construct the packet:
        void PreparePacket(uint16_t val){
        //obtain the packet pointer
            my_msg_t* msg = call Packet.getPayload(& _packet, sizeof(my_msg_t));
        // Specify the sequence number of the message
                msg->msg_id= TOS_NODE_ID;
        //affect the sensed information to the message value
                msg->msg_value=HumidBool;
        }
        event void T2.fired(){
        //prepare packet
                PreparePacket(HumidBool);
        //send the packet
       call AMSend.send(AM_BROADCAST_ADDR, &_packet, sizeof(my_msg_t));
        }
        ......
        event void HumidRead.readDone(error_t result, uint16_t val){
           humidity = -4.0 + 0.0405*val + (-2.8 * pow(10.0,-6))*(pow(val,2));
                        if (humidity > 25){
                                call Leds.led0On();
                                humidBool = 1;
                        } else {
                                call Leds.led0Off();
                                humidBool = 0;
                        } }

}
```

Listing 7: Skeleton of Humidity Sensor.

```
configuration SenseC{
}
implementation{
       components MainC, LedsC;
       components TempM;
       components ActiveMessageC;
       components new AMSenderC(AM_RADIO);
       components new TimerMilliC() as TempTimer;
       components new TimerMilliC() as NetworkTimer;
       components SerialPrintfC;
       components new SensirionSht11C() as HumidRead;

       TempM.Boot -> MainC;
       TempM.Leds -> LedsC;
       TempM.T1 -> TempTimer;
       TempM.T2 -> NetworkTimer;
       TempM.HumidRead ->  HumidRead.Humidity;
       TempM.Packet -> AMSenderC;
       TempM.AMSend -> AMSenderC;
       TempM.AMControl -> ActiveMessageC;
}
```

Listing 8 : Skeleton of Humidity Configuration

IMPLEMENTATION OF A SMART HOUSE APPLICATION USING WIRELESS SENSOR NETWORKS

Ismail MOHAMMED[1] and Erkan DUMAN[2]

[1]P.G. Student, Department of Information Technology, Technical College of Informatics, Sulaimani Polytechnic University, Sulaimani, Iraq
[2]Asst. Professor, Department of Computer Engineering, Faculty of Engineering, Firat University, 23119 Elazığ, Turkey

ABSTRACT

Digitisation and automation are becoming increasingly prevalent in our daily lives, both inside and outside the house. They are all attempts to simplify, use easily, monitor and be aware of any devices in a house that are connected to the system. These automated systems can be an alternative to other home manual settings. Smart homes are rarely found in my country, iraq. People's unfamiliarity with them, the lack of use, the costs and a failure to see the importance of such systems are all possible explanations for this. Nowadays, it is generally advancing very rapidly, the internet and smart phones in particular. Using such technology and devices is inevitable and difficult to avoid. Such smart systems are able to indicate and control light, temperature, dew point, gas flow, fire ignition, opening doors and buzzing alarms. Now, as iot is emerging more widely than before, it is expected that IOT will become more prevalent and occupy wider areas of the world. IOT enables a better connection among devices in a smart home or to other smart systems via the internet and WSN. In this paper, a smart home via smart phones and pcs will be explained. Controlling and indicating electrical and electronic devices in houses and buildings will also be presented.

KEYWORDS

Smart home, IoT, Arduino Mega2560 and UNOR3, Ethernet Shield 5100, WSNs, sensors, GSM, Wi-Fi ESP8266-12E, 01, Solar Tracker.

1. INTRODUCTION

A smart home is an automated and intelligent home which can be designed and set due to the technology being used [1]. Smart homes can be programmed to fit the client's needs. Programming can lead to a fully automated system where every device in the home can communicate with others via sensors. A smart home can provide good security, convenience, entertainment, good communication, economy and an information system [2].

Different smart house systems have been created where the control is through Android applications, Bluetooth, WIFI, web, call and/or short message administrations (SMS) based

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 53– 70, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71205

through GSM808. WIFI capability is great and the vast majority of current devices can be connected easily and co-ordinated so as to reduce the system's costs. Bluetooth's range, however, limits its usage to inside the home [3].

Several methods can be used to secure a smart home, such as: RFID, Password or fingerprint. These smart homes serve well in pre-alarming and controlling the temperature, air-conditioning, bulbs equipped with sensors, gas detecting sensors and flame sensors. As well as all this, a smart home alerts the home owner to thieves and burglars, several unwanted conditions with the use of IR waves. Furthermore, smart home sensors take advantage of different ranges of electromagnetic waves including GSM, IR, Bluetooth and Wi-Fi. With these different frequencies a home's devices are connected in a Wireless Home Area Network (WHAN)[4]

It is also important in this project to alert homeowners to any unwanted events and intruders who might try to enter the house without legal permission. The alert can be given via a ring to a phone, an SMS or an email. More interestingly, the system can be set to alert governmental and national bodies about unwanted accidents in cases when the owner of the house might be travelling to another country. In this way, fire fighters can reach the home and tackle a fire even without the help of the homeowner. Another aspect of this project is to involve the smart house in the upcoming era of the Internet of Things (IoT). Thus, different sections of the house can be accessed via a phone, tablet or PCs [5]. The programs of the system can easily be edited and updated and its problems fixed properly. WHAN, which is established via a Wireless Sensor Network (WSN) and an IP, is granted to each peripheral to run the system via the internet too [6]. Having this access remotely enables the homeowner to perform preparation, such as turning an air conditioner ON for their return while they are away from home. Two types of Wi-Fi have been used: ESP8266 -01,12E. Each of these is important. An Ethernet shield and Router can be used as a third way. A smart home reduces the use of electric energy or can use alternative energy, such as solar energy [7]. It reduces costs and is environmentally friendly [8].

As an Iraqi student I study in Turkey and encountered several problems regarding establishing this project. In Iraq, such systems are new and a smart home designer usually faces numerous problems, like inability to get hold of the pieces for the project, high prices and unreliable delivery of the parts for a project. Such homes are very important for our country in terms of security, economy and a proper use of electricity. Such systems would work very well for those who are disabled and have lost their limbs in the continuous battles in our country [9].

In this paper, a prototype of a smart home is presented. Generally, the system consists of these parts: WiFi, IR, Keypad, GSM, Ethernet shield, Arduino (Uno and Mega), humidity, Temperature, Gas sensor, Flame sensor, ESP8266-01,12E, Buzzer, PIR and LCD.Numerous trials been carried out before the results of this paper were presented [10].

2. RELATED WORK

2.1. Smart Home System

Our system runs with Operating Systems (OS) such as Android and the Internet of Things (IoT), which are well known to those who deal with such systems. We have endeavoured to abide by the limitations of commonly used systems. Comfort, security and flexibility are some common features of the system [11]. This system is able to work with android phones, laptops, tablets or a

PC. The main components are: a board of Arduino (MEGA 2560& Uno R3), which can be called the brain of the system, a Router, Wi-Fi (Node MCU-ESP8266-12E and ESP8266-01) with a good range, an IR device, an Ethernet shield (network interface W5100), a GSM device and a User device with an Android OS, such as smartphone or computer (through the web). These devices are connected through either the internet or Wireless Application Protocols (WAP). The user controls the entire system through an android enabled device (by either a smart phone or tablet).

The system is also controlled through an IR device (control). This has made the system more desirable and famous, as it does not necessarily need the internet or a wired connection. The Wi-Fi network is the main grid. Through it, the instructions of the user are transmitted. Bluetooth and IR are also used inside the house as alternatives when needed despite their specific importance. The user's necessities are the foundation steps that the system is built upon.

Home appliances are also controlled through a relay circuit, which can be connected to many house devices, like heating, ventilation and air conditioning (HVAC) .

Along with the Arduino and network interface (W5100), the RDIF module and sensors also play a very important role in the system. An RFID (Radio Frequency Identification) reader and a key pad are integrated into the system, to sense the state changes of the main door of the house and to control it (door lock).

RFID is a module to transmit data over a short range. It includes a tag scanner/reader that is capable of scanning as many tags as we set to it. Every tag has an ID that can be determined (authorised or not authorised) [12].

All sensors communicate with a microcontroller (where all input data are processed), depending on the program; the microcontroller decides and sends signals to the user and related relays to operate the house devices.

Therefore, the flame or smoke detection sensor collaborates with an Arduino, which is attached to relays that open to the door and operate the fan when the gas or fire spreads inside the home. The Arduino also signals the buzzer to shoot out and the leads to come on. These all constitute the (fire alarm system) which can call the owner of the house (ringing) through the GSM system.

The GSM system is also activated by Pyroelectric Infrared Sensor (PIR), which is used to detect human bodies as it can detect any change in infrared radiation. When a human passes through this sensor, the temperature in the background will rise from room temperature to the body temperature and thus motion will be detected; i.e. it detects motion. The motion detector sensor collaborates with an Arduino and the GSM808, which contain a mini simcard (mobile simcard) which calls the number saved in the program.

The temperature sensor coordinates with an Arduino that is connected to an LCD and a relay that controls the heater or an air-conditioning system.

The light detector sensors collaborate with an Arduino which is connected to relays that control the lamps. Besides these, this system has a photo sensor (photo resistor) which is economic and friendly to the environment.

In our work, the essential concepts of the automated house system were included (that is developing daily).

This system has the ability to expand and be integrated as an up-to-date involved system, so new devices and more security levels may be added to the systems..

2.2 Wireless Sensor Network (WSNs) :

WSN is a collection of many sensor nodes. Each can sense, process and connect to a microprocessor unit to work together in a coordinated manner (Figure 1). Physical or environmental conditions, such as motion, temperature, humidity, smoke and gas, etc., are monitored by these sensors in a coordinated way [13].

These sensors run through an algorithm to the main control unit (Arduino Node MCU ESP8266-12E), which has much more computational power and links and acts as the main way that links the sensor nodes and the end users. The principal features of WSN include power consumption constraints for node; capability of accommodation with node and communication tasks; scalability to large devices; ability to resist hard environmental conditions, mobility of node and variation of nodes [14].

It is the WSN that plays a major part in enabling highly accurate sensor and actuation systems, in homes, buildings and surrounding spaces, by supplying a trusted, cheap and wide solution. Their tools can be used in current, as well as future, structures, without significant changes in the existing infrastructure [15].

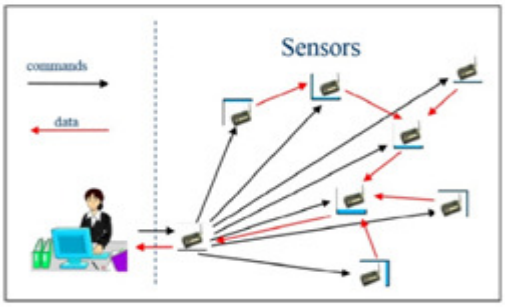


Figure 1 WSN Sensors

2.3 The Internet of Things:

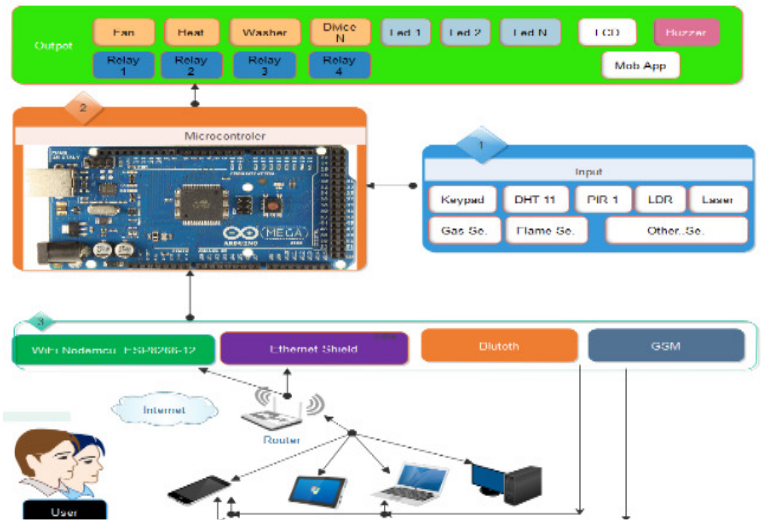
The Internet of Things refers to a network of objects where all things are uniquely and universally addressable, identified and managed by computers and smart phones. It is a collection of technologies that make it possible to connect things like sensors and actuators to the Internet (Figure 2) [16][17].

The Internet of Things is defined as an integrated part of the Future Internet and could be defined as a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual things have identities, physical attributes and virtual personalities, and use intelligent interfaces, and are seamlessly integrated into the information network [18].

Figure 2 The Internet of Things image.

2.4 System Block Diagram:

According to the design needs and desires, the system block diagram (Figure 3) is made. Our block diagram shows all the parts of the system and their functions. The project design follows the modular approach, using an Arduino Mega 2560 (Atmel 2560) microcontroller. The system is intended to enhance security, adaptability and effectiveness. The framework is planned to give ease in everyday life, and additionally spares power and human endeavour. This framework incorporates Arduino, sensors (like LPG gas, PIR, humidity and temperature), LCD show, LAN, 8 Relays channel, RFID, Keypad, smartphone and PC.



(Figure 3) System Block Diagram.

2.5. System Implementations

2.5.1 Project Flow:

This section explains the steps that need to be undertaken in order to achieve the goal of the project (Figure 4).

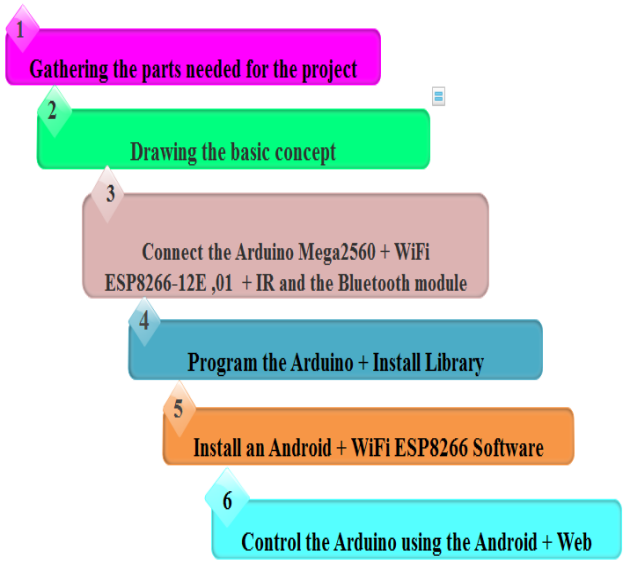


Figure 4 Project Flow

2.5.2. Hardware Implementation:

This includes PC, Laptop or Tablet, and Smart Phone, Arduino Mega 2560 (Figure 3.2.), Arduino UNOR3 (Figure 3.2.b), Breadboard Circuit, Ethernet Shield, Router, Led, LCD for Arduino, LDR Sensor, RFID Card System, Motion sensor, Servomotors, Temperature sensor, GSM Modem, smoke sensor and LPG gas sensor.

2.5.2.1 Arduino Mega 2560 (Figure 5) and Arduino UNOR3 (Figure 6)

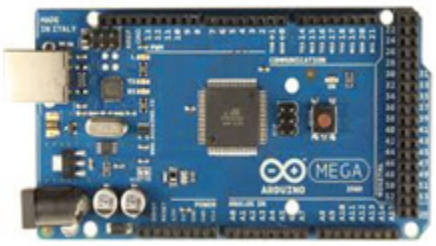


Figure 5 Arduino-Mega2560 Structure.

Figure 6 Arduino-UNO Structure.

2.5.2.2 WiFi Shield (ESP8266 WIFI SERIAL TRANSCEIVER):

The ESP8266 Serial-to-WiFi adapter has been widely adopted as a cost-effective solution for IoT and WiFi-capable devices. We used two types of ESP8266 in our project: MCU Node ESP8266-12E WIFI (Figure 7 and 8) and ESP8266-01 WIFI [19].

MODULE ESP826-01 is a cheap, high quality WiFi module with potent on-board processing and storage capacities [20]. (Figure 9).

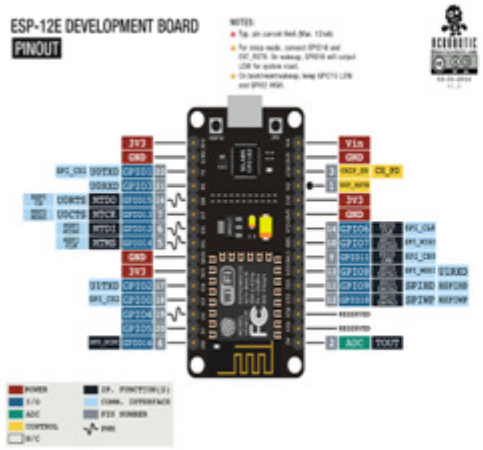


Figure 7 ESP8266-12E WIFI SERIAL TRANSCEIVER.

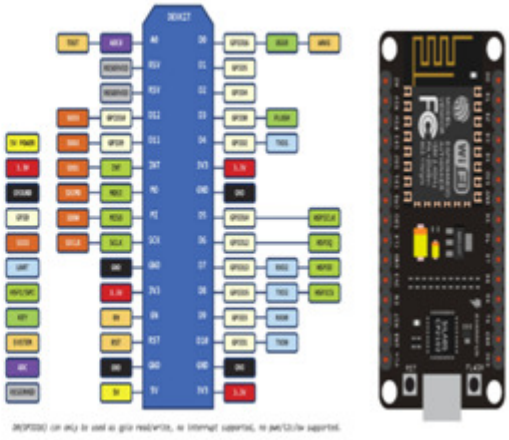


Figure 8 ESP8266-12E

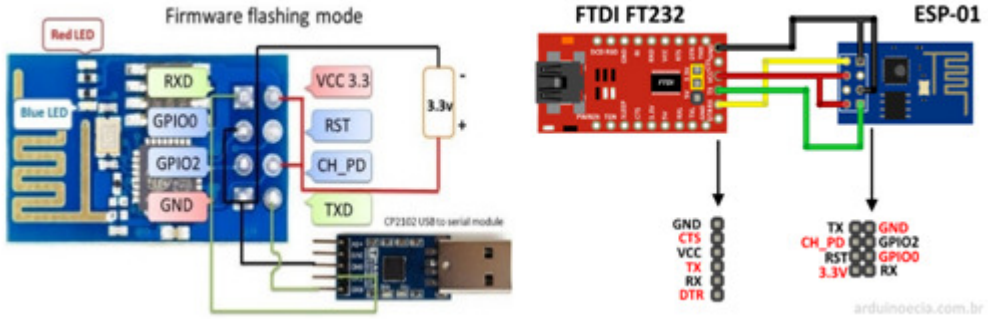


Figure 9 Top View of ESP8266-01

2.5.2.3.BluetoothHC-5

Bluetooth is a short field communication technology (less than10 metres) and is considered a low energy utilisation. It is not appropriate for all our demands. The information transmission rate is less than that of WI-FI, yet it is still beneficial for our application (720 Kb/s) (Figure 10)[21].

2.5.2.4. SIM808 Module (GPS-GSM-GPRS):

SIM808 module is a GSM, GPRS and GPS three-in-one limit module. In light of the latest GSM/GPS module SIM808 from SIMCOM, it supports GSM/GPRS Quad-Band framework and joins GPS innovation for satellite route (shipping). It is controlled by AT charge by methods for UART and sponsorships 3.3V and 5V clever level(Figure 11)[22].

2.5.2.6. Ethernet Shield (W5100)

In our design, W5100 interface has been used (Figure 12) this uses 0.18 µm CMOS technology and has 16Kbytes of memory (TX/RX buffert) to supply a 10Mb/100Mb Ethernet association [23].

Figure 10 Bluetooth HC-5 device Figure 11 SIM808 Module Figure 12 Ethernet shield

2.5.2.7. LAN Cable (Local Area Network):

To get to the web in Arduino LAN connection is required. The LAN speed is considerably quicker than the wireless speed [24] .

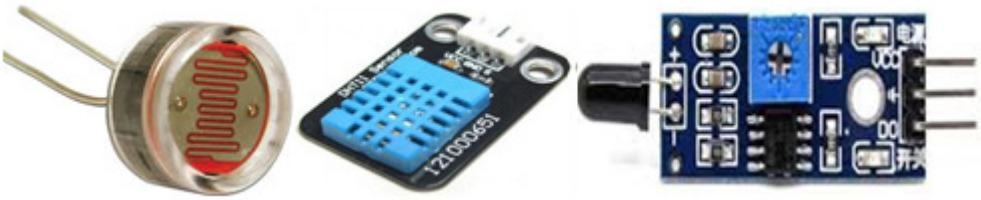
2.5.2.8. LCD Display:

LCD display is used to see the present status of the home devices and the sensors (password and temperature).

2.5.3. Sensors

The sensors used in this project include: photo resistor (Figure 13 a), temperature and humidity sensors (DHT11) (Figure 13 b), flame sensor (Figure 13 c), Gas Sensor (MQ-2) (Figure 13 d) and Human Body Pyroelectric Infrared Sensor (PIR) (Figure 13 e).

There are many distinctive gas sensors that recognise LPG, CO2, methane and fire. The case given here utilises the MQ2 sensor to distinguish air quality.



(a). Photo resistor (b). Humidity sensors (DHT11) (c). Flame sensor

(d). Gas Sensor (MQ-2) (e). Human PIR Sensor (anterior and posterior views)

Figure 13 (a, b, c, d and e) Sensors used in the project.

The following flowcharts show different sensors and their connection to the Arduinos and how temperature, humidity(Figure 14) and gases are detected (Figure 15):

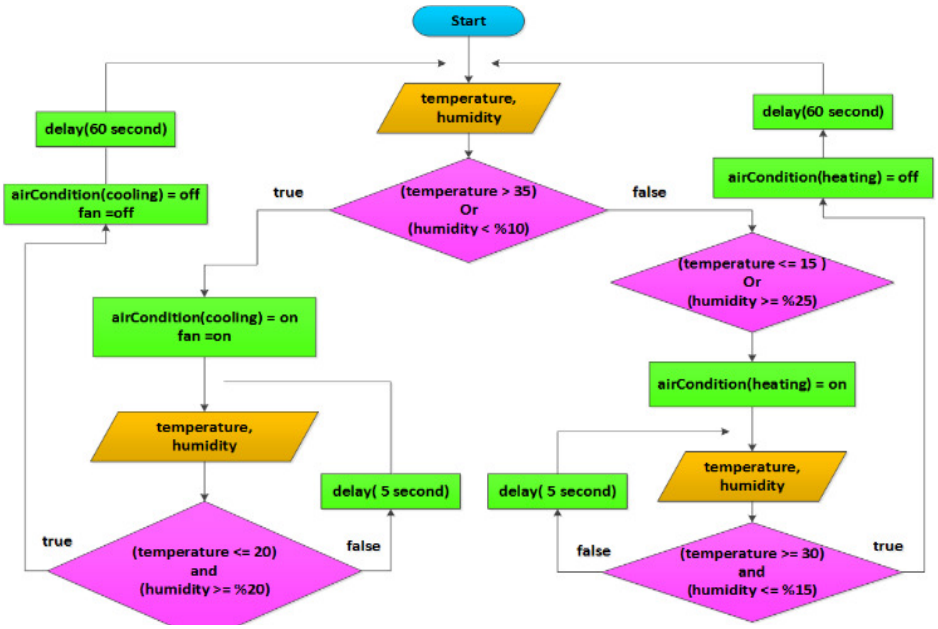


Figure 14. Flowchart of Temperature and Humidity sensors (DHT11).

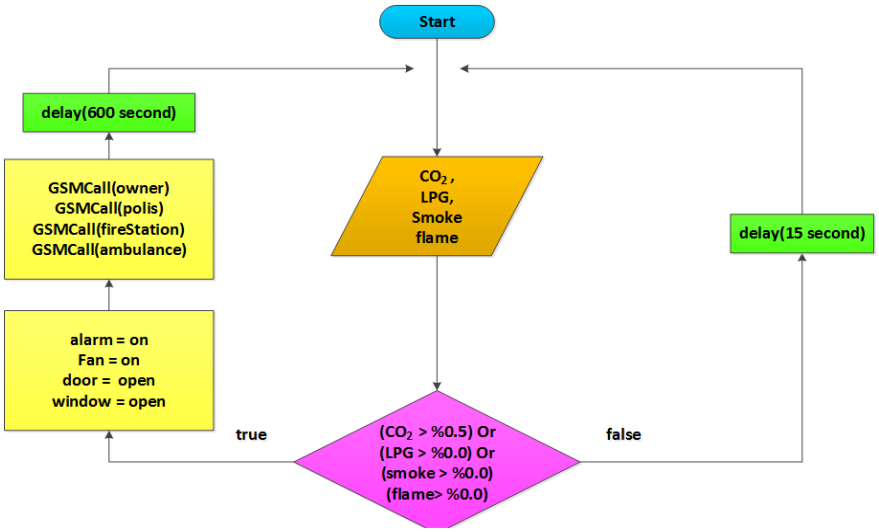


Figure 15. Flowchart of LPG gas sensor and Gas Sensor (MQ-2).

Human Body Pyroelectric Infrared Sensor (PIR): (Figure 13.e) feels any difference in IR radiation. In this manner, it could be utilised to distinguish movement. PIR's IC is BISS0001 Micro Power utilises low-energy CMOS technology. This sensor distinguishes the movement of an individual. As a human passes through this sensor, the background temperature (room temperature) will increase (to the body temperature) and in this manner the movement or human will be recognised [25]. This sensor may be used inside or outside the house. Outside the house: when the password (entered through a keypad) and/or RFID card (Figure 16) is used correctly, the buzzer (shooting) will stop, a green light will go on and the main door will be opened. When an incorrect password is entered or an RFID card is not used or wrongly used, the buzzer remains working, the red light will go on and the main door will remain closed. Inside the house, when the

PIR sensor is on, if it recognises any motion the buzzer starts to work and the red light goes on. The device starts calling the saved mobile numbers (residents or police station).

Figure 16 RFID RC522

The following flowcharts (Figure 17 and 18) show the sequence of detection of any motion by PIR and LASER, then alarming through sound (buzzer) an, light and GSM :

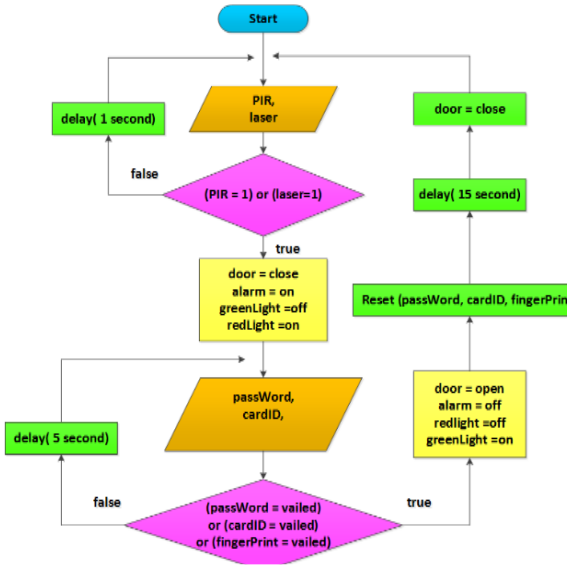


Figure 17. Flowchart of PIR and Laser

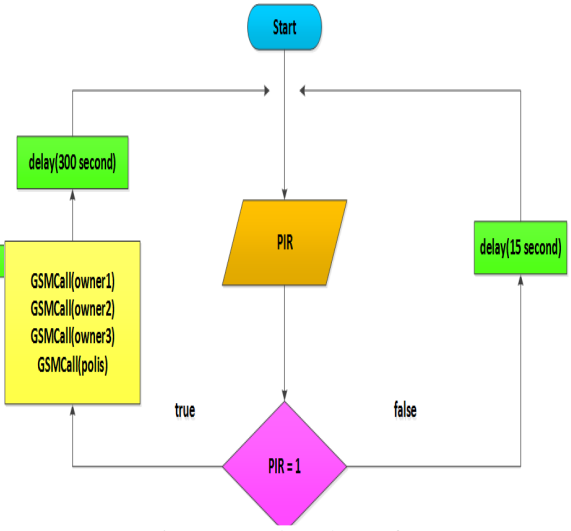


Figure 18. Flowchart of PIR

2.5.4. Software Implementation: (SYSTEM SOFTWARE DESIGN)

This part deals with the configuration of the software, program procedures and framework approaches used in programming the Arduinos in our smart home model. The software program roles are receiving information and orders, performing distinctive orders, controlling operational stations and supporting information input/output gates. We have attempted to offer general thoughts on program stream and usage (see flow charts). We divided our design into various parts, contingent upon the useful connection between these parts, to be dealt with easily and when a section or a piece is not working appropriately, it can be detected and treated.

The application created for this work (Arduino IDE 1.8.1) is responsible for the participation, communication and collaboration between the hardware and the portable PC or smart cell. It has been written in C language to follow the guidelines of coding for the project. WiFi MCU Node ESP8266-12E needs a software introduced inside Arduino IDE [26].

First we install the Arduino Software/IDE, then we open the Arduino program IDE, then select (File), where one can find the (Preferences) Dialog. Choose (Preferences), a (Settings) list will be

opened, in “Additional Boards Manager URLs" add this line and click on "OK":" http://arduino.esp8266.com/stable/package_esp8266com_index.json "then go to (tools again), from the list choose (board), select (boards manager) at the top. Select "ESP8266" and install it [27].

To determine the type of the board, connect the Board to the PC, then select Device Manager in (my computer). One will see the serial ports in (other devices). Again, select (tools) and choose the COM number (=serial port). In the same list (i.e. tools), choose type of the board (e.g. Node MCU ESP8266-12E or Arduino Mega 2560) [28].

The technique used to outline this product is an upper to lower down organised programming plan. The ports must be defined and determined in the start of the program along with the installing of required libraries then entering the code then uploading, and after that saving the code by a name inside the Arduino`s memory.

After the port locations and definitions, the program initially calls various libraries, e.g. Serial Peripheral Interface (SPI) Library, Ethernet Library, Temperature and Humidity Library and RFID Library.

Our smart home system software involves programming the Arduino by C and C++ languages (utilising IDE accompanies the microcontroller itself), and Applications such as Arduino IDE 1.8.1 for windows10, ESP8266 WIFI control, Arduino WIFI control and Arduino Bluetooth control device for Android smartphones.

The application program is a NET based application. Entering an IP in the URL of the browser connects the devices to the ESP. The application programs are mindful of setup and arrangement, and keep up the entire smart home system utilising a database to keep log of smart home system parts, we utilise XML files to spare system log.

The Arduino programming is through utilising C and C++ languages. Utilising IDE accompanies the microcontroller itself. Arduino software collects information about events from associated sensors, then applies activity to devices and what is pre-programmed in the server.

The communication inside the house is through the following:

1-TP-Link Router that boosts wireless network throughout the home, broadcasting the WIFI that connects MCU Node, ESP8266-12E, ESP8266-01 to smartphone and PC. The Router is also connected to the Ethernet shield via a cable (RJ-45), connecting it to an Arduino.

2-The GSM 808 communicates with a saved mobile number during any event that reaches an Arduino through any connected sensors. When you are outside the home, the communication is through the Internet. Via IoT we can control home devices (e.g. lamps and TV) from far away.

Table 1.The following table contains the parts of our project:

No	Components	Sensors	Display	Wireless Technology	Other
1 2 3 4 5 6 7 8 9 10 11 12	PC Smart Mobile Tablet Arduino Mega 2560 Arduino UNO R3 GSM GPRS	RFID Card System LDR Sensor Motion sensor Infrared Remote Control Temperature node Light control Door alarm buzzer Windows control Smoke sensors Gas leakage detection Flame Sensor Ultrasonic Sensor	LCD Monitor	WiFi ESP8266-01 WiFi Node MCU ESP8266-12E Bluetooth HC-5 IR Ethernet Shield Router	Registers Coil Switch Wire Transistor Breadboard Circuit Led buzzer Servo motors Solar panel tracking system

3. EXPERIMENTAL RESULTS

The planned procedure in building the smart home was successfully applied using equipment such as: WiFi, Sensor, GSM and Arduino in order to operate and administer all the instruments properly.

- The house can detect any body approaching it through PIR or LDR and Laser.
- It can be operated using a personal password by using RFID or Keypad, which turns on a green light that can be seen in the LCD screen.
- After entering the Password or the RFID the door opens automatically and then closes.
- In cases of entering incorrect passwords or using the system incorrectly, a red light and a buzzer turn on to give warning. It enables the user to switch the lights on and off. It also works automatically by turning the indoor and outdoor lights on when it is dark and turning all of them off when there is daylight(Figure 19).
- It also enables the user to check the temperature and humidity in the house, as well as working sensitively to detect any strangers entering the house and the existence of any gas, CO2, or fire. It also operates to take some necessary reactions and giving the user caution through their mobile phone, as well as operating ventilators and contacting related parties such as fire rescue services, police and ambulance.
- It is worth mentioning that the results can be accessed through web, smart phone and tablet using Bluetooth, WiFi, and IOT (Figure 21) (Figures 20).
- The user can also be informed about the amount of water left in the roof storage tank. They can also stop water consumption at a certain point, which means controlling water consumption for the purpose of protecting it as an important national treasure.
- Reducing energy consumption, which is vital for the environment and the economy at the same time, as most of the equipment only works when necessary.
- The system is also able to draw and close all the curtains and drapes in the house.

- Using Bluetooth and IR the user is also able to turn on/off and use most of the house equipment.
- All the above-mentioned points facilitate their users in so many aspects which gives them comfort, protection, security and insurance.
- Eventually, by using Solar Tracking generation of light, we managed to obtain more powers of light. The system is able to search for light automatically in a horizontal and a vertical way (Figure 22).

In short, this system is able to work with all its pieces through the internet properly and show the results, as well as performing the tasks using a computer, mobile phone or tablet. It also suggests different ways to activate and use the system properly.

System (IOT): Android implementation`s capacity for loading and carrying out scripts and incorporating a library to create and plan the Android application is valuable in our work. Show the web interface and Android Application Interface that have been used in our system (IOT) (Figures 19 (a, b, c, and d)).

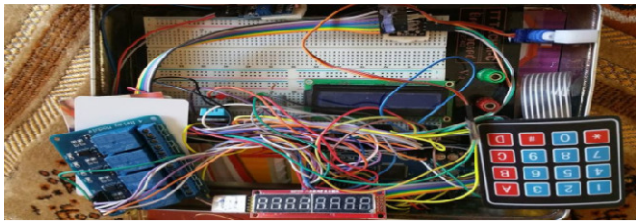
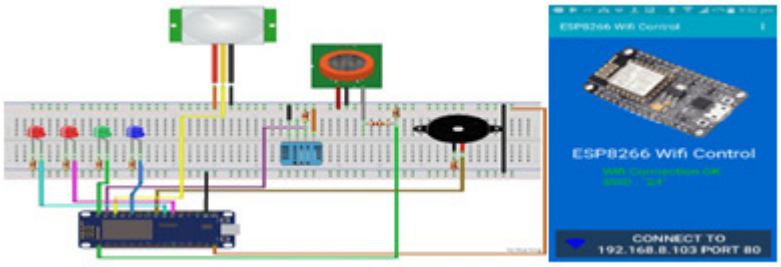


Figure 19. The result of this Control the Home



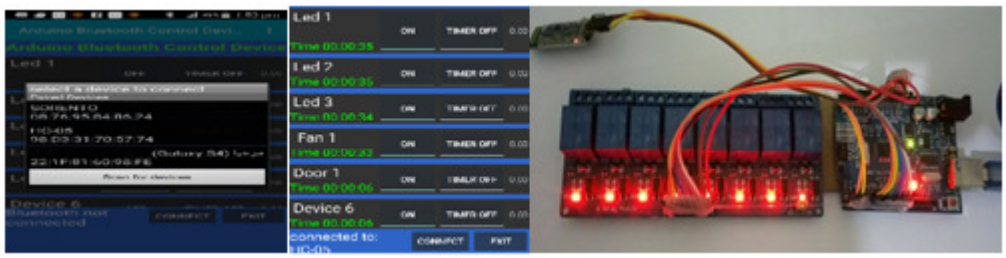
(a) ESP8266-12E WiFi Circuit (b) ESP8266-12E WiFi Control

(c) ESP8266-01 (d)Result IOT(Web Interface Node MCU ESP8266-12E)

Figures 20. Android Application Interface and Web using WIFI used in our system.

Bluetooth System: Android Application Interface using Bluetooth that has been used in our system with relays(for AC) (Figure 21 (a, b, c, d and e)).

(a) Bluetooth HC-05. (b) Arduino Bluetooth Control Device (c) Arduino Bluetooth Control



(d)Connects mobile and Bluetooth. (e)Setting the application. (e) Result the Bluetooth Section.

Figure 21. Android Application Interface using Bluetooth that has been used in our system.

Solar Tracker: In this section result of the project which is specific in (Figure 22).

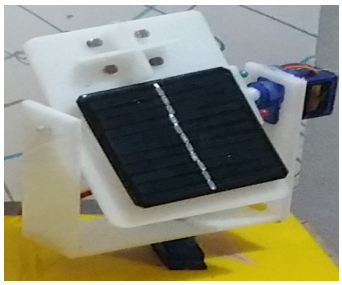


Figure 22. Solar Tracker

4. CONCLUSION

- Nowadays, technology has developed in so many ways. In this respect, smart homes could be built considering the following important points:

1. Making sure they are strong and effective as well as being user friendly.
2. Considering security and safety aspects.
3. Making sure it costs little and is economic.
4. For the development of this project, the Solar Tracking generation of light is used to save more energy.
5. Building in a way which leads to less energy consumption and is environmentally friendly.
6. Making sure it is easy to develop as well as easy to mend.

- Smart home is able to connect a number of machines and sensors together and enables them to work properly and automatically. It also facilitates its users, especially disabled people.

- In this paper, Smart Home Security is considered, so as to be applied properly. It suggests two different ways to connect it to internet, so that it can be used and observed from a distance. This can be achieved through the web by using a PC, laptop, smart phone and tablet after entering their special IP and connecting them together through WSN.

- The paper also suggests using more than one type of wireless, such as IR, Bluetooth and WiFi, in the same project.

- In case of any unexpected incidents, such as strangers entering the house, the existence of fire, smoke, CO2 gas, or gas exposure, etc. the project suggests building a relationship between the smart home, its user and related parties through GSM.

- The paper also focuses on protecting the environment, reducing electrical energy consumption and avoiding the waste of water. These are the main points, as the mentioned sources are among the main issues in our country as well as being the primary daily needs that are not available these days.

- It is recommended to develop this project and connect it to Face Recognition and Voice Recognition, which aims to enhance the facility and provide more safety and security at the same time.

- It also considers children, patients and elderly people in order to provide them with better care, by observing and watching their movement, safety, body temperature, heart beats, etc. so that the system contacts the owner and ambulance services in any problematic situations.

Advantages of our System:

The implementation of a Smart House Application Using Wireless Sensor Networks was managed successfully as it was user friendly and cost effective, because of better flexibility via an android device through the web and a wider range of scalability. This system provided comfort, authentication and security.

Future Recommendations

1. Each component is able to be tested before using them, especially the relays for safety purposes.

2. Instead of WiFi, IR and Bluetooth modules, try LiFi technology which is 100 times faster than WiFi and which amplifies the signal for working at a greater distance.

REFERENCES

[1] Jackie Craven, "What Is a Smart House?" [Online]. Available: http://architecture.about.com/od/buildyourhous1/g/smarthou e.htm. [2017, May 14].

[2] Andi Adriansyah, Akhmad Wahyu Dan, "Design of Small Smart Home System Based on Arduino", Electrical Power, Electronics, Communications, Controls, and Informatics Seminar (EECCIS), Malang, Indonesia, 121 p., 2014.

[3] Mohd Nor Azni, M. N. H., Vellasami, L., Zianal, A.H., Mohammed, F. A.., Mohd Daud, N. N., Vejasegaran, R; N. W.Basharudin; Jusoh, M; Ku Azir, K.N.F., P. L. Eh Kan. "Home Automation System with Android Application", 3rd International Conference on Electronic Design (ICED), August 11-12, 2016, Phuket, Thailand.

[4] Jen Rossey, Ingrid Moerman, Piet Demeester, Jeroen Hoebeke, "Wi-Fi helping out Bluetooth Smart for an improved home automation user experience", Ghent University – iMinds, Department of Information Technology (INTEC),Technologiepark-Zwijnaarde 15, 9052 Ghent, Belgium, 978-1-5090-4361-3/16/$31.00 ©2016 IEEE.

[5] "Internet of Things Strategic Research and Innovation Agenda" p. 24

[6] Hemant Ghayvat, Subhas Mukhopadhyay, Xiang Gui and Nagender Suryadevara , "WSN- and IOT-Based Smart Homes and Their Extension to Smart Buildings".

[7] Oihane Kamara-Esteban, Gorka Sorrosal, Ander Pijoan, Tony Castillo-Calzadilla, Xabier Iriarte-Lopez, Ana M. Macarulla-Arenaza, Cristina Martin, Ainhoa Alonso-Vicario, Cruz E. Borges, "Bridging the Gap between Real and Simulated Environments: a Hybrid Agent-Based Smart Home Simulator Architecture for Complex Systems", 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress, 2016 IEEE pp. 220-227. DOI: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.116 https://www.researchgate.net/publication/305710190_Bridging_the_Gap_between_Real_and_Simulated_Environments_a_Hybrid_Agent-Based_Smart_Home_Simulator_Architecture_for_Complex_Systems

[8] Mohamed Hadi Habaebi, Nur Izzati Nabilah Bt Azizan, "Harvesting WiFi Received Signal Strength Indicator (RSSI) For Control/Automation System In SOHO Indoor Environment with ESP8266", Department of Electrical and Computer Engineering, IIUM habaebi@iium.edu.my,416-418pp.

[9] Taylor, Joseph, et al. "SenseBox: A low-cost smart home system." Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on. IEEE, 2017.

[10] R.Aravindhan, M.Ramanathan, D.SanjaiKumar, R.Kishore, "HOME AUTOMATION USING Wi-Fi INTERCONNECTION",Department of Mechatronics engineering, Chennai Institute of Technology, Anna University Chennai, Tamil Nadu, India- 600069, (IRJET) e-ISSN: 2395 -0056 Volume: 04 Issue: 03 | Mar -2017 ,www.irjet.net.2542-2544pp.

[11] Andi Adriansyah, Akhmad Wahyu Dani, "Design of Small Smart Home System Based on Arduino" Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana Jl. Raya Meruya Selatan, Kembangan, Jakarta, 11650, Indonesia,121p.

[12] Ehsan Dolatshahi, "Smart Home Automation System." CALIFORNIA STATE UNIVERSITY, NORTHRIDGE. December 2015. 35p.

[13] Mantoro, Teddy and Ayu, Media Anugerah and Ali, Haroon Shoukat, Usino, Wendi and Kadhum, Mohammed M. (2012), "Energy efficiency mechanisms using mobile node in wireless sensor networks". Networked Digital Technologies: Communications in Computer and Information Science, 293. pp. 536-550. ISSN 1865-0929.
http://irep.iium.edu.my/24767/1/Energy_Efficiency_Mechanisms_Using_Mobile_Node.pdf

[14] Andreas Kamilaris, " ENABLING SMART HOMES USING WEB TECHNOLOGIES", University of Cyprus. December, 2012.

[15] Karwan MUHEDEN, Ebubekir ERDEM, "Mobile Design and Implementation Fire Alarm System Using Wireless Sensor Network", Firat University,2017.

[16] Andreas Kamilaris, "ENABLING SMART HOMES USING WEB TECHNOLOGIES", University of Cyprus. December, 2012. 60p.

[17] Muhammad Asadullah, Ahsan Raza, "An Overview of Home Automation Systems", National University of Computer and Emerging Sciences Peshawar, Pakistan, 978-1-5090-4059-9/16©2016 IEEE ,30p.

[18] Internet of Things – Strategic Research Roadmap (IoT-SRR). 15 September, 2009.

[19] http://espressif.com/products/hardware/esp8266ex/overview/

[20] Rui Santos, "Home Automation Using-Esp8266",2nd Editions, Version 2.1.

[21] Muhammad Asadullah, Ahsan Raza, "An Overview of Home Automation Systems", National University of Computer and Emerging Sciences Peshawar, Pakistan, 978-1-5090-4059-9/16©2016 IEEE ,27-29pp.

[22] R.HARINATH, Dr. S. Santhi, "GSM BASED HOME AUTOMATION SYSTEM USING APP-INVENTOR FOR ANDROID MOBILE PHONE", IJCSMC, Vol. 4, Issue. 4, April 2015, pg.158 – 167,158-159pp.

[23] Andi Adriansyah, Akhmad Wahyu Dani, "Design of Small Smart Home System Based on Arduino" Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana Jl. Raya Meruya Selatan, Kembangan, Jakarta, 11650, Indonesia,126p.

[24] Shruti G. Suryawanshi1, Suresh A. Annadate, "Implementation of Smart Home Automation System through E-mail using Raspberry Pi and Sensors", Electronics Engineering Dept., M.G.M's Jawaharlal Nehru Engg., College, Aurangabad, India, DOI 10.17148/IJIREEICE. 2016.4347,181p.

[25] https://www.arduino.cc/en/main/software accessed on May 1st, 2017.

[26] http://blog.ubidots.com/esp8266-arduino-ide-tutorial accessed on May 1st, 2017.

[27] Rui Santos ,"ESP8266 Web Server with Arduino IDE", accessed on jul 2017.

AUTHORS

Ismail Asaad MOHAMMED
Nationality: Iraqi
Place of birth: Sulaimani-Iraq
Date of birth: 01 / 07 / 1972
Marital status: Married
Mobile: 00964(0)7701537418 - 00964(0)7501537418
E-mail: ismail.asaad@spu.edu.iq
ismailasaad1972@gmail.com

EDUCATION

2/2015 – Started MSc (Master of Computer Engineering Department), Firat University, Elazig, Turkey.
9/2003 – 7/2007 Bachelor degree from Kirkuk University, College of Science, Computer Science Dep. Iraq, Kirkuk.
9/2010 – 6/2012 Diploma, Institute of Sermon in Sulaimani, Iraq.
9/1991 – 6/1993 Diploma, Medical Institute. Institute of Technology in Mosul, Mosul, Iraq.
9/1988 – 6/1991 Higher School from Chamchammal Preparatory School, Sulaimani, Iraq.
9/1985 – 6/1987 Intermediate School, Ronaky Intermediate School, Sulaimani, Iraq.
9/1978 – 6/1984 Primary School, Chamchammal Primary School, Sulaimani, Iraq.

WORK EXPERIENCES

- Volunteer as Assist. Doctor at faqya Merza health centre 8/1996-5/1997.
- Assist. Doctor at Shahid Peshraw Hospital 5/1997-10/2010.
- Visiting Teacher at Dukan Technical Institute (Practical Teaching). 10/2007 - 02/2013.
- IT department decision's at Computer Science Institute 10/2012-07/2014.
- Teacher Computer Science at Sardam Nongovernmental Computer Institute. 10/2013.
- Assistant programmer at Sulaimani Polytechnic University 07/2014.

Erkan DUMAN
Nationality: Turk
Place of birth: Elazig - TURKEY
Date of birth: 02 / 02 / 1979
Marital status: Single
Mobile: 0090(0)533 660 4963
E-mail: erkanduman@firat.edu.tr
erkan_duman@hotmail.com

EDUCATION

2003 – 2010 PhD., Electric Electronic Engineering Department, Firat University, Elazig, TURKEY.
2001 – 2003 Master, Computer Engineering Department, Firat University, Elazig, TURKEY.
1997 – 2001 Undergraduate, Computer Engineering Department, Firat University, Elazig, TURKEY.

WORK EXPERIENCES

- Research Assistant, Computer Engineering Department, Firat University, Elazig- TURKEY, 2001-2011
- Asst. Prof., Computer Engineering Department, Firat University, Elazig- TURKEY, 2011-2017

A FOUR-VALUED LOGIC

J. Ulisses Ferreira

Trv Pirapora 36 Costa Azul, 41770-220, Salvador, Brazil

ABSTRACT

This short and informal article briefly introduces a four-valued logic.

KEYWORDS

four-valued logic, logic, philosophy of computer science

1. INTRODUCTION

In early 20th century, Kleene, Łukasiewicz and Priest individually proposed their three-valued logics[1], whereas Łukasiewicz and Tarski introduced their propositional calculi into the literature[2], while a considerable number of contributions appeared on many-valued logics.
The paraconsistent logic was simultaneously and independently created by two logicians, namely, the Polish logician Stanisław Jaśkowski and the Brazilian logician Newton da Costa[3].

In 1977, Nuel Belnap (1930-) proposed his logic on four values[4]. His two extra values are *N* (none) and *B* (both) in the four-valued Belnap's logic, and they correspond to *uu* and *ii*, respectively, in both referred logics of the present author, as well as in his four-valued logic introduced in this paper.

In 1988, the present author started his Master research inserting a third value called "unknown" in a programming language[5] that he was designing at that time. The *unknown* constant has been theoretically referred to as *uu* since the end of nineties. Later, a five-valued logic was introduced containing the values in *{tt, ff, uu, ii, kk}*. In 2004, the same logic was published as a journal article[6] and a seven-valued logic was also published in a Conference in San Diego[7], adding the values *{fi, it}* ("false or inconsistent", "inconsistent or true", respectively) for being able to be used together with the same uncertainty model that had been proposed during the present author's Master course in 1990.

As part of the present author's previous contribution, the *kk* value means "knowable", and it is usable when something is not already known, but it is already known that it is consistent. It can either be *true* or *false* but not both. It can be known by God or someone else or some machine, for instance, but it is not already known by the machine which is deductively reasoning, or by the person who is deductively doing, and it may be unknown forever, but at least its consistency is guaranteed beforehand. This is the meaning of the *kk* value, which fits in the referred uncertainty model when a variable thresholds collapse: *False* = *True*, which means that there is nothing strictly between the *False* and the *True* thresholds.

In 2017, the four-valued logic being introduced was briefly presented in [8].

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 71– 84, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71206

Section 2 introduces the present author's four-valued logic. Section 3 makes some comparisons to other logics. Section 4 shows the program that the present author wrote to discover his logic. Subsection 4.1 presents an example of input data whereas subsection 4.2 shows a C source code. Section 5 presents the properties of the logic that is being introduced, whereas section 5 contains the conclusions.

2. FERREIRA FOUR-VALUED TRUTH TABLES

Table 1. Ferreira four-valued logic truth tables

A	¬A
U	I
F	T
T	F
I	U

∧	U	F	T	I
U	U	F	U	U
F	F	F	F	F
T	U	F	T	I
I	U	F	I	I

∨	U	F	T	I
U	U	U	T	I
F	U	F	T	I
T	T	T	T	T
I	I	I	T	I

→	U	F	T	I
U	T	T	T	T
F	T	T	T	T
T	U	F	T	T
I	F	U	T	T

↔	U	F	T	I
U	T	F	F	F
F	F	T	F	F
T	F	F	T	F
I	F	F	F	T

3. SOME COMPARISONS

In some multi-valued logics, which regard uncertainty such as probability, after some considerations, the numeric values (e.g. in the interval [0,1]) are translated to a few values using words (such as true, false, unknown and inconsistent) in order to be used in sentences and complex expressions containing the well-known logical operators. In the end, one computes and knows the logical value of the whole expression.

In da Costa's paraconsistent logic, *(tt ∨ ff = ii)* as well as *(tt ∧ ff = uu)* which do not look natural. In Belnap's four-valued logic, *(B ∨ N = T)*, where *T* represents the true value. That is to say, in terms of applications, the expression *(ii ∨ uu = tt)* does not look natural either. Such criticisms led to the work on the logic being introduced here.

4. A PROGRAM FOR HELPING DISCOVER THE DESIRED LOGIC

In 2007, having the present author set his own requirements in a form towards constraint programming, and written the corresponding program below, which seems to be correct, he checked whether such a four-valued logic which he desired exists, and its computation resulted in several such logics. He observed that one of the solutions was Belnap's four-valued logic. One of the solutions written by the program computation was chose for being the logic which was the most interesting. In 2011, one could not claim the authorship of that four-valued logic, but the referred truth tables are those introduced in section 2, above.

The computation of the program `a4vlogic.c` makes use of the input data `a4vlogic.dat`, and, before resuming, it generates the output file called `a4vlogic.sol` containing the solutions found if any, and reads and writes an auxiliary file called `a4vlogic.aux`.

The input data contains the truth tables for ∧, ∨, ¬ as well as ↔ operators, respectively, followed by the → partial truth table. Each truth table element (implicitly indexed by a row and a column) contains one of the four values. Originally, the value was not in the same order as shown in section 2, table 1, but in the following order, *(ff, uu, ii, tt)*, and the labels of the tables are not written, as they are implicitly represented by just their contents, except for the output on screen as well as the row labels written to the output file. Wherever one of the four values is read from the input file, it means that the same value is a constraint programmed by the present author. The

only truth table that is not fully filled in is the truth table for →. In subsection 4.1, all entries of the truth table for implication are left for the program, although some constraints could be imposed. In fact, there were a number of experiments, and subsection 4.1 just presents the last one. In every table entry that contains "..", it means that the program role is to try all possibilities for that entry and, thus, to produce all possible and different truth tables. In this way, the program permits the corresponding logician to either impose a value or let the program to try all n possibilities in that entry, where $n = 4$ in the present logic. In terms of efficiency, the program can be improved for eliminating an intermediate number of possibilities if the corresponding logician wishes, in such a way that its computation can be carried out much faster, for some large n. Since 4, 5 and 7 are relatively small numbers for the present C program, and just those numbers have been part of the present author's work, he has not been concerned about making the suggested improvement.

The auxiliary file is important for the program can be easily adapted for another number n of values. The greater this number n is, the longer time the computation takes. If n is sufficient large, it is important that the computation often write the current state in a file, in particular, because the computation was being performed in the present author's personal computer, which can be sometimes switched off. In the present logic, where $n = 4$, the response was as fast as while the present author was slipping in one night. Nonetheless, for greater n, the computation may take even many years for giving the first solution if there is any.

Whenever the program starts running, the calculation of the possible logics continues from the point contained in the auxiliary file. Thus, this file contains the last state only, whereas the output file is meant to concatenate all the solutions that are being found, if any.

While in the loop, for regarding the current truth table as a solution, the program computation checks more than 12 properties, which are written as part of the C code. If all desired properties hold, the program computation appends the current truth table as a new solution, advances one value, and the loop carries on.

4.1 THE CONSTRAINTS

```
ff ff ff ff
ff uu uu uu
ff uu ii ii
ff uu ii tt

ff uu ii tt
uu uu ii tt
ii ii ii tt
tt tt tt tt

tt ii uu ff

tt ff ff ff
ff tt ff ff
ff ff tt ff
ff ff ff tt

.. .. .. ..
.. .. .. ..
.. .. .. ..
.. .. .. ..
```

4.2 THE C CODE

```
#include<stdlib.h>
#include<stdio.h>
#include<time.h>
#include<string.h>
//#include<sys/time.h>

// NN é o número de linhas ou de colunas da matriz lógica verdade
#define NN 4
#define SN "4"

// M = NN - 1 ?
#define M 3
// IX é NN * NN - 1
#define IX 15
// NN2 = NN * NN
#define NN2 16

// NNM1 = NN + 1
#define NNM1 5

// True, o índice do valor máximo, = M
#define TT 3

//#define BT 8
//#define IT 7
//#define UU 3

typedef char word[3];

// "tt" e ".." devem ser sempre os últimos símbolos,
// e "ff" deve ser o primeiro.
const word v[NNM1] =
{"ff", "uu", "ii", "tt", ".."};
// "ub", "ff", "fb", "fi", "bb", "kk", "ii", "bt", "it", "tt",
"ui", ".."};

#define FALSE 0
#define TRUE 1

typedef int tab[NN2][NN2];

typedef struct {
  int i, j;
} indices;

// índices de 0 a IX
indices hash[NN2];

tab and,or,imp,eqv,impmsk/*,imp2,imp3,imp4*/;

int not[NNM1];
```

```
FILE *in, *sol;

int nn = -1;

void ler(char im, tab t) {
  word w;
  char achou;
  int i, j, k;
  for (i=0; i<NN; i++) {
      for (j=0; j<NN; j++) {
        fscanf(in,"%s ",w);
        printf("w = %s\n",w);
        achou = FALSE;
        for (k = 0; !achou && k<=M+1; k++) {
            if (strcmp(w,v[k])==0) {
              achou = TRUE;
              t[i][j] = k;
              if (k == M+1) {
            if (im) {
                    hash[++nn].i = i; hash[nn].j = j;
                    imp[i][j] = 0;
            }
            else {
                    printf(
"Erro: máscara %s fora da tabela da implicação.\n",w);
                    exit(1);
            }
        }
         }
        }
       if (!achou) {
         printf("Dado incorreto no arquivo de entrada: %s.\n",w);
            exit(1);
        }
      }
  }
  printf("LEU\n");
}

void escreve(FILE *f, tab t) {
  int i,j;
  for (i=0; i<NN; i++) {
      fprintf(f,"%s |",v[i]);
      for (j=0; j<NN; j++) {
        fprintf(f," %s",v[t[i][j]]);
      }
      fprintf(f,"\n");
  }
  fprintf(f,"\n");
}

unsigned long exp(int base, int x) {
  unsigned long n = 1;
```

```
  while (x > 0) {
    n = n * base;
    x--;
  }
  return n;
}

tab A, B, C;

void err(char *e, char n) {
  (*e)++;
  printf("Erro #%d\n",n);
}

char mask(tab imp) {
  int a, b, c;
  char erro = 0;

  a = 0;
  while (erro == 0 && a<=M) {
    // nagação dupla
    if (a != NN && not[a] != NN && eqv[a][not[not[a]]] != TT)
      err(&erro,1);

    // P1. Identity:
     if (imp[a][a] != NN && eqv[imp[a][a]][TT] != TT)
err(&erro,2);

    b=0;
    while (erro == 0 && b<=M) {

      // contrapositiva
      if (imp[a][b] != NN && imp[not[b]][not[a]] != NN &&
          eqv[imp[a][b]][imp[not[b]][not[a]]] != TT)
err(&erro,3);
      // a and b --> a or b
      if (imp[and[a][b]][or[a][b]] != NN &&
          eqv[imp[and[a][b]][or[a][b]]][TT] != TT) err(&erro,4);

       // De Morgan 1
      if (a != NN && b != NN && or[a][b] != NN &&
          not[or[a][b]] != NN && not[a] != NN && not[b] != NN &&
          and[not[a]][not[b]] != NN &&
          eqv[not[or[a][b]]][and[not[a]][not[b]]]!=TT) {
         printf("a=%d b=%d\n");
           err(&erro,5);
      }

      // De Morgan 2
      if (a != NN && b != NN && and[a][b] != NN &&
          not[and[a][b]] != NN && not[a] != NN && not[b] != NN &&
          or[not[a]][not[b]] != NN &&
        eqv[not[and[a][b]]][or[not[a]][not[b]]]!=TT) {
```

```
          printf("a=%d b=%d
%d=%d\n",a,b,not[and[a][b]],or[not[a]][not[b]]);
            err(&erro,6);
      }

      /*
      if (imp[a][b] != imp2[a][b] && imp[a][b] != imp3[a][b]
       && imp[a][b] != imp4[a][b]) erro++;
      */

      // P6. a => (b => a)
      if (a != NN && b != NN && imp[b][a] != NN &&
          imp[a][imp[b][a]] != NN &&
          eqv[imp[a][imp[b][a]]][TT] != TT) err(&erro,7);

      // P7. ((a => b) => a) => a
      if (a != NN && b != NN && imp[a][b] != NN &&
          imp[imp[a][b]][a] != NN &&
          imp[imp[imp[a][b]][a]][a] != NN &&
          eqv[imp[imp[imp[a][b]][a]][a]][TT] != TT) err(&erro,8);

      c = 0;
      while (erro == 0 && c<=M) {
        // P2. (a => (b => c)) => (b => (a => c))
        if (a != NN && b != NN && c != NN && imp[b][c] != NN &&
            imp[a][imp[b][c]] != NN && imp[a][c] != NN &&
            imp[b][imp[a][c]] != NN &&
            imp[imp[a][imp[b][c]]][imp[b][imp[a][c]]] != NN &&
            eqv[imp[imp[a][imp[b][c]]][imp[b][imp[a][c]]]][TT] !=
TT)
              err(&erro,9);
        // P3. (c => a) => ((b => c) => (b => a))
        if (c != NN && a != NN && imp[c][a] != NN &&
            b != NN && imp[b][c] != NN && imp[b][a] != NN &&
            imp[imp[b][c]][imp[b][a]] != NN &&
            imp[imp[c][a]][imp[imp[b][c]][imp[b][a]]] != NN &&
            eqv[imp[imp[c][a]][imp[imp[b][c]][imp[b][a]]]][TT] !=
TT)
              err(&erro,10);
        // P4. (c => a) => ((a => b) => (c => b))
        if (c != NN && a != NN && imp[c][a] != NN &&
            imp[a][b] != NN &&
            imp[c][b] != NN && imp[imp[a][b]][imp[c][b]] != NN &&
            imp[imp[c][a]][imp[imp[a][b]][imp[c][b]]] != NN &&
            eqv[imp[imp[c][a]][imp[imp[a][b]][imp[c][b]]]][TT] !=
TT)
              err(&erro,11);
        // P5. (a => (b => c)) => ((a => b) => (a => c))
        if (a != NN && b != NN && c != NN && imp[b][c] != NN &&
            imp[a][imp[b][c]] != NN &&
            imp[a][b] != NN &&
            imp[a][c] != NN && imp[imp[a][b]][imp[a][c]] != NN &&
            imp[imp[a][imp[b][c]]][imp[imp[a][b]][imp[a][c]]]
```

```
            != NN && eqv[
               imp[imp[a][imp[b][c]]][imp[imp[a][b]][imp[a][c]]]
            ][TT] != TT) err(&erro,12);
        /*
        // associativa and
        if (eqv[and[and[a][b]][c]][and[a][and[b][c]]]!=TT)
erro++;
        // associativa or
        if (eqv[or[or[a][b]][c]][or[a][or[b][c]]]!=TT) erro++;
        // distributiva and-or
        if (eqv[or[and[a][b]][c]][and[or[a][c]][or[b][c]]]!=TT)
          erro++;
        // distributiva or-and
        if (eqv[and[or[a][b]][c]][or[and[a][c]][and[b][c]]]!=TT)
          erro++;
        */
        c++;
      }
      b++;
    }
    a++;
  }
  return erro == 0;
}


char tabok(tab imp) {
  int a, b, c;
  char erro = 0;

  a = 0;
  while (erro == 0 && a<=M) {
    // nagação dupla
    if (eqv[a][not[not[a]]] != TT) erro++;

    // P1. Identity:
    if (eqv[imp[a][a]][TT] != TT) erro++;

    b=0;
    while (erro == 0 && b<=M) {

      // contrapositiva
      if (eqv[imp[a][b]][imp[not[b]][not[a]]] != TT) erro++;
      // a and b --> a or b
      if (eqv[imp[and[a][b]][or[a][b]]][TT] != TT) erro++;

      // De Morgan 1
      if (eqv[not[or[a][b]]][and[not[a]][not[b]]]!=TT) erro++;
      // De Morgan 2
      if (eqv[not[and[a][b]]][or[not[a]][not[b]]]!=TT) erro++;

      /*
      if (imp[a][b] != imp2[a][b] && imp[a][b] != imp3[a][b]
```

```
       && imp[a][b] != imp4[a][b]) erro++;
      */

      // P6. a => (b => a)
      if (eqv[imp[a][imp[b][a]]][TT] != TT) erro++;

      // P7. ((a => b) => a) => a
      if (eqv[imp[imp[imp[a][b]][a]][a]][TT] != TT) erro++;

      c = 0;
      while (erro == 0 && c<=M) {
        // P2. (a => (b => c)) => (b => (a => c))
        if (eqv[
              imp[imp[a][imp[b][c]]][imp[b][imp[a][c]]]
            ][TT] != TT)
          erro++;
        // P3. (c => a) => ((b => c) => (b => a))
        if (eqv[
              imp[imp[c][a]][imp[imp[b][c]][imp[b][a]]]
            ][TT] != TT) erro++;
        // P4. (c => a) => ((a => b) => (c => b))
        if (eqv[
              imp[imp[c][a]][imp[imp[a][b]][imp[c][b]]]
            ][TT] != TT) erro++;
        // P5. (a => (b => c)) => ((a => b) => (a => c))
        if (eqv[
              imp[imp[a][imp[b][c]]][imp[imp[a][b]][imp[a][c]]]
            ][TT] != TT) erro++;

           // transitiva
           if (imp[and[imp[a][b]][imp[b][c]]][imp[a][c]] != TT)
             erro++;
           // associativa and
           if (eqv[and[and[a][b]][c]][and[a][and[b][c]]]!=TT)
             erro++;
           // associativa or
        if (eqv[or[or[a][b]][c]][or[a][or[b][c]]]!=TT)
          erro++;
        // distributiva and-or
        if (eqv[or[and[a][b]][c]][and[or[a][c]][or[b][c]]]!=TT)
          erro++;
        // distributiva or-and
        if (eqv[and[or[a][b]][c]][or[and[a][c]][and[b][c]]]!=TT)
          erro++;
        c++;
      }
      b++;
    }
    a++;
  }
  return erro == 0;
}
```

```
int main(void) {
  char fn[13];
  sprintf(fn,"a%svlogic.dat",SN);
  in=fopen(fn,"rt");
  if (in==NULL) {
     printf("Não pôde abrir arquivo %s.\n",fn);
  }
  else {
    int i, j, k;
    unsigned long ii;
    FILE *aux;
    word w;
    char c, achou, fim;
    ler(FALSE,and);
    ler(FALSE,or);
    for (j=0; j<NN; j++) {
      fscanf(in,"%s ",w);
      achou = FALSE;
      for (k=0; !achou && k<=M; k++) {
        if (strcmp(v[k],w)==0) {
          achou = TRUE;
          not[j] = k;
        }
      }
      if (!achou) {
        printf("Dado incorreto no arquivo de entrada: %s.\n",w);
        exit(1);
      }
    }
    ler(FALSE,eqv);
    ler(TRUE,impmsk);
    /*
    ler(FALSE,imp2);
    ler(FALSE,imp3);
    ler(FALSE,imp4);
    */
    fclose(in);
    printf("Conjunção:\n");
    printf("   | ");
     for (i=0; i<NN; i++) printf("%s ",v[i]);
     printf("\n---+------------------------------\n");
     escreve(stdout,and);

     printf("Disjunção:\n");
     printf("   | ");
     for (i=0; i<NN; i++) printf("%s ",v[i]);
     printf("\n---+------------------------------\n");
     escreve(stdout,or);

     printf("Equivalência:\n");
     printf("   | ");
     for (i=0; i<NN; i++) printf("%s ",v[i]);
     printf("\n---+------------------------------\n");
```

```
      escreve(stdout,eqv);

      printf("Negação:\n");
      printf("   | ");
      for (i=0; i<NN; i++) printf("%s ",v[i]);
      printf("\n---+-----------------------------\n");
      printf("   | ");
      for (j=0; j<NN; j++) printf("%s ",v[not[j]]);
      printf("\n");

      for (i = 0; i < NN; i++) {
        for (j = 0; j < NN; j++)
            imp[i][j] = impmsk[i][j];
      }

      printf("\nImplicação:\n");
      printf("   | ");
      for (i=0; i<NN; i++) printf("%s ",v[i]);
      printf("\n---+-----------------------------\n");
      escreve(stdout,imp);
      printf("\n");

      if (!mask(imp)) {
        printf("A máscara impossibilita qualquer solução!\n");
      }
      else {
        printf(
"exp(10,%d)-1 = %u (talvez muito grande p/ se representar
assim).\n",
                 nn+1,exp(10,nn+1)-1);

        sprintf(fn,"a%svlogic.aux",SN);
        aux = fopen(fn,"rt");
        if (aux == NULL)
            printf("Não pôde abrir arquivo %s.\n",fn);
        else {
            int x = 0, j = 0;
            time_t t; struct tm *tv;
            printf("Buscando soluções a partir da tentativa #");
            k = 0;
            do {
              fscanf(aux,"%c",&c);
              if (c >= '0' && c <= '9') {
                  printf("%c",c);
                  if (k > 0)
                    imp[hash[k].i][hash[k].j] = c - '0';
                  else
                    imp[hash[k].i][hash[k].j] = c - '0' - 1;
                  k++;
              }
            } while (c >= '0' && c <= '9');
            fclose(aux);
            printf("\n");
```

```
            fim = FALSE;
            while (!fim) {
              time(&t);
              tv = localtime(&t);
              if (tv->tm_min % 5== 0 && tv->tm_sec == 0) {
                  sprintf(fn,"a%svlogic.aux",SN);
                  aux = fopen(fn,"wt");
                  for (k = 0; k <= nn; k++)
               fprintf(aux,"%c",'0'+imp[hash[k].i][hash[k].j]);
             fprintf(aux,"\n");
             fclose(aux);
             while (tv->tm_min % 5 == 0 && tv->tm_sec == 0) {
               time(&t);
               tv = localtime(&t);
                  }
             }

              imp[hash[x].i][hash[x].j]++;
              while (x <= nn && imp[hash[x].i][hash[x].j] > TT) {
                  imp[hash[x].i][hash[x].j] = 0;
                  x++;
                  if (x <= nn) imp[hash[x].i][hash[x].j]++;
              }

              /*
              printf("Verificando a seguinte implicação:\n");
              escreve(imp);
              */
              if (x > nn) fim = TRUE; else {
                  x = 0;
                  if (tabok(imp)) {
                    printf("Solução #%d\n",++j);
                    escreve(stdout,imp);
                    printf("\n\n");
                    sprintf(fn,"a%svlogic.sol",SN);
                    sol = fopen(fn,"at");
                    if (sol == NULL) {
                        printf("Não pode abrir o arquivo de
soluções %s.\n",fn);
                        exit(1);
                    }
                    else {
                        fprintf(sol,"Solução #%d\n",j);
                        escreve(sol,imp);
                        fprintf(sol,"\n\n");
                        fclose(sol);
                    }
                  }
              }
          }
        printf("Total, %d soluções.\n",j);
      }
    }
```

```
  }
  return(1);
}
```

5. THE LOGICAL PROPERTIES

Some properties of this four-valued logic are the following:

Identity: $(A = A)$ holds as the $\leftrightarrow$ truth table ensures this principle. $(A \rightarrow A)$ also holds for any A. The truth tables ensure that $\wedge$, $\vee$, $\leftrightarrow$ are commutative and associative:

$(A \leftrightarrow B) \leftrightarrow (B \leftrightarrow A)$.

$(A \wedge B) \leftrightarrow (B \wedge A)$. $(A \vee B) \leftrightarrow (B \vee A)$.

$(A \leftrightarrow (B \leftrightarrow C)) \leftrightarrow ((A \leftrightarrow B) \leftrightarrow C)$.

$(A \wedge (B \wedge C)) \leftrightarrow ((A \wedge B) \wedge C)$. $(A \vee (B \vee C)) \leftrightarrow ((A \vee B) \vee C)$.

The law of excluded third: $(A \vee \neg A)$. It holds for any Boolean value, but results in *ii* for either *uu* or *ii*. The intuitionistic logic also rejects this law. Thus, there is no unexpected result here, since the logic makes use of four values.

Contradiction (or non-contradiction) principle: $\neg(A \wedge \neg A)$. It holds for Boolean values, but results in *ii* for the extra values.

Furthermore, all the following properties hold for any input values:

Double negation: $A \leftrightarrow (\neg \neg A)$.

Contraposition: $(A \rightarrow B) \leftrightarrow (\neg B \rightarrow \neg A)$.

De Morgan laws: $\neg (A \vee B) \leftrightarrow (\neg A \wedge \neg B)$. $\neg (A \wedge B) \leftrightarrow (\neg A \vee \neg B)$.

Modus ponens: $(A \wedge (A \rightarrow B)) \rightarrow B$.

More properties:

$(A \wedge B) \rightarrow (A \vee B)$.

$A \rightarrow (B \rightarrow A)$.

$(A \rightarrow (B \rightarrow A)) \rightarrow A$.

Commutativity with respect to the implication: $(A \rightarrow (B \rightarrow C)) \rightarrow (B \rightarrow (A \rightarrow C))$.

More properties:

$(C \rightarrow A) \rightarrow ((B \rightarrow C) \rightarrow (B \rightarrow A))$.

$(C \rightarrow A) \rightarrow ((A \rightarrow B) \rightarrow (C \rightarrow B))$.

$(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$.

Distributive and-or: $((A \wedge B) \vee C) \leftrightarrow ((A \vee C) \wedge (B \vee C))$.

Distributive or-and: $((A \vee B) \wedge C) \leftrightarrow ((A \wedge C) \vee (B \wedge C))$.

Transitive: $((A \rightarrow B) \wedge (B \rightarrow C)) \rightarrow (A \rightarrow C)$.

6. CONCLUSIONS

The four-valued logic introduced in this paper is more useful than Belnap's and da Costa's ones.

REFERENCES

[1] Łukasiewicz, Jan (1920) On three valued-logic. in Borkowski, L. (editor) Selected Works by Jan Łukasiewicz, (1970) North–Holland, Amsterdam, pp87-88.

[2] Ackermann, Robert (1967) An introduction to many-valued logics. Routledge & Kegan Paul Ltd, London, and Dover Publications Inc, New York.

[3] da Costa, Newton C. A. et al (1999) Lógica Paraconsistente Aplicada. Atlas, São Paulo, Brazil, 214 pages.

[4] Belnap Jr, Nuel, (1975) A useful four-valued logic. In J. Michael Dunn and George Epstein, editors, Proceedings of the Fifth International Symposium on Multiple-Valued Logic, Modern Uses of Multiple-Valued Logic. Indiana University, D. Reidel Publishing Company, pp8-37.

[5] Ferreira, Ulisses (2000) uu for programming languages. ACM SIGPLAN Notices, 35(8): pp20-30.

[6] Ferreira, Ulisses (2004) A five-valued logic and a system. Journal of Computer Science and Technology, 4(3): pp134-140, October.

[7] Ferreira, Ulisses (2004) Uncertainty and a 7-valued logic. In Pradip Peter Dey, Mohammad N. Amin, and Thomas M. Gatton, editors, Proceedings of The 2nd International Conference on Computer Science and its Applications, National University, San Diego, CA, USA, June, pp170-173.

[8] Ferreira, J. Ulisses (2017) A Note on Gödel's Theorem, International Journal of Computer Science and Information Technology, Vol. 9, No. 2, pp69-76.

AUTHOR

The present author studied as a Master student at the Universidade Federal da Paraíba in Campina Grande, Brazil, and as a postgraduate student in the Department of Computer Science at the University of Edinburgh, and did some further research work at Trinity College in Dublin from 1998 until 2001.

QINCLOUD: AN AGENT-BASED SYSTEM FOR QUERING ENCRYPTED DATA IN CLOUD DATABASES

MASHAEL M. ALSULAMI and AMIN Y. NOAMAN

Department of computer science,
King AbdulAziz University, Saudi Arabia

ABSTRACT

With the rapid growth of technology, cloud computing become more and more popular. Many organizations have been attracted by the variety of services that have been offered by the clouds in form of resources and applications. Database system is one of the most widely used systems in industry. Cloud providers offer database-as-a-service to attract more clients to use their services. However, executing queries against a cloud database is a challenging process since data are stored in encrypted form for security purposes. Cloud database server is responsible of performing the user's query on the encrypted database without any knowledge about the meaning of the requested data to ensure the data confidentiality. Most of the existing methods focus on data confidentially and do not guarantee the performance of their techniques. In this paper, an agent-based system called QinCloud is introduced to execute query efficiently over encrypted data in cloud databases. The proposed system allows users to execute queries on a cloud database server without having any intermediate proxy as a trusted server. Nevertheless, the proposed system is designed to support wide range of queries that are performed completely on the cloud database server side. The proposed system overcomes many issues in existing systems.

KEYWORDS

Cloud Databases, Database as a Service, Security, Query Execution and Data Confidentiality

1. INTRODUCTION

Cloud computing is a promising technology for hosting other important technologies in industry. Many companies and organizations have moved towards cloud computing to grant better performance or to pay less for the infrastructure. Furthermore, users use cloud databases to store and access their data and benefit from other services that provided in the cloud [9].

A cloud database can be defined as a database that can be accessed remotely via the internet from cloud database service provider and application owners pay based on their usage [4]. Customers can plug-in with their own applications to cloud databases. Amazon Aurora, Google Cloud SQL and Microsoft Azure are some examples of cloud database services that are available in the market and offer SQL and NoSql database services. [10]

Natarajan Meghanathan et al. (Eds) : NeCoM, SEAS, CMCA, CSITEC - 2017
pp. 85– 96, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.71207

As a result of the evolution on the area of cloud database services, many issues and challenges have been addressed and been under researches. Cloud database services promises include scalability, availability and elasticity. These promises or some of them are keys for many companies and customers to move toward cloud and reduce the overhead of providing these characteristics by themselves. However, several issues have been raised in order to satisfy these promises. Security, privacy, integrity and data confidentiality are the fundamental concerns in using cloud databases services.

When it comes to store data over cloud databases, clients need to perform certain queries for certain purposes on that cloud database. For security and data confidentiality sake, data are usually stored in encrypted form in cloud databases. The encryption process can be end-to-end encryption or client-side encryption. Many well-defined cloud database vendors provide security guarantee; however, the leak of data could come from the provider itself. Thus, executing queries over encrypted data in a cloud database server is an expensive process in term of performance and security. Some companies focus on the performance of executing their queries with other factors, which can be measured by computing the response time of query execution. On the hands, other companies focus on securing their sensitive data. Many researches have been focused on preserving the data confidentiality when executing encrypted queries over encrypted data in cloud databases. Furthermore, many encryption techniques have been proposed and discussed in the literature to ensure secure and efficient query execution. The ultimate goals for customers to use cloud databases are:

1. Preserving the privacy of their data from any outside (hackers/attacker) leak or inside leak (service provider).

2. Executing queries efficiently in term of computational cost and communication cost.

3. Supporting wide range of queries [5].

The term database-as-a-service refers to a database that is ran on the cloud environment and maintained by a service provider. The structure of cloud databases is complex since data that held in them located in different locations and stored in different data centers. Many companies are moving towards cloud databases [13].

Adopting cloud database has several advantages. First, the main advantage of cloud database is the cost reduction. Users do not have to install specific software to use certain technology. Instead, they can benefit from the capabilities of DBMS without installing one in their local machines. Second, pay-as-you-use feature drives many individuals and companies to use cloud database since they save a lot of money by doing so. Third, companies that are running their databases on the cloud do not have to pay attention to configuration details or maintaining the infrastructure because these consider as provider's responsibilities. Forth, the distribution feature of the cloud allows databases running on it to access variety of information that could be located in different locations. Finally, the most important advantage is the ability of cloud databases to handle big data and perform queries over them [11].

On the other hands, using cloud databases has some disadvantages, which could drive companies to rethink of using cloud environment instead of their local ones. Low security, insured privacy and data confidentiality are the most fundamentals concerns when using services in the cloud [2].

The most challenging issues that face service providers and vendors are related to security. Since database owner does not always control cloud databases but instead a service provider control the cloud DBMS. The focus on recent researches is on preserving the security properties such as confidentiality, integrity and privacy of data in the cloud. Many methods have been proposed to solve issues related to security in the cloud [14].

Data in databases in general should satisfy two aspects: data privacy (confidentiality) and data integrity. Privacy or data confidentiality is the concept of keeping information safe from any disclosure or any unauthorized access. Integrity means that data need to be always valid and correct. More precisely, query integrity means that results returned by database server must be always valid, consistent and correct to ensure that nobody has modified the database except those who are authorized. Furthermore, query results should include all possible records that satisfy the query condition and be the most updated ones [1].

2. RELATED WORK

Many studies have been conducted to discuss the problem of executing queries over encrypted data in cloud databases. Some companies work with sensitive data that are related to their customers like credit card number, annual income and other. Usually, companies try to encrypt these sensitive information if they want to use cloud technology for their business. Literature has discussed this problem from two perspectives: architectural designs and encryption techniques [6].

In architectural perspective, several studies suggest some models and architecture designs to ensure data confidentiality while performing queries against encrypted database in the cloud. In encryption techniques perspective, many research discuss various encryption techniques to ensure the security of data while performing queries.

There are two well-defined systems that are used as solutions to this issue with some limitations in both. These two systems are: CryptDB system and MONMI system. These systems are designed to allow queries to be performed over encrypted data in cloud databases.

CryptDB system has been proposed by Raluca Ada Popa et. al [8]. It mainly works on encrypted databases stored in cloud. The idea of this system based on proxy-based architecture. A trusted intermediate proxy is used between clients and cloud DB server. The proxy is responsible for getting data from database owner as a plain text then encrypts these data and stores them in cloud database. In case of a client publishing a query, the proxy is responsible for getting this query from the client as a plain text then encrypts it and sends it to the cloud database to be executed over encrypted data. Cloud DBMS sends the encrypted results to proxy, which in turn decrypts the results, and sends it back to the client. CryptDB system uses column based encryption algorithm to encrypt both data and query. This encryption algorithm is a powerful one in preserving the confidentiality of the data by encrypting each column in the database using a different key. This technique makes inferring any information from client's queries is a hard task [8].

On the other hands, CryptDB system suffers from three limitations or drawbacks. First, the system suffers from a communication overhead since the existence of proxy as intermediate server between client and DB server, which causes an increase in the communication cost. In this situation, client communicates with the proxy when issuing a query, the proxy communicates

with DB server, then the DB server communicates with the proxy, and finally the proxy communicates back with the client. The second drawback of CryptDB system is the single point of failure, which could make the system hard to scale. The last and critical drawback of the system is that CryptDB doesn't support range and computational queries [7].

MONOMI system is a system that is proposed by [12]. It designed to execute queries over analytical encrypted databases. The idea of MONOMI system is to divide the execution of the query between client side and DB server side. On the other words, a query is analyzed and divided to be executed based on the computations that DB server could handle with no need to decrypt data. Furthermore, MONOMI system executes part of a query in client side and the other part in DB server side to allow more kinds of queries to be executed. One of the biggest contributions of this system is that it could handle 19 out of 22 kinds of queries. However, the drawback of this system includes overhead of splitting queries, poorly use of DBMS capabilities, no security constraints are considered, and finally the cloud DBMS needs to be modified to allow this system to work [7].

As stated previously, many studies have been conducted to solve the issue of executing queries over encrypted data in cloud databases. These solutions focused on the problem from two points of views: architectural designs and encryption techniques.

2.1 Architectural Designs

Many architecture designs have been proposed to find secure and efficient way to execute queries over encrypted data. There are three types of architectures that describe the process of query execution in the cloud databases and preserve the data confidentiality.

The first architecture is a proxy server-based architecture. This type of architecture has been proposed by [12]. The idea of this architecture is to have an intermediate proxy (server) between clients and DB server. The role of this proxy is to perform encryption for both data and queries and to decrypt results coming from DB server. The drawbacks of this kind of architecture include: single point of failure, scalability and consistency.

The second architecture is a proxy server-less architecture with distributed metadata. This type of architecture proposed in [3]. The basic idea of this type is to have a proxy within each client instead of having an intermediate proxy between client and database server. The architecture is based on the elimination of the intermediate proxy to solve the problem of single point of failure. Each client in this architecture has a proxy containing an encryption engine and a maintained copy of metadata. Nevertheless, each client is responsible of encrypting queries sent to cloud database server. Also, each client is responsible of maintaining the metadata after any modification to it. This behavior could result in consistency problem among different copies of metadata especially when multiple clients access the cloud database at the same time.

The third architecture is a proxy server-less architecture with metadata in the cloud. This type of architecture has been proposed by [6]. It designed to solve the inconsistency issue that occurs when concurrent clients access the cloud database at the same time. The idea of this architecture is to store the metadata in the cloud database server. In this case, all clients will access the same copy of metadata and allow for more availability, scalability and elasticity. However, one again this type suffers from single point of failure limitation.

2.2 Encryption techniques

Encryption is a technique that required while using cloud databases to preserve the data confidentiality. Encryption could be performed in the cloud environment from two points of views: End-to-End encryption or Client-side encryption. End to end encryption indicates that cloud database providers are trusted and have keys to decrypt data being transferred or stored on their side. This approach focuses on the security of data while transferring via the Internet. Furthermore, many service providers in cloud environment have depended on this approach of encryption to protect data of their customers. However, this kind of encryption seems to not be applicable with users' needs to protect their sensitive data even from potential threats that could come in the form of curious service providers. Thus, Client-side encryption seems to be the suitable solution for privacy issue. In this approach, data are encrypted by the database owner and stored in the cloud in encrypted format. In this case, the database owner is the only one capable of decryption data or distributing keys over trusted clients. Client-side encryption illustrates the 0-knowledge policy that prevents any sort of unauthorized access to information in the cloud [10].

Generally, encryption techniques can be categorized into two categories: symmetric key encryption (secret key encryption) and asymmetric key encryption (public key encryption). Symmetric key encryption techniques use one master key (secret key) between sender and receiver. Data are encrypted using the secret master key and transferred via the Internet then a receiver decrypts the transferred data using the same master key. The main issue of this technique is the exchanging keys between sender and receiver is not always secured [13].

On the other hands, asymmetric key encryption technique is based on having pair of keys: public and private. Public key is available for anyone wants to send data while private key is used for decryption. The main issue of this technique is the cost of encryption and decryption process [14].

There are various encryption techniques that used by client-side to provide data confidentiality. These techniques used to encrypt data before storing them in the cloud databases and to encrypt queries that need to be performed over encrypted data. Some of these encryption techniques are: *Random encryption schema (RND), Deterministic encryption schema (DET), Homomorphic encryption schema (HOM), Order preserving encryption schema (OPE), Format preserving encryption schema (FPE), Column based encryption schema, and Search encryption schema.*

3. PROPOSED SYSTEM

The proposed system is designed to solve the problem of executing queries over encrypted data in cloud databases efficiently. The main challenges are regarding security in term of data integrity and data confidentiality, and performance in term of communication cost and computational cost. The aims of this system is to reduce the communication cost by using Agent Oriented architecture style and enhance performance by reducing the computational cost while performing computational queries. The motivation of the proposed system is to allow users to perform wide range of queries efficiently over encrypted data in cloud databases.

Assumptions:

- Cloud DB server is not trusted.

- All authenticated clients are trusted.

- Clients are allowed to perform read-only operations such as select.

- DB owner is the only one who is allowed to create tables, alter tables, drop tables, insert, update, and delete records from the cloud database.

The proposed system is shown in figure 1. It is based on agent oriented architecture style. This style has been chosen to benefit from the feature of reducing the communication cost between clients and cloud DBMS and the asynchronous computing feature.

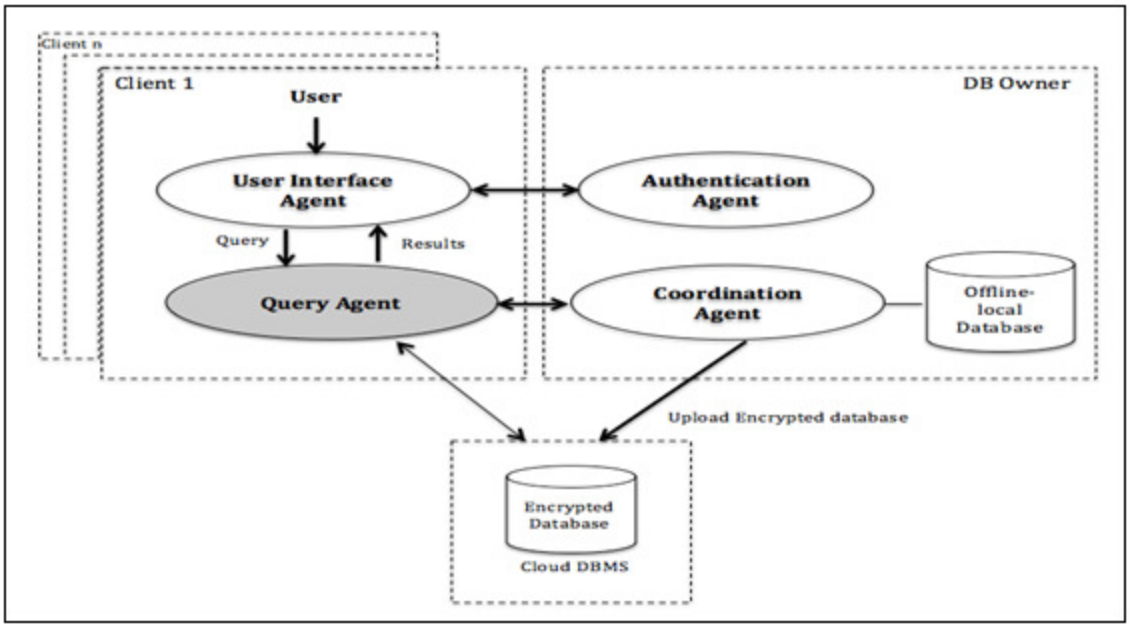


Figure 1: Proposed system

The proposed system consists of the following agents:

User Interface agent:

It is a static agent that is responsible of two main functions. First, it gets the user account information from the user after they log in to the system then determines if he/she is an authenticated user. After a user account has been verified, user can issue a query as a plain text.

Authentication Agent:

It is a static agent. The functionality of this agent is to determine if a user is authenticated or not by checking his/her account information.

Query Agent:

It is a mobile agent. It is responsible of the following functions:

- Encrypting query using the public key that has been published by DB owner.

- Sending the encrypted query to the cloud DBMS to be executed.

- After getting the encrypted results from the cloud DB server, Query agent travels to coordination agent to get the results decrypted.

- After the coordination agent decrypts the results, the results will be forwarded to the query agent.

- It forwards the decrypted result back to the user interface agent.

Coordination Agent:

This agent is responsible of the following functions:

- Publishing public keys for all clients.

- Storing private key in a local offline database.

- Decrypting any encrypted results that are coming from Query agent.

To support wide range of queries, coordination agent in the DB owner side and query agent in the client side use a proposed encryption schema. It consists of two main encryption algorithms: **Algorithm #1** for database creation and **Algorithm #2** for query execution. Database owner encrypts the whole database by using Algorithm #1 before storing it in cloud database server. Each authorized client uses Algorithm #2 to encrypt their queries before sending them to the cloud database server. Each client needs to have an account to use the system. Database owner has two agents: Authentication agent and coordination agent. As stated previously, Authentication agent checks if the logged client is who he/she claims to be. The Coordination agent is the main agent in the database owner side. It is responsible of getting the encrypted results of a query from Query agent then it uses a private key that is stored in a local- offline database to decrypt the results and forward it back to the Query agent. Figure 2 shows a sequence diagram of the proposed system.

Database creation encryption algorithm is shown in figure 3. It is used only once by the database owner to encrypt all tables in a database before storing them in the cloud database server. The main concept of this algorithm is to encrypt each column based on its data type. This mechanism allows applying operations over some columns in a database.

On the other hands, client who issues a query performs query execution algorithm that is shown in figure 4. This algorithm allows each client to encrypt his/her query using the public key that has been published by database owner.

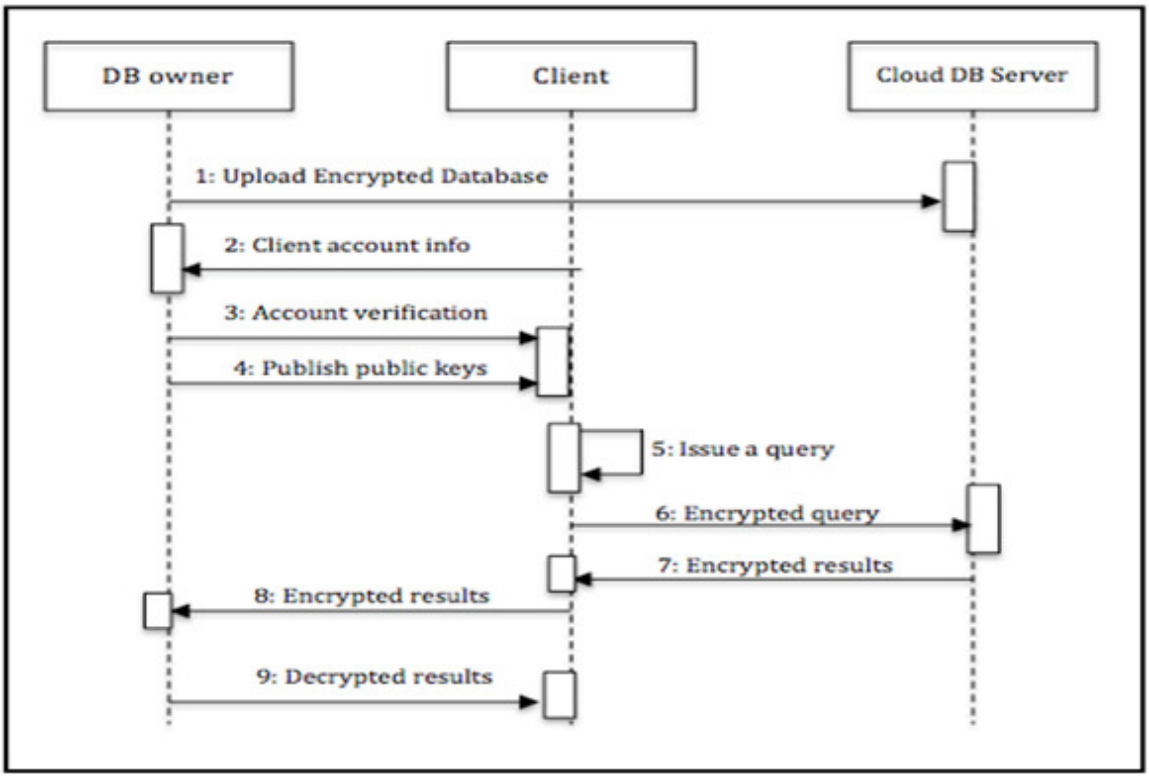


Figure 2: Sequence diagram of QinCloud

Algorithm #1: Database Creation (Setup)

Input: Database schema *S*, $S=\{R_1, R_2, \ldots\ldots, R_n\}$ where *R* is a relation in a database and *n* is the number of the relations in S, public key *k*

Output: Encrypted database schema, $S_E = \{R_{1E}, R_{2E}, \ldots\ldots, R_{nE}\}$ where R_E is the encrypted form of R.

Steps:

1. For every relation R, I created R_E where:
   $R = \{a_1, a_2, \ldots\ldots, a_n\}$ and $R_E = \{Enc_k(a_1), Enc_k(a_2), \ldots\ldots, Enc_k(a_n)\}$
2. Each attribute ***a*** in ***R*** is evaluated based on its data type:

```
if( a is a numerical value) then
  {
   Enc_a= a *432 + 9403;// Applying the concept of OPE
   Insert Enc_a into RE;
  }
 else
  {
   Enc_a = AES (k , a);
   Insert Enc_a into RE;
  }
```

Figure 3: Database creation algorithm

Algorithm #2: Query Execution

Input: Query *Q*, Command file *cmdfile*, operations file *opfile*, and symbol file *symbfile*
Output: Encrypted Query *cipherQuery*

Steps:

1. Client issues a query *Q*.
2. Q is tokenized into n tokens.
3. **For** each token *i* **in** *Q* **do**

```
if ( i ∈ cmdfile OR i ∈ opfile OR i ∈ symbfile ) then
        cipherQuery.write(i);
else if(i is a numeric value) then
        {
          encryptedi = i  *432 + 9403;
          cipherQuery.write(encryptedi);
        }
else
        {
          x = encryptAES(k , i);
          cipherQuery.write(x);
        }

End of for
```

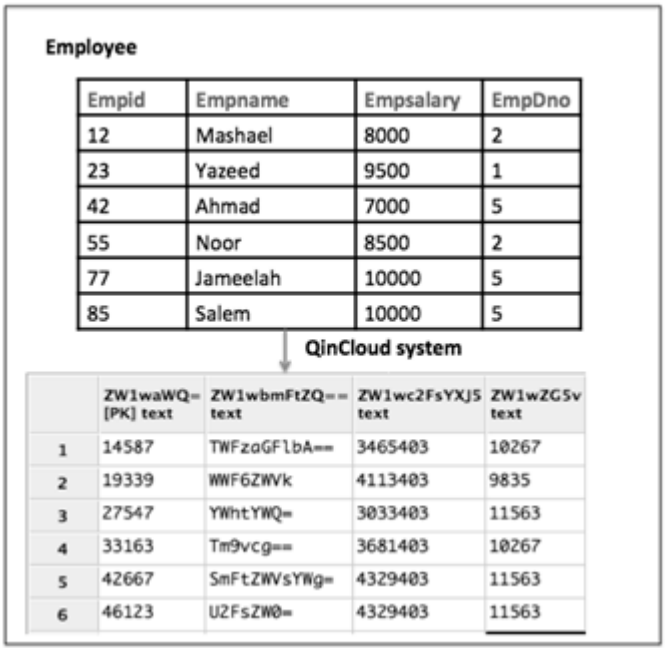
Figure 4: Query Execution algorithm

4. EVALUATION

The proposed architecture is built to enhance the performance of executing queries over encrypted data in cloud databases with preserving the data confidentiality. The benefits of this architecture are as follows:

1. It doesn't require any modifications to the cloud DBMS.
2. All computations are done on the cloud DBMS side.
3. Clients know nothing about how to decrypt the results of the query.
4. All public keys are hidden within the implementations but they are used by the system.
5. The proposed encryption schema uses multiple layer of encryption (AES, OPE and FPE).

To test the proposed system, a simple implementation of QinCloud has been designed using Java as programming language, postgreSQL JDBC Driver as a DBMS, Java cryptographic extension as an encryption framework, and finally Windows builder framework. Figure 5 shows a tested table before encryption and the its encrypted form.



Employee

Empid	Empname	Empsalary	EmpDno
12	Mashael	8000	2
23	Yazeed	9500	1
42	Ahmad	7000	5
55	Noor	8500	2
77	Jameelah	10000	5
85	Salem	10000	5

QinCloud system

	ZW1waWQ= [PK] text	ZW1wbmFtZQ== text	ZW1wc2FsYXJ5 text	ZW1wZG5v text
1	14587	TWFzaGFlbA==	3465403	10267
2	19339	WWF6ZWVk	4113403	9835
3	27547	YWhtYWQ=	3033403	11563
4	33163	Tm9vcg==	3681403	10267
5	42667	SmFtZWVsYWg=	4329403	11563
6	46123	U2FsZW0=	4329403	11563

Figure 5: Sample table

Figure 6 shows a comparison between the execution time of the encrypted query and unencrypted one. It is noticeable that the encrypted query takes a few more msec to be executed than the unencrypted on. However, in some cases such as count operation, the performance is constant since there is no need to deal with the data itself rather counting rows in both tables.

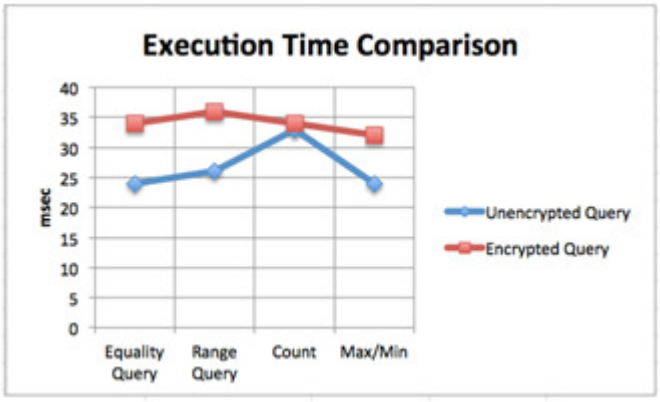


Figure 6: Execution time comparison

5. CONCLUSION AND FUTURE WORK

Recent researches discuss the problem of executing queries over encrypted data from two points of views: data confidentiality and performance. To enhance the performace, three factors need to be taken into considerations: communication cost ,computational cost, and encryption cost. The proposed system has been designed to enhance the performance by reducing the communication cost by using agent oriented architecture. To reduce the computtional cost, a proposed encryption

schema has been used to support equality, count, max, min and range queries and to reduce the response time of executing queries. The future works include:

- Use JADE framework to design the system as agents.
- Enhance the encryption schema to support SUM, AVE and MULT operations.
- Reduce the overhead of the decryption process.
- Use the 22 queries suggested in TPC-H benchmark to test the performance of the proposed system.

REFRENCES

[1] Ghazizadeh, Puya, Ravi Mukkamala, and Stephan Olariu. "Data Integrity Evaluation in Cloud Database-as-a-Service." IEEE Ninth World Congress on Services, 2013. Accessed 26 Nov. 2016.

[2] Ghobadi, Alireza, Roozbeh Karimi, Farnaz Heidari, and Masoud Samadi. "Cloud computing, Reliability and Security Issues." ICACT2014, 2014.

[3] H. Hacigu¨mu¨ s¸, B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over Encrypted Data in the Database-Service- Provider Model,” Proc. ACM SIGMOD Int’l Conf. Management Data, June 2002.

[4] Joshi, V., & Patil, S. SecureDBaaS Architecture For Encrypted Cloud Database. International Journal of Computer Applications, 5(4), 2015

[5] Kumar, R. R., & Hussain, M. Query Execution over Encrypted Database. Second International Conference on Advances in Computing and Communication Engineering, 2015

[6] Luca Ferretti, Michele Colajanni, and MircoMarchetti: Distributed, Concurrent, and Independent Access to Encrypted Cloud Databases. IEEE Transactions On Parallel And Distributed Systems, Vol. 25, No. 2, February 2014.

[7] Munir, Kashif. "Security Model for Cloud Database as a Service (DBaaS)." IEEE, 2015.

[8] Raluca Ada Popa, Catherine M. S. Redeld, Nickolai Zeldovich, Hari Balakrishnan ”CryptDB: Protecting Confidentiality with Encrypted Query Processing”,Twenty-Third ACM Symposium on Operating Systems Prin- ciples,October 2011.

[9] Raluca Ada Popa, Frank H. Li, and Nickolai Zeldovich, ”An Ideal- Security Protocol for Order-Preserving Encoding”, In the Proceedings of 34th IEEE Symposium on Security and Privacy (IEEE S&P/Oakland) , May 2013.

[10] Refaie, R., Ahmed, A., Hamza, N., Al-monem, M., & Hefny5, H. A secure Algorithm for Executing Queries over Encrypted Data. IEEE, 2015.

[11] Sonali, J., & Patil, B. M. Integrating Encrypted Cloud Database Services using Query Processing. International Journal of Computer Applications, 148(12), 2016.

[12] Stephen Tu, M. Frans Kaashoek, Madden.S and Zeldovich.N. " Processing Analytical Queries over Encrypted Data". In Proc. of the 39th International Conference on Very Large Data Bases (VLDB), Riva del Garda, Italy, August 2013.

[13] Syed, Sadia, and M Ussenaiah. "The Rise of Bring Your Own Encryption (BYOE) for Secure Data Storage in Cloud Databases." IEEE Second International Conference on Multimedia Big Data, 2015. Accessed 28 Nov. 2016.

[14] Waghmare, Vivek, Kaveri Gojre, and Akshaya Watpade. "Approach to Enhancing Concurrent and Self-Reliant Access to Cloud Database: A Review." International Conference on Computational Intelligence and Communication Network, 2015. Accessed 29 Nov. 2016.

AUTHOR INDEX