International Journal on Advances in Intelligent Systems

2011 vol. 4 nr. 1&2

The *International Journal on Advances in Intelligent Systems* is Published by IARIA.
ISSN: 1942-2679
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Intelligent Systems
Volume 4, Number 1 & 2, 2011

- Kenji Saito, Keio University, Japan
- Thomas C. Schmidt, University of Applied Sciences – Hamburg, Germany
- Karolj Skala, Rudjer Bokovic Institute - Zagreb, Croatia
- Chieh-yih Wan, Intel Corporation, USA
- Hoo Chong Wei, Motorola Inc, Malaysia

Ubiquitous Systems and Technologies

- Matthias Bohmer, Munster University of Applied Sciences, Germany
- Dominic Greenwood, Whitestein Technologies AG, Switzerland
- Arthur Herzog, Technische Universitat Darmstadt, Germany
- Reinhard Klemm, Avaya Labs Research-Basking Ridge, USA
- Vladimir Stantchev, Berlin Institute of Technology, Germany
- Said Tazi, LAAS-CNRS, Universite Toulouse 1, France

Advanced Computing

- Dumitru Dan Burdescu, University of Craiova, Romania
- Simon G. Fabri, University of Malta – Msida, Malta
- Matthieu Geist, Supelec / ArcelorMittal, France
- Jameleddine Hassine, Cisco Systems, Inc., Canada
- Sascha Opletal, Universitat Stuttgart, Germany
- Flavio Oquendo, European University of Brittany - UBS/VALORIA, France
- Meikel Poess, Oracle, USA
- Said Tazi, LAAS-CNRS, Universite de Toulouse / Universite Toulouse1, France
- Antonios Tsourdos, Cranfield University/Defence Academy of the United Kingdom, UK

Centric Systems and Technologies

- Razvan Andonie, Central Washington University - Ellensburg, USA / Transylvania University of Brasov, Romania
- Kong Cheng, Telcordia Research, USA
- Vitaly Klyuev, University of Aizu, Japan
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Willy Picard, The Poznan University of Economics, Poland
- Roman Y. Shtykh, Waseda University, Japan
- Weilian Su, Naval Postgraduate School - Monterey, USA

GeoInformation and Web Services

- Christophe Claramunt, Naval Academy Research Institute, France
- Wu Chou, Avaya Labs Fellow, AVAYA, USA
- Suzana Dragicevic, Simon Fraser University, Canada
- Dumitru Roman, Semantic Technology Institute Innsbruck, Austria
- Emmanuel Stefanakis, Harokopio University, Greece

Semantic Processing

- Marsal Gavalda, Nexidia Inc.-Atlanta, USA & CUIMPB-Barcelona, Spain
- Christian F. Hempelmann, RiverGlass Inc. - Champaign & Purdue University - West Lafayette, USA
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Massimo Paolucci, DOCOMO Communications Laboratories Europe GmbH – Munich, Germany
- Tassilo Pellegrini, Semantic Web Company, Austria
- Antonio Maria Rinaldi, Universita di Napoli Federico II - Napoli Italy
- Dumitru Roman, University of Innsbruck, Austria
- Umberto Straccia, ISTI – CNR, Italy
- Rene Witte, Concordia University, Canada
- Peter Yeh, Accenture Technology Labs, USA
- Filip Zavoral, Charles University in Prague, Czech Republic

International Journal on Advances in Intelligent Systems
Volume 4, Numbers 1 & 2, 2011

CONTENTS

Randomized Consensus in Wireless Environments **1 - 12**
Bruno Vavala, LaSIGE, University of Lisbon, Portugal
Nuno Neves, LaSIGE, University of Lisbon, Portugal
Henrique Moniz, LaSIGE, University of Lisbon, Portugal
Paulo Verissimo, LaSIGE, University of Lisbon, Portugal

A Quality of Experience management module **13 - 19**
Vlado Menkovski, Eindhoven University of Technology, The Netherlands
Georgios Exarchakos, Eindhoven University of Technology, The Netherlands
Antonio Liotta, Eindhoven University of Technology, The Netherlands
Antonio Cuadra-Sánchez, Parque Tecnologico de Boecillo, Spain

Speech Translation Statistical System for Teaching Environments and Conference Speeches **20 - 30**
Jesus Tomás, Universidad Politecnica de Valencia, Spain
Alejandro Canovas, Universidad Politecnica de Valencia, Spain
Jaime Lloret, Universidad Politecnica de Valencia, Spain
Miguel García, Universidad Politecnica de Valencia, Spain

Identifying Potentially Flawed Items in the Context of Small Sample IRT Analysis **31 - 42**
Panagiotis Fotaris, University of Macedonia, Greece
Theodoros Mastoras, University of Macedonia, Greece
Ioannis Mavridis, University of Macedonia, Greece
Athanasios Manitsaris, University of Macedonia, Greece

A Ubiquitous System for Secure Management of Critical Rescue Operations **43 - 56**
Suleyman Kondakci, Izmir University of Economics, Fac. of Eng. & Computer Sciences, Turkey

Randomized Consensus in Wireless Environments

Bruno Vavala, Nuno Neves, Henrique Moniz, Paulo Veríssimo
Department of Computer Science
LaSIGE, University of Lisbon - Portugal
Email: {vavala, nuno, hmoniz, pjv}@di.fc.ul.pt

***Abstract*—In many emerging wireless scenarios, consensus among nodes represents an important task that must be accomplished in a timely and dependable manner. However, the sharing of the radio medium and the typical communication failures of such environments may seriously hinder this operation. In the paper, we perform a practical evaluation of an existing randomized consensus protocol that is resilient to message collisions and omissions. Then, we provide and analyze an extension to the protocol that adds an extra message exchange phase. In spite of the added time complexity, the experiments confirm that our extension and some other implementation heuristics non-trivially boost the speed to reach consensus. Furthermore, we describe an interesting relationship with a totally different protocol, which explains why the speed-up holds and improves also under particularly bad network conditions. As a consequence, our contribution turns out to be a viable and energy-efficient alternative for critical applications.**

***Keywords*-randomized consensus; wireless networks; message omissions; asynchrony**

I. Introduction

Consensus is a generic abstraction for activity coordination in distributed environments, where nodes propose some local value and then they all reach the same result. In several emerging wireless scenarios, the nodes' ability to perform coordination tasks is of growing interest due to various practical applications. Car platooning in vehicular networks and computing with swarms of agents are just few examples. The first is aimed at grouping cars and making them agree on a common speed, in order to improve highway throughput. The second enables a set of agents to self-coordinate to take advantage of the collective behavior, and its usage spreads from the control of unmanned vehicles to sensor monitoring and actuation. In these settings, since the presence of faults can neither be disregarded nor prevented, it becomes necessary to tolerate them using appropriate protocols.

Fault-tolerant consensus protocols have been matter of research for decades. Proposals have been made for a range of timing models, from synchronous to asynchronous. The asynchronous model allows for the most generic implementations as it avoids any sort of timing assumptions, increasing the resilience to unplanned delays (e.g., because of node or network overloads). However, it is bound by an impossibility result that prevents the deterministic solution of consensus in presence of one faulty node (FLP result) [2]. In any case, even increasing the assumed synchrony is not a panacea, if the communication among nodes is not reliable. Under the *dynamic omission failure model*, a majority of nodes cannot deterministically reach consensus if more than $n-2$ omission faults can occur per communication step, in a synchronous system with n nodes (SW result) [3]. Due to this restrictive result, this model has not been used often, even though it captures well the kind of failures that are observed in wireless ad hoc networks. For instance, dynamic and transient faults caused by environmental conditions, and the temporary disconnection of a node.

Over the years, several extensions to the asynchronous model have been proposed to evade the FLP result: randomization is one of these [4]. Only recently, has this same technique been successfully applied to circumvent the SW impossibility result [5]. Randomization however has always been considered a significant theoretical achievement, but much less a practical one. According to many theoretical studies, randomized consensus protocols are inefficient because of their expected high time and message complexities.

In this paper, we argue and provide evidence that it is possible to build randomized protocols smartly, so that they represent a feasible and practical alternative in wireless environments. Firstly, we analyze the performance of the randomized protocol [5], which has been built for the dynamic omissions failure model. Currently, there is a lack of experience on the implementation and evaluation of protocols for this model. The selected protocol has some nice characteristics, such as ensuring *safety* despite of an unrestricted number of omission faults, and *liveness* when the number of such faults is less than some bound. Secondly, we propose an extension to the protocol, in particular the addition of an extra message exchange phase (a third phase). Results show that even though our new solution slightly worsens the best case scenario, it allows significant improvements in all the other cases, even in presence of bad network conditions. In this last case, the algorithm is sometimes even faster than under normal network conditions. We explain such unexpected result by showing a connection between our protocol and a previously studied other form of consensus. In the end, the reached speed-up translates not only into lower

latencies, but also into less broadcasts, less network usage, thereby enabling our extended protocol to be practical for both time and energy critical environments.

The rest of this paper is organized as follows: in Section II we give a roadmap of the research work in the area; in Section III we describe the system model that underlies our algorithm; in Section IV we briefly detail the k-consensus problem; in Section V we present the extended algorithms, with a detailed explanation of each step; in Section VI we sketch the proof of correctness of the algorithms; in Section VII we supply the results of several experiments, justifying them; in Section VIII we shortly discuss our result and outline some directions for future research.

II. Related work

Consensus plays a pivotal role in distributed computing, particularly when a system needs to cope with accidental faults (e.g., node crashes). The use of randomization in this context arose due to the necessity to circumvent the well know FLP impossibility result [2]. The first seminal works that used this technique were due to Ben-Or [6] and Rabin [7]. Both of them provided protocols to deal with arbitrary node faults, which terminate in an expected exponential number of rounds. Later, Bracha [8] published an optimal protocol to cope with fail-stop processes based on a local coin paradigm. Cachin et al. [9] presented the ABBA protocol for Byzantine agreement, resorting to the shared coin paradigm and asymmetric cryptography operations. A more detailed survey on this class of protocols is available in [4].

To the best of our knowledge, research in randomized protocols has been mostly theoretical, probably because of their exponential complexity. Moniz et al. [10] made a detailed performance comparison between ABBA (for the shared coin class) and Bracha (for the local coin class) protocols. According to their results, the local coin protocol outperformed the shared coin protocol when there is high availability of network bandwidth, which is typical in a LAN. When the bandwidth becomes smaller and the communication delays increase, as in WANs, the cost of cryptographic operations is less important and the shared coin protocols can take advantage of their constant expected running time. The same authors also did an evaluation of a speed agreement algorithm in the context of car platooning, using a stack of intrusion-tolerant protocols [11]. Another interesting evaluation is conducted in [12]. A deterministic algorithm for atomic broadcast is tested under high load, in order to chase (without success) the FLP result.

In this paper, we study a different type of randomized protocol [5], which was designed to work under the dynamic omissions failure model. We also propose a set extensions, including a third phase which, contrarily to intuition, results in a new version of the protocol much more efficient in many practical circumstances. The omission failure model was proposed to show that a reliable asynchronous system is not so different from an unreliable synchronous one, bounded by the SW impossibility result [3]. Indeed, [13] provides a note on their equivalence by simulating one with the other. Other similar and extended impossibility results can be found in [14], [15], by reasoning about knowledge [16], or through a layered analysis of distributed systems [17]. Recently, the work in [5] has been further extended to accommodate Byzantine failures [18], thus closing the long-standing open gap in synchronous systems. However, as well as it happened in any other work that used randomization against a strong adversary to circumvent an impossibility result, the complexity of the proposed protocol in the worst-case scenario is exponential in the number of processes.

III. System Model

The system is composed by a set of n processes with identities $\mathcal{P} = \{p_0, p_1, \ldots, p_{n-1}\}$. It is completely asynchronous in the sense that there is no upper bound on the delays to deliver a message and on the relative speeds of processes. Since our aim is to provide a protocol for wireless networks, the communication medium is shared among processes and every transmission turns out to be a message broadcast. It is assumed that processes are within range of each other. We consider the dynamic communication failure model [3], which captures well the nature of communication problems that may occur, such as dynamic and transient message omissions. It does not make any assumption about the fault patterns, it only presupposes that such faults last for a finite period of time. What may happen is that a process omits a message broadcast or fails to receive a message. The first case might be due to a process that crashes or is temporarily disconnected, and the second case can be related to collisions or environmental noise.

IV. The Consensus Problem

The *k-consensus* problem considers a set of n processes where each process p_i proposes a binary value $v_i \in \{0, 1\}$, and at least $k > \frac{n}{2}$ of them have to decide on a common value proposed by one of the processes. The remaining $n-k$ processes do not necessarily have to decide, but if they do, they are not allowed to decide on a different value. Our problem formulation is designed to accommodate a randomized solution and is formally defined by the properties:

- Validity: If all processes propose the same value v, then any process that decides, decides v.
- Agreement: No two processes decide differently.
- Termination: At least k processes eventually decide with probability 1.

V. The Randomized Consensus Protocol

The paper studies two versions of a consensus protocol (see Algorithm 1). The first corresponds to the protocol of [5], whose authors proved the correctness but did not

provide an experimental evaluation. This protocol was originally built for a synchronous environment, but with the right receive primitive it can also be used in an asynchronous setting. The second version is an extension that incorporates a third phase (darker box in Algorithm 1). We also provide a sketch of its correctness proof, showing how it can be easily adapted from the one of the former algorithm.

A. Overall Execution

Computation proceeds in asynchronous rounds. In each one of these, a process performs a message broadcast of its status (line 6), and after that, it invokes *smart-receive()* to get some messages (line 7). Then, based on the collected messages, it may perform some local computation (lines 9-36). In our context, *smart-receive()* can have different implementations that do not compromise correctness and will be pointed out later.

The state of a process p_i defines the current configuration and it is composed by a set of internal variables: the phase number $\phi_i \geq 0$ (initially set to 0); the proposal $v_i \in \{0, 1\}$ (initially set to the proposal provided as parameter); the decision status $status_i$ (initially is $undecided$).

In the protocol execution, there is a difference between the concepts of *round* and *phase*. A round corresponds to a full iteration of the $while$ loop, starting from line 5 and ending in line 37. A phase is implemented as a process' local variable (ϕ_i), whose value increases monotonically as enough good messages are received, namely when a process is able to update its state (line 32). Processes do not necessarily have the same phase while they execute consensus concurrently. A process may be temporarily outside the communication range of the others, thereby being unable to make progress and to increase the phase number (but continues to execute the loop). However, due to the transitory nature of such situation, as soon as it is able to receive messages, possibly carrying a phase higher than its, it can catch up immediately with the other processes by updating the state (lines 9-13).

In more detail, after broadcasting the state, the process blocks in *smart-receive* to obtain some messages (line 7). This function returns a set $\mathcal{M}$ of messages, all of which are stored in a vector V_i (line 8), if not yet received (this is implied by the union which avoids storing duplicates). More than $\frac{n}{2}$ of these are needed to pass successfully through the main *if* and make progress (line 14). Indeed, after the *if*, no matter what happens next, the process at least updates its state by increasing the phase (line 32).

Now, the process executes instructions in the black box, the extra phase that we call *Pre-Prepare Phase*, because the current phase number is $\phi_i = 0$. This preliminary phase was added to help processes converge rapidly to a decision. Basically, a process sets the local proposal value to a (weak) majority of the proposals carried in the messages (if there is a tie, the process selects value 0). We will show that if processes start with divergent proposals (some of them with 0 and others with 1), this additional step makes most (possibly all) of them choose an equal value before moving to the next phase (line 32).

After receiving enough messages, the process executes the second phase (lines 16-22), that we call *Prepare Phase*, because $\phi_i = 1$. Here, if more than $\frac{n}{2}$ messages carry the same proposal value v, then the process updates the local proposal to this majority value (line 18). Otherwise, the process chooses the default value $\perp \notin \{0, 1\}$ (line 20). One should note that this procedure ensures that if any other process p_k sets $v_k \in \{0, 1\}$, then v_k will be equal to v_i because of the strong majority imposed by $\frac{n}{2}$ (line 17). Before moving to the next round, the process increases the phase number (line 32).

In the third phase, $\phi_i \bmod 3 = 2$, called *Decision Phase*, the process tries to make a decision (lines 22-31). If it receives more than $\frac{n}{2}$ messages with the same value (different from $\perp$), then it is allowed to decide on that value (line 24, 27 and then lines 34-36). Moreover, there is the guarantee that if any other process decides, it will do it for the same value because of the imposed strong majority of more than $\frac{n}{2}$ messages with the same value. If all messages carry as proposal $\perp$, meaning that no process has a preference for a decision value, then the process sets the proposal to an unbiased coin that returns 0 or 1 with equal probabilities (line 29). Eventually, after some rounds, with probability 1 enough processes will get the same coin value that will allow the protocol to make a decision.

B. Receive Operation

A process p_i blocks in the *smart-receive()* operation to collect messages in order to make progress in the protocol execution (either by entering the *if* in line 9 or 14). However, as there may be omission failures, the process does not know how many messages may arrive in a given round, and therefore a timeout mechanism must be implemented inside the *smart-receive()*. When the timeout expires, even if not enough messages have been delivered, the operation is required to return (the reader should note that the use of this timeout does not violate our asynchronous assumption, as it is local and could be implemented with a simple counter). This allows the process to initiate a new round, where the status message is retransmitted (line 6), and then p_i can wait for the reception of a few more messages (line 7). Since this procedure is carried out by all processes, eventually p_i will get sufficient messages to advance.

Consequently, a careful implementation of this operation is fundamental to achieve good performance because it defines the instants when progress can be made and how often messages are broadcast. We adopted and evaluated two strategies in our current implementation.

In the first strategy, the operation waits for the arrival of $\lfloor \frac{n}{2} + 1 \rfloor$ different messages with the same phase of the process (or for the timeout to expire). As soon as this amount

Input: Initial binary proposal value $proposal_i \in \{0, 1\}$
Output: Binary decision value $decision_i \in \{0, 1\}$

1 $\phi_i \leftarrow 0$;
2 $v_i \leftarrow proposal_i$;
3 $status_i \leftarrow undecided$;
4 $V_i \leftarrow \emptyset$;

5 **while** *true* **do**
6 broadcast($m_i := \langle i, \phi_i, v_i, status_i \rangle$);
7 $\mathcal{M} \leftarrow$ **SMART-RECEIVE(*timeout*);**
8 $V_i \leftarrow V_i \cup \mathcal{M}$;

9 **if** $\exists \langle *, \phi, v, status \rangle \in V_i : \phi > \phi_i$ **then** /* Catch-Up Block */
10 $\phi_i \leftarrow \phi$;
11 $v_i \leftarrow v$;
12 $status_i \leftarrow status$;
13 **end**

14 **if** $|\{m \in \{\langle *, \phi_i, *, * \rangle\} \subseteq V_i\}| > \frac{n}{2}$ **then**
if $\phi_i \bmod 3 = 0$ **then**
$v_i \leftarrow \max_{v \in \{0,1\}} |\{m \in \{\langle *, \phi_i, v, * \rangle\} \subseteq V_i\}|$; /* Pre-Prepare Phase $\phi_i \pmod 3 = 0$ */
else
15
16 **if** $\phi_i \bmod 3 = 1$ **then** /* Prepare Phase $\phi_i \pmod 3 = 1$ */
17 **if** $\exists\, v \in \{0, 1\} : |\{m \in \{\langle *, \phi_i, v, * \rangle\} \subseteq V_i\}| > \frac{n}{2}$ **then**
18 $v_i \leftarrow v$;
19 **else**
20 $v_i \leftarrow \perp$;
21 **end**
22 **else** /* Decision Phase $\phi_i \pmod 3 = 2$ */
23 **if** $\exists\, v \in \{0, 1\} : |\{m \in \{\langle *, \phi_i, v, * \rangle\} \subseteq V_i\}| > \frac{n}{2}$ **then**
24 $status_i \leftarrow decided$;
25 **end**
26 **if** $\exists\, v \in \{0, 1\} : |\{m \in \{\langle *, \phi_i, v, * \rangle\} \subseteq V_i\}| \geq 1$ **then**
27 $v_i \leftarrow v$;
28 **else**
29 $v_i \leftarrow \texttt{coin}_i()$;
30 **end**
31
32 $\phi_i \leftarrow \phi_i + 1$;
33 **end**
34 **if** $status_i = decided$ **then**
35 $decision_i \leftarrow v_i$;
36 **end**
37 **end**

Algorithm 1: The 3-Phase Consensus Protocol.

is received, the function immediately returns. Other messages received with that phase (in the next rounds) will be considered old and discarded. To simplify the calculations, we always set the timeout to $10ms$. We call this option *Immediate Progress* (ip).

In the second strategy, which we call *No Immediate Progress* (no-ip), the operation waits for all messages that may arrive in a timeout period, possibly much more than $\frac{n}{2}$. After the timeout expires, the operation returns this whole set of messages. Here the timeout value is more critical as we want to wait for a set of messages, such that $\lfloor \frac{n}{2} + 1 \rfloor \leq |\mathcal{M}| \leq n$, but without wasting too much time if messages get lost. In our experimental environment, it was found that $timeout = n \times 1.25\ ms$ provides good results.

It is clear that having such static timeouts is not the best option, and things like network load should be taken into consideration. Therefore it would be useful to devise mechanisms to adapt the timeouts to the current network

conditions. However, this topic falls outside the scope of this paper and will be matter of future investigation. For instance, the work of [19] could be used as a starting point, since it is proposed a pro-active method for checking the network quality of service and estimating the timeout accordingly.

C. Protocol Termination

It is worth to notice that the protocol guarantees termination, in the sense that consensus is eventually reached, but it does not (and cannot) stop the execution. Here the problem is that when a process decides, it must keep on broadcasting its status to let the others decide. Furthermore, even if it learns that everyone has decided, the process is not allowed to terminate because someone else may have not received its decision status and would never terminate. Again, even if everyone else had received it, no process would be sure that any other process knows that they all reached consensus. In summary, this is the classic problem that arises in any unreliable message-passing systems, and which is impossible to solve through a finite number of message exchanges, without further assumptions. More precisely, as it has been formally proved in [17], in such systems the processes are unable to attain the required level of knowledge (i.e. common knowledge) about consensus termination, to stop sending messages safely.

One solution to this problem is to use a centralized node that is informed when any process decides, and then tells everyone that they should terminate when enough processes have finished. This approach has several limitations, such as how to address the failure of the centralized node. In our implementation, we resorted to a pragmatic solution that is based on giving *sufficient* time after decision for the processes to terminate. In more detail: a process continues to broadcast up to a certain time after decision (1 second); then, the process is allowed only to receive messages, to empty its network buffer; if no message is received for some interval (2 seconds), then the process terminates the consensus. Clearly it is a far from being a perfect solution, but it worked very well in our experimental setting. Additionally, it did not impact the evaluations because the utilized metric was the latency (defined in the next section). The same applies to the case when we count the number of rounds to reach consensus, because counting is stopped by that time.

VI. Correctness

As previously underlined, the algorithm is an extension of the one proposed in [5], hence its correctness follows almost immediately from it. For this reason, we only sketch the proofs for the sake of completeness.

Firstly, we give some straightforward hints on why the asynchronous model does not represent a problem and why the usage of timeouts does not contrast with it. By simply looking at the algorithm, it is clear that there is no time assumption, but the solely presence of a timeout. The computation of each process is thus message-driven, and not directed by clock-synchronization, in the sense that they need either a single message carrying a larger phase or more than $\frac{n}{2}$ of them with the same phase to set some value and go ahead. The algorithm however does not have any mechanism to detect failures (message dropping). Therefore, a process cannot determine if a message has not been delivered because it travels too slowly or because it was dropped. The use of a timeout is sufficient to solve this issue, by making the sender broadcast the same message infinitely often, if it is unable to hear anything from (enough of) the other processes. The timeout does not violate any asynchrony assumption because it does not necessitate any synchronization with global time. As stated in [20], it can be implemented locally, following just the process' local clock, by a simple (instruction) counter. The only drawback of this implementation is that the rebroadcast operation is static and timeout-driven, so in practice it could overload the network. However, there exist ongoing works, aimed at overcoming such inefficiency by dynamically linking the timeout to the network conditions (see [19]).

For what concerns the correctness proof, we divide it into two branches: *safety* and *liveness*. The safety part is straightforward because the algorithm has been designed to be always safe no matter what the time requirements, the number of omissions and their patterns are. So, starting from an initial safe state, either it does not make any progress (it does not receive enough messages for the computation to evolve to higher phases), thus remaining in a safe state, or it does, but without ever executing a step that might bring two processes to divergent decisions, thereby preserving safety.

The liveness part is a little more complicated because it must deal directly with the main feature that characterizes our model: the presence of dynamic and transient message omissions. It must be guaranteed that the processes receive enough messages to make progress despite omissions. The transient aspect ensures that this requirement is met: processes may be unable to communicate properly for an arbitrarily long period of time, during which the best they can do is to avoid taking wrong decisions (perhaps stubbornly probing the channel's status), but eventually the network conditions get better to allow them to complete the task. The dynamic aspect is instead aimed at generalizing the particular pattern that the omissions can have (e.g., distributed among the processes or concentrated to few ones). The bound on omissions to ensure progress, no matter their pattern, is $f \leq \lceil \frac{n}{2} \rceil (n-k) + k - 2$ (if all the processes must decide, then $k = n$ and $f \leq n-2$). Since our algorithm is an extension, such bound holds also in our case, but we refer to [5] for a more complete and formal explanation of liveness.

A. Safety

Theorem 1 (Validity)**.** *If all processes propose the same value v, then any process that decides, decides v.*

Proof Sketch: Each process has its proposal initialized to the same value v. After the (perhaps repeated) message exchange, as all the messages contain the same value v, processes get the same (weak) majority in the pre-prepare phase, thus setting their local proposal to the same majority value (value set in the box). Then, they wait to receive more than $\frac{n}{2}$ messages for the prepare phase (line 14), where they get the same strong majority thus setting the same value (line 18). Subsequently, they again wait for more than $\frac{n}{2}$ messages, all of them with the same value v, and they decide on v (lines 24, 27 and 35). ■

Theorem 2 (Agreement)**.** *No two processes decide differently.*

Proof Sketch: By contradiction, suppose they take different decisions, respectively on v_0 and v_1. This cannot happen when a process catches up with another one that is executing a later phase, by copying its state (line 9). So, there are two cases.

1) They decide in the same phase. Then it must be that they received a strong majority of messages (line 23) for the decided values with the same phase number ϕ. So, it must be that at least one process broadcast two messages with two different proposals. As all processes follow the algorithm faithfully, this is clearly a contradiction.

2) They decide in different phases. In this case, one should notice that in the prepare phase each process sets its proposal either to a value contained in a strong majority of the received messages (in that phase), or to a default value $\perp$. So, in the decision phase, the only values allowed are the default $\perp$ and a single binary value, either 0 or 1 (but not both of them). The first process that decides on a value, receives a strong majority for that value in the decision phase. Therefore, all the other processes in the same phase receive at least one message with that value. As a consequence, any other process can either decide that value in that phase, or can just set its proposal to that value (line 27, without tossing a coin). The decision of a process thus *locks* a particular value, which is the only one that can be proposed and decided by all processes in subsequent phases. For this reason, the process supposed to take a different decision is a clear contradiction.

■

B. Liveness

In the next, we say that a process can *make progress*, each time it is able to execute either line 10 or line 32, a new phase. In the first case, the process makes progress when it receives at least one message carrying a phase number higher than its, by copying all the information to its state. In the second, it receives a set of messages from more than half of the processes carrying the same phase number of its.

Lemma 1. *If a process has phase ϕ, then there are more than $\frac{n}{2}$ processes with phase at least $\phi - 1$.*

Proof Sketch: Consider the first process that sets its phase to value ϕ. Then it must have received more than $\frac{n}{2}$ messages with phase $\phi - 1$. Hence there are more than $\frac{n}{2}$ processes with phase at least $\phi - 1$ ■

Lemma 2. *If a process has phase ϕ, then there will be at least $k > \frac{n}{2}$ processes with phase at least $\phi - 1$.*

Proof Sketch: By lemma 1 more than $\frac{n}{2}$ processes (call their set A) have phase at least $\phi - 1$, we have to show that eventually also $k - |A|$ other processes reach that phase. Suppose by contradiction that they cannot. Then all messages from A to all the other processes must be dropped. In this setting, we can rewrite the number of processes as composed by the following groups: $n = |A| + (k - |A|) + (n - k)$. Therefore the number of omissions should be

$$|A|(n-k)+|A|(k-|A|) > \frac{n}{2}(n-k)+|A|(k-|A|) \quad (1)$$

$$\text{for } k-|A|,|A|>0 \qquad \geq \frac{n}{2}(n-k)+(|A|+k-|A|-1) \quad (2)$$

$$= \frac{n}{2}(n-k)+k-1 \quad (3)$$

$$> f \quad (4)$$

As the number of omissions exceeds the bound given above, this is a contradiction, so eventually k processes have phase at least $\phi - 1$. ■

Lemma 3. *If at least $k > \frac{n}{2}$ processes have phase at least ϕ, then some process in the system must eventually increase its phase value.*

Proof Sketch: Our system can be described by three sets. A is the set of processes with phase greater than ϕ, B is the set of those with phase ϕ, and C is the set of those with phase smaller than ϕ. The cardinality of each set is a, b, c respectively, such that: $a + b = k$. By contradiction, suppose that no process ever increases its phase. Therefore: 1) no message from A must reach either B or C; 2) no message from B must reach C; 3) if there are more than $\frac{n}{2}$ processes in either A or B, they must be prevented from exchanging enough messages among themselves. For what concerns 1) and 2), the number of omissions that must be imposed is at least $ab + ac + bc = c(a + b) + ab = (n - k)k + ab > \frac{n}{2}(n - k) + ab$. Now, let us compute a bound for ab. If either A or B is empty, then $ab = 0$, but the other set contains $k > \frac{n}{2}$ processes that, by exchanging messages among themselves can still make progress. So, in this case, at least k omissions (one for each process when k is minimal) are necessary. If otherwise none of them is

empty, then $a, b > 0$ so $ab > a+b-2 = k-2$. In either case, the total number of omissions required exceeds the bound f given above. This is a contradiction, so eventually some process increases its phase value. ■

Lemma 4 (Progress)**.** *If ϕ is the highest phase reached in the system, then eventually some process sets its phase to $\phi + 1$.*

Proof Sketch: As there is at least one process with phase ϕ, according to lemma 2, there will eventually be $k > \frac{n}{2}$ processes with phase at least $\phi - 1$. Our system can thus be described by three sets. A is the set of processes with phase ϕ, B is the set of those with phase $\phi - 1$, and C is the rest of the system. The cardinality of each set is a, b, c respectively, such that: $a > 0$, $a+b = k$. According to lemma 3, since there are k processes with phase at least $\phi - 1$, some process must increase its phase value. In the worst-case, by applying the lemma repeatedly, we have that C gets empty, so processes in B must necessarily start increasing their phase value. When B gets too small, $|B| < k$, we can apply lemma 3 to processes in A, that all have phase ϕ. Again, in the worst-case, we have to wait for B to become empty. Anyway, as soon as all processes have phase ϕ, at least one of them must eventually set its phase value to $\phi+1$. ■

This lemma is very important because it proves that, as soon as the bound on the omissions is respected, the processes are guaranteed to be able to make progress, thereby reaching an arbitrarily high phase. However, it is important to recall from the impossibility results in [2], [3] that making progress is not a sufficient condition for the processes to reach consensus. Indeed, they demonstrate that processes can make progress infinitely often, by touring across bivalent system states, in which no safe decision can ever be taken. What makes consensus possible (i.e. terminate) is randomization, and the termination property is sketched in the following theorem.

Theorem 3 (Termination)**.** *At least k processes eventually decide with probability 1.*

Proof Sketch: By contradiction, suppose that no process ever takes a decision, but everyone can make progress (see lemma 4) infinitely often. In the decision phase, each process can either set its proposal to a binary value present in some message or to a random one, by tossing a coin. By means of arguments provided in previous proofs in relation to the quorum of messages received, even if no decision takes place, it is only one (if any) the binary value that each process is allowed to set in the decision phase. So, we can partition the processes into two groups: (A) processes that set the same value, (B) processes that toss a coin. If B is empty, then the processes set the same value v. Since v becomes the only value proposed, it is the only one that can be decided in the next phases. By lemma 2, 3 and 4, there are at least k processes able to make progress, so at least k that reach a decision. If B is not empty, then there is a non-zero probability that these processes get the same binary value of the processes in A, after tossing a coin, with $p = 2^{-|B|}$. Since processes make progress infinitely often, if (by assumption) they do not decide, they eventually have to toss a coin infinitely often. However, the probability of not setting the same value is asymptotically $\lim_{phase\to\infty}(1-p)^{phase} = 0$. Hence eventually at least k processes decide. ■

VII. Performance Evaluation

In this section we evaluate the two versions of the protocol, stressing how little modifications may produce so different and perhaps unexpected results.

A. Testbed and Implementation

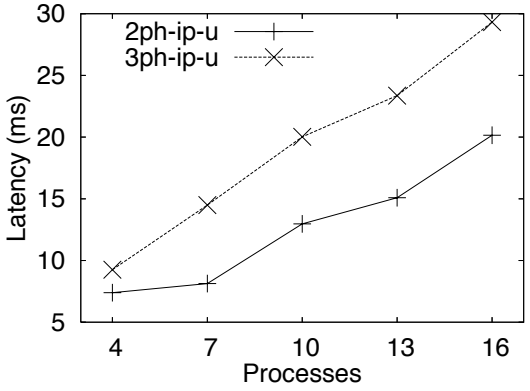
The experiments were carried out in the Emulab testbed [21]. All the nodes, located few meters away from each other, were equipped with a 600 MHz Pentium III processor, 256 MB of RAM, and the 802.11g D-Link DWL-AG530 WLAN interface card. The nodes run Fedora Core 4 Linux, with the 2.6.18.6 kernel version.

The protocol versions were implemented in C, and they used UDP for lower level communication to take advantage of the broadcast medium. During the evaluations, we observed some omission failures, which in part were due to collisions caused by our own traffic (the testbed is shared with other researchers, and therefore, their experiments also created collisions). Nevertheless, in order to evaluate the protocols in presence of omissions, we decided to develop a layer to emulate network losses. Such layer represents our *adversary*. The adversary is able to delete a complete message broadcast and to prevent specific processes from successfully receiving a message. The decision of which messages should be discarded is made randomly. The user can specify different percentages of messages to be removed at the source and at the destination.

The main metric we use to compare the different approaches is *latency*. The *latency at process* p_i is defined as the interval between the moment the protocol starts and the instant when the local decision is reached. The *latency of an experiment* is the average value of the latencies of all processes. In the graphs, we present average values of several experiments to increase the confidence on the results.

B. Experimental Results

1) Uniform proposal distribution: We evaluate the best scenario in which all processes start with the same (uniform) proposal value. As by the consensus specification (see Section IV), this value is the only one that can be decided. The two protocol versions reach a decision after 2 phases in the original version, and after 3 phases in our version. In



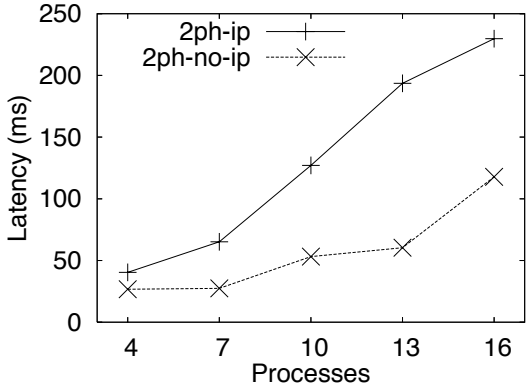


Figure 1: Latency with uniform proposal values.

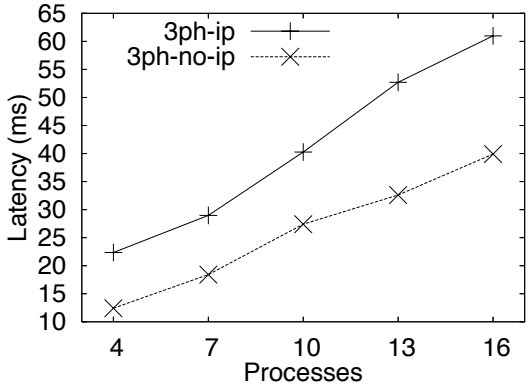


Figure 2: Evaluation of the *smart-receive()* strategies.

the experiments, we observed that processes always receive enough messages to make progress in every round, allowing termination to be reached in these minimum number of phases (and with the minimum number of rounds). *Fig. 1* shows clearly the increase in delay that the new phase adds. The slopes of the lines reflect the difference between executions that require respectively $2n$ and $3n$ broadcasts.

2) Divergent proposal distribution: In the rest of the experiments, we will always consider a divergent initial proposal distribution among processes (half of them propose 0 while the others propose 1). Before highlighting the difference in execution speeds between the protocol versions, we supply some results related to the introduction of the *no immediate progress* (*no-ip*) concept in the *smart-receive()* operation. *Fig. 2* shows that the concept is important in enhancing performance (smaller latency is better).

In a context where initial proposals differ from each other, even though the *ip* strategy allows a process to make progress as soon as possible, this will (very probably) lead to nowhere. In order to make progress towards a decision, a process needs more than $\frac{n}{2}$ messages with the *same value* to avoid default values (line 20) and random choices (line 29). Therefore, with *ip*, the process needs to be lucky with the order of the messages it receives to converge to a decision. On the other hand, with *no-ip*, waiting for more messages allows one to exploit the power of random choice to bias the proposals toward a particular value.

Let us clarify this issue with an example in a fault-free scenario. Suppose that n is even, $n \geq 4$ and processes have *divergent* proposals. Since there is no strong majority, all of them will be compelled to toss a coin (line 29, with or without *immediate progress*) after running the first two or three phases, depending on the protocol version. Now, let $V[i],\ 0 \leq i \leq n-1$ be a vector with the new proposals and consider the event $\mathcal{E} := \{\sum_{i=0}^{n-1} V[i]\ equals\ \sum_{i=0}^{n-1} 1 - V[i]\}$. After one coin tossing we have that $P(\neg\mathcal{E}) > P(\mathcal{E})$, hence: (1) proposals tend to be equally distributed between the values, but (2) the chances to get a strong majority overtake the others (more and more as n increases). Therefore, with *no-ip* processes will probably progress toward consensus because of (2), while with *ip* it would be very difficult for processes to notice a strong majority (among only $\lfloor \frac{n}{2} + 1 \rfloor$ messages received) in the first phase because of (1).

3) The addition of the third phase: The addition of an extra phase has potentially several drawbacks. Namely, it increases the network usage because consensus can only be reached in phases that are multiple of 3 rather than 2. The results in *Fig. 3* show instead that the 3-phase protocol overwhelms the 2-phase one in both strategies.

Actually, the third phase represents a smart algorithm design that at first might escape our (theoretical) analysis. Without this phase, a process needs to receive more than $\frac{n}{2}$ messages with the same value to set its proposal (line 18)

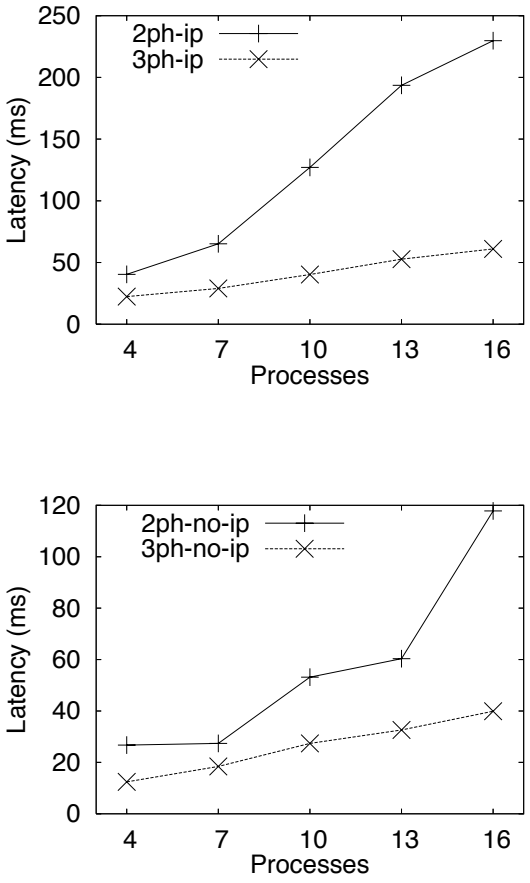


Figure 3: The impact of the third phase.

and eventually decide (line 24). However, according to all possible configurations, it is unlikely that such strong majority occurs, even more with a divergent proposal distribution. This situation forces a lot of processes to set default (line 20) and random values (line 29), preventing them from deciding and making them start all over again. The problem here is that messages end up being discarded too easily and with little consideration, without exploring as much as possible the information being transmitted.

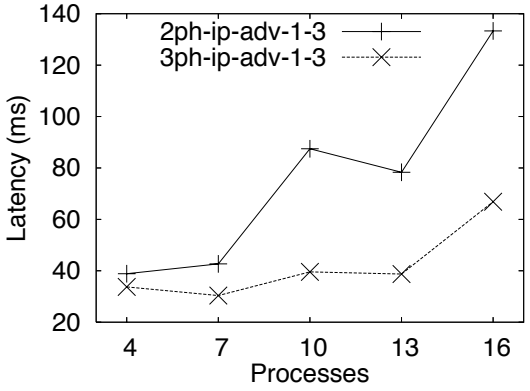
This is what the third phase does – it uses better the data carried in the messages, to allow progress towards a decision much sooner. In the original protocol, the first *prepare phase* was intended to make the processes set the same value (lines 16-22), while the second *decision phase* was to let them learn and decide on this value (lines 22-31). Our third phase is executed just before these two (resulting in an extra *pre-prepare phase*), and it relaxes the need of a strong majority. Although more than $\frac{n}{2}$ messages are always needed to make progress, here there is no other requirement: a process sets its proposal to the majority value received (and in case of a tie selects value 0). Clearly, it is not true that the processes will always set their proposal to the same value, since this depends on the ordering of message arrival. Nevertheless, since processes share the same wireless medium, reordering is expected to be small and all processes should receive approximately the same set of messages. Hence, if there is a value that is predominant in the set, this enables lots of processes (if not all of them, in expectation) to pre-set that value. Therefore, this has a positive influence on the subsequent phases, improving convergence. Indeed, our experiments show that most of the times consensus is reached in few rounds.

4) Performance under failure scenarios: The three-phase version outperforms the original protocol in presence of an adversary that creates various sorts of omission failures. Such adversary may make a process discard an entire broadcast (causing n receive events to be lost), with probability $\mathcal{P}_s$, or may cause a single message omission at a specific receiver, with probability $\mathcal{P}_r$. The probabilities that were used in the experiments are: $\mathcal{P}_s = 0.1$ and $\mathcal{P}_r = 0.3$, abbreviated as adversary *adv-1-3*; and $\mathcal{P}'_s = 0.3$ and $\mathcal{P}'_r = 0.6$, abbreviated as adversary *adv-3-6*. According to these probability, defining the event $\mathcal{E} := \{\text{message reaches destination}\}$, $\mathcal{P}[\mathcal{E}] = (1 - \mathcal{P}_s)(1 - \mathcal{P}_r) = 0.63$ and $\mathcal{P}'[\mathcal{E}] = (1 - \mathcal{P}'_s)(1 - \mathcal{P}'_r) = 0.28$. Adding up to these failures, we should not forget that there is already some packet loss due to wireless links and the (unreliable) UDP protocol. The experiments show that these omissions are almost null with a few processes but they can increase up to 30% of the traffic with 16 nodes. The results with such severe conditions are visible in *Fig. 4* and *Fig. 5*.

A curious latency trend is visible in the figures, where often it decreases when the number of nodes is raised from 4 to 7 and from 10 to 13. This is due to the majority threshold of $\lfloor \frac{n}{2} + 1 \rfloor$. While the number of processes is always increased by 3 units, the majority threshold increases less regularly. More specifically, it increases in 1 unit in those previous cases, and in 2 units in the others. Furthermore, the ratio between majority and number of processes turns out to decrease in those cases and to increase in the others.

It is also possible to notice that the 4 nodes configuration in the *2ph-ip-adv-3-6* experiment has particularly high latency. This setting is very sensitive to packet dropping, and therefore, nodes need to perform more broadcasts, most of which are duplicates (retransmissions of their status).

It is noteworthy to consider something unexpected – the adversary *adv-1-3* can make the 2-phase protocol with *immediate progress* go faster (compare *Fig. 3* with *Fig. 4*). This is not (only) due to the entire broadcasts that are discarded, thereby reducing wireless medium contention, and therefore improving message delivery. According to the test data retrieved in those cases: less phases and less broadcasts are needed to reach consensus; less status information



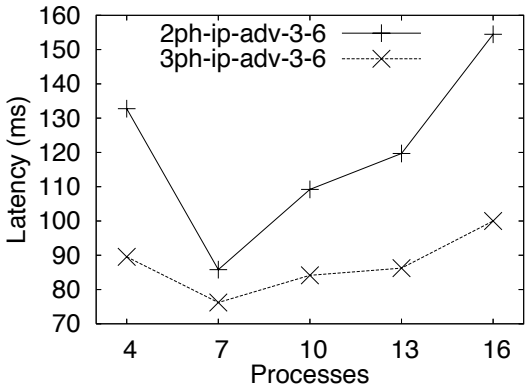
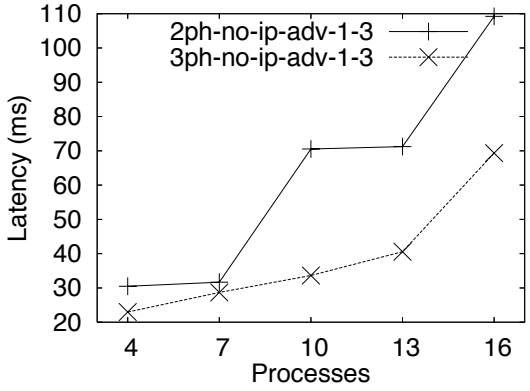


Figure 4: *Immediate Progress* in presence of an adversary.



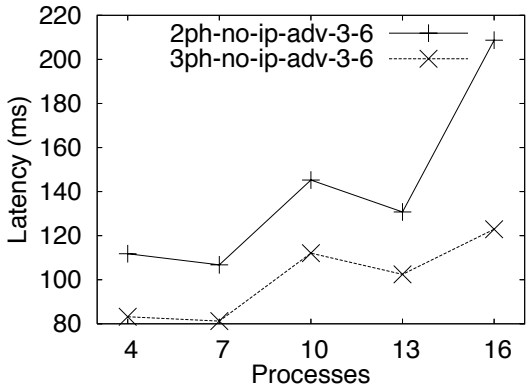


Figure 5: *No Immediate Progress* in presence of an adversary.

is sent and received for every phase; more phase jumps occur due to the arrival of messages with higher phase; few processes reach decision by themselves. Therefore, what happens is that, in every phase, fewer processes are able to make progress and, as soon as they update their status and broadcast it, other processes that were left behind can immediately catch up with them, learning by copying their status (lines 9-13). It is this copy that boosts the decision procedure because processes use it as a sort of *pre-prepare phase*, avoiding setting default and random values.

In any case, though more onerous in theory, our phase extension enables the processes to complete the task even more quickly in all the cases and, above all, provides results much more stable, less subject to relevant changes due to the lower number of coin flips induced.

C. An interesting relationship

In [22], a deterministic version of a randomized wait-free shared-memory consensus algorithm is proposed and analyzed. The intention of the paper is to show that randomization helps even if it is not present inside the algorithm but rather in the environment itself. More precisely, the system is not asynchronous but each process takes a random time (characterized by a random variable X, different for each process) to complete each operation. The result holds also if these random variables do not have a finite expectation.

As for every binary consensus algorithm, each process prefers either 0 or 1. The decision is settled by means of a race. There are two arrays (each one representing a binary proposal 0 or 1) of atomic read/write registers, accessible to all processes, starting from their first position. The computation of each process proceeds in rounds, whose number is used as offset to address the correct register position in the array. At every round, a process computes the location of the registers in the two arrays. If one of these registers has a bit raised, the process updates its proposal to the value that the array represents. Otherwise it raises a bit

TABLE I: AVERAGE NUMBER OF ROUNDS IN WHICH 16 PROCESSES REACH CONSENSUS IN A 802.11B NETWORK AND CONFIDENCE LEVEL OF 95%.

Group Size = 16	Average Termination Round ± Confidence Interval
2ph-ip	17.80 ± 1.48
2ph-no-ip	8.99 ± 0.49
2ph-ip-adv-1-3	10.23 ± 0.63
2ph-ip-adv-3-6	7.21 ± 0.35
2ph-no-ip-adv-1-3	6.60 ± 0.23
2ph-no-ip-adv-3-6	6.00 ± 0.15
3ph-ip	6.85 ± 0.28
3ph-no-ip	4.60 ± 0.16
3ph-ip-adv-1-3	5.50 ± 0.23
3ph-ip-adv-3-6	4.90 ± 0.21
3ph-no-ip-adv-1-3	4.60 ± 0.18
3ph-no-ip-adv-3-6	4.30 ± 0.13

in the register of the array corresponding to its proposal. At this point, of the two registers localized at the same position p (indicated by the round/offset) in each array, only one has a bit raised, and the process has its proposal set to the array to which this register belongs. Then the process checks if the register in position $p - 1$ of the array representing the other proposal is not set. If it is not, it can safely infer that its computation has fallen ahead the one of the other processes, and can safely decide on its own proposal. Otherwise it increases the round number and repeats the procedure.

This algorithm obviously does not work if processes run in lock-step (synchronized) execution, because none of them may eventually be able to impose its proposal to the others. However, by leveraging on the random execution time of the operations, it is likely that the computation of the processes gets dispersed in time and space. In particular, as proven in [22], there is one process (the leader) that emerges faster than the others, decides on its proposal, and makes the others (learners) set and decide on the same value.

Our model and algorithm are clearly very different from the ones just presented. Yet we can provide evidence of a very simple relationship. The crucial part in our algorithm is due to the lines 9-13, where a process, once it has received a message carrying a later phase, just copies (learns) the sender's state and possibly also its decision. Therefore, if a particular process is allowed to get/send messages and to make progress immediately, it would (unconsciously) rise to a leader position, dictating its proposal to the other processes. In our practical situation, the more processes start a phase with the same proposal, the more likely is for them to get good quorums and decide immediately.

This relationship is evidenced in the evaluation section, particularly in presence of an adversary. By dropping messages, the adversary indeed worsen the network situation, but makes more probable that few lucky processes come up being faster. By looking at the latency results, it can be seen that the 2-phase algorithm turns out to be quicker under bad network conditions. This may not be noticed for the 3-phase version, because of its added time complexity and the need to rebroadcast messages. Nevertheless, it is observable by dealing with a different notion of speed, namely the number of rounds needed to reach consensus. In this case, it is clear (see Table I) that this relationship indeed exists, and that message dropping can be leveraged either to make a leader rise quickly, or at least to reduce the number of processes involved in the decision process, and so its complexity.

VIII. CONCLUSIONS

The paper provides evidence of the practicality of randomized consensus protocols. Even though such protocols have a high theoretical complexity, the experiments show that performance can be significant even with high failure rates. This is particularly true in realistic scenarios where a small number of processes is being used and if algorithms are carefully engineered to be as efficient as possible. A protocol previously presented in the literature, which aimed at solving consensus in wireless environments, has been analyzed and extended. We show how these extensions can exploit the available information about the state of the system in order to increase performance. The result is a protocol that is much faster in terms of latency, more thrifty in terms of messages and hence of network usage, properties that make if effective for time and energy constrained environments. Also, we discuss an interesting connection with a different protocol, which was formally proved to be fast. In the future we plan to carry out extensive comparisons with other consensus protocols under the unreliable network communication model.

ACKNOWLEDGMENTS

This work was partially supported by the EC through project FP7-257475 (MASSIF), and by the FCT through the Multiannual and the CMU-Portugal Programmes, and the project PTDC/EIA-EIA/113729/2009 (SITAN).

REFERENCES

[1] Bruno Vavala, Nuno Neves, Henrique Moniz, and Paulo Veríssimo. Randomized consensus in wireless environments: a case where more is better. In *Proceedings of the 3rd International Conference on Dependability (DEPEND)*, pages 7–12, 2010.

[2] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374–382, 1985.

[3] Nicola Santoro and Peter Widmeyer. Time is not a healer. In *Proceedings of the 6th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 304–313, 1989.

[4] James Aspnes. Randomized protocols for asynchronous consensus. *Distributed Computing*, 16(2-3):165–175, 2003.

[5] Henrique Moniz, Nuno Ferreira Neves, Miguel Correia, and Paulo Veríssimo. Randomization can be a healer: consensus with dynamic omission failures. In *Proceedings of the 23rd International Conference on Distributed Computing (DISC)*, pages 63–77, 2009.

[6] M. Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols. In *Proceedings of the 2nd Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 27–30, 1983.

[7] M. O. Rabin. Randomized Byzantine generals. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 403–409, 1983.

[8] G. Bracha. An asynchronous $\lfloor (n-1)/3 \rfloor$-resilient consensus protocol. In *Proceedings of the 3rd Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 154–162, 1984.

[9] C. Cachin, K. Kursawe, and V. Shoup. Random oracles in Constantinople: Practical asynchronous Byzantine agreement using cryptography. *Journal of Cryptology*, 18(3):219–246, 2005.

[10] H. Moniz, N. F. Neves, M. Correia, and P. Veríssimo. Experimental comparison of local and shared coin randomized consensus protocols. In *Proceedings of the 25th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 235–244, 2006.

[11] Henrique Moniz, Nuno Ferreira Neves, Miguel Correia, Antonio Casimiro, and Paulo Verissimo. Intrusion tolerance in wireless environments: An experimental evaluation. In *Proceedings of the 13th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 357–364, 2007.

[12] P. Urban, X. Defago, and A. Schiper. Chasing the flp impossibility result in a lan or how robust can a fault tolerant server be? In *Proceedings of the 20th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 190 –193, 2001.

[13] Michel Raynal and Matthieu Roy. A note on a simple equivalence between round-based synchronous and asynchronous models. In *Proceedings of the 11th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 387–392, 2005.

[14] Nicola Santoro and Peter Widmayer. Agreement in synchronous networks with ubiquitous faults. *Theoretical Computer Science*, 384(2-3):232–249, 2007.

[15] U. Schmid, B. Weiss, and I. Keidar. Impossibility results and lower bounds for consensus under link failures. *SIAM Journal on Computing*, 38(5):1912–1951, 2009.

[16] Y. Moses and S. Rajsbaum. A Layered Analysis of Consensus. *SIAM Journal on Computing*, 31(4):989–1021, April 2002.

[17] J. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37(3):549–587, 1990.

[18] H. Moniz, N.F. Neves, and M. Correia. Turquois: Byzantine consensus in wireless ad hoc networks. In *Proceedings of the 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 537–546, 2010.

[19] Monica Dixit and Antonio Casimiro. Adaptare-FD: A Dependability-Oriented Adaptive Failure Detector. In *Proceedings of the 29th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 141–147, 2010.

[20] Rachid Guerraoui and Andre Schiper. Consensus: The big misunderstanding. In *Proceedings of the 6th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS)*, pages 183 –188, 1997.

[21] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, pages 255–270, 2002.

[22] James Aspnes. Fast deterministic consensus in a noisy environment. *Journal of Algorithms*, 45(1):16–39, 2002.

A Quality of Experience management module

Vlado Menkovski, Georgios Exarchakos, Antonio Liotta

Electrical Engineering Department
Eindhoven University of Technology
Eindhoven, The Netherlands
{v.menkovski, g.exarchakos, a.liotta}@tue.nl

Antonio Cuadra-Sánchez

Indra
Parque Tecnologico de Boecillo
47151 Valladolid, Spain
acuadra@indra.es

***Abstract*—User centric management of multimedia services necessitates understanding of how different system parameters affect the perceived quality of the service. Today's multimedia delivery systems are highly complex and diverse and span over multiple domains commonly in control of different entities. Furthermore, the end user devices are highly variable in capabilities and characteristics, which directly affect the perceived quality. In addition to these factors the perceived quality in multimedia is subjective by nature, which leads to complexities in accurate estimation with objective methods and costs with subjective estimation. To deal with the necessities of estimation of Quality of Experience (QoE) we have implemented a software module that aggregates information from probes and subjective quality feedback to develop QoE prediction models. These models are then used as part of the QoE management module to estimate the QoE of multimedia delivery streams in real-time. Additionally, the module calculates and suggests QoE remedies (improvements) for multimedia streams that underperform. The remedies are in the form of a single or multiple target parameter values that need to be reached in order for the stream to reach the desired QoE. The functionalities of this module enable an implementation of a user-centric management loop for multimedia services.**

Keywords-Quality of Experience, QoE, Machine Learning, Subjective Testing, User-centric services

I. INTRODUCTION

Management of multimedia streaming services incorporates optimizing the service performance and balancing the trade-off between the system resources and the delivered quality. To efficiently manage the services we need to understand how the different system resources and parameters affect the delivered quality.

First of all, perception of quality in multimedia content is subjective and highly varies on the mode of viewing such as handheld, desktop, large screen displays, on the characteristics of the display devices, screen size, resolution and color depth, to name a few. Next, the performance of multimedia services particularly streaming services also depends on factors prior to processing on the user terminal. The transport and encoding factors heavily influence the perceived quality or Quality of Experience (QoE) as well.

These factors, due to the high variability and diversity of the transport and communication systems are many and often outside of the control of the service provider. In light of this diversity and lack of performance guarantees, service providers tend to deploy management technique that uses 'standard' encoding parameters, regardless of the type of content or the display device used with the service. This technique does not deliver standard level of quality to the viewers, but a fixed burden on the system's resources. The lack of understanding of perceived quality from the service may result in suboptimal management decisions for the multimedia services.

User-centric management of multimedia services focuses on the quality as perceived by users rather than as delivered to them. In the former case the aim is user satisfaction while in the latter is the cancellation of any impairments during transmission. In certain environments, end-users may not be able to detect certain impairments, which do not affect their perceived quality. However, delivering more resources to those users is in fact, without reward. QoE-based multimedia streaming management is a more pragmatic approach and increases the capacity of the network as it contributes to a more careful resource allocation. That approach was initially introduced in [1] and consists of three phases: a) QoE estimation, b) design of potential recovery plans (remedies) and finally c) application of the most appropriate plan. As the latter is an operator specific step, the current work provides a method for predicting QoE with limited subjective data and an automated way for detecting appropriate recovery plans. In this paper we are additionally present specific software design and implementation details for the QoE estimation and remedies platform.

The basic idea behind is the execution of subjective studies and simultaneous network probing so that the correlation between network conditions and perceived quality is possible. This can provide enough information to teach a learning algorithm with the conditions that negatively affect QoE. Therefore, any detected condition that falls within that set of learning samples can be classified to the corresponding QoE level. Even though this process results in QoE estimation, the information collected have an even higher value, if combined. For instance, there is a clear view which conditions allow better QoE. A

comparison of these conditions can indicate suitable actions and resource management decisions to move from one QoE level to another. These conditions of target parameter values are the QoE remedies. The application module outputs them as a list or group of lists of one or more parameter values. Each group represents one possible remedy. Based on some defined cost functions the utility of each remedy can be calculated and based on that the most useful can be implemented. However, this is highly implementation specific so this final step can only be implemented within the system of the provider itself.

II. QoE Estimation

QoE is by definition what the end-user experiences while using a service [2]. This characterization is in one form or another is what most agrees upon. However, this does not mean that there is such an agreement on the means of measuring the QoE. There is quite a variety of methods that focus on measurement of QoE; they vary in accuracy and complexity [3]. Most common methods use objective techniques to measure the signal distortion whether in the encoding or in the transport stage. The International Telecommunication Union (ITU) has developed standardization document [4] for the various QoE models. The ITU classifies models as parametric, bit-stream, media layer and hybrid model. This classification characterizes the models focus. The parametric models look at network statistics and protocol information through network monitoring. They measure the signal distortions based on transport error statistics and network performance. The bit-stream models derive the quality via analysing content characteristics collected from the coded bit-stream information. The media layer models focus on the media signal and uses knowledge of the Human Visual System (HVS) to predict the subjective quality of video. This can be computationally expensive, but more over none of these methods analyse the QoE from a holistic perspective. Each model has a subset of aspects of QoE and does not take into account all the factors that affect it. Discussing the QoE estimation method efficiency is difficult because there is lack of a standard for comparing between each other [3].

However, even without being able to compare them directly we can understand the drawbacks of the objective approaches. The methods that only look at the fidelity of the audio and video neglect the effect that the type of content has on the perceived quality, as well as how the content is perceived by the HVS.

Typical examples of measurement of signal distortion by pixel to pixel comparison are the Peak Signal to Noise Ratio (PSNR) and Mean Squared Error (MSE) methods. The drawback of these methods is that they compare the signals without any understanding of the HVS [5]. A simple shift in the image will decrease the PSNR value significantly even though this will not be perceived as loss of quality by the viewers.

Modeling the effects that the transport has on the delivered quality would mean looking at the Quality of Service (QoS) parameters. This approach is not very efficient and yields weaker results [6]. The authors of [6] propose looking at the problem in three layers. The bottom layer being the network layer produces the QoS parameters or more precisely the Network QoS (NQoS) parameters. The layer above presents the application layer which is concentrated on parameters like resolution, frame rate, color, codec type, and so on. These parameters are referred to as Application QoS (AQoS). The third or the top layer is the perception layer which is driven by the human perception of the multimedia content and is concentrated on spatial and temporal perception and acoustic bandpass [6]. The QoE, which is measured on the top layer, is a function of both AQoS and NQoS (1).

$$QoE = f\left(AQoS, NQoS\right) \quad (1)$$

The proposed framework in [6] discusses that arbitrating all of the QoS parameters together is significantly more effective in maximizing the QoE than looking at each of them individually.

Due to the subjectiveness of QoE, the most accurate way to measure it is by executing subjective test. The subjective studies are of significant importance because they can accurately convey the satisfaction of the viewers with the service. This is why subjective tests are commonly used for comparing the capabilities of different QoE estimation methods. Subjective testing usually entails execution of tests in tightly controlled environment with carefully selected group of subjects that statistically represent the population, which is using the service. Guidelines for the execution of different subjective studies are provided by the ITU [7].

The drawbacks of subjective studies are obvious from their description. They require significant effort and resources to be put into their design and execution. In [8] and [9], the authors present a method that only relies on initial limited subjective tests. From the results of these tests, statistical models are build that can predict the QoE on unseen cases. The subjective tests executed in this work are based on the method of limits [10]. The viewers are presented with video in descending or ascending quality. The viewer detects the point where the perceived quality changes from acceptable to unacceptable in the descending series (or vice versa in the ascending). From these results the authors using discriminate analysis [11] have developed models for predicting the quality on unseen cases. This approach is suitable for minimizing the need for cumbersome subjective studies while providing for estimation based on the user's subjective feedback. However, the accuracy of the prediction models is limited due to the statistical method used to build the prediction models. The work is further extended in [12], where Machine Learning methods are used to build prediction models for QoE. These prediction models both Decision Trees (DT) [13] and Support Vector Machines (SVM) [14] outperform the discriminate analysis approach.

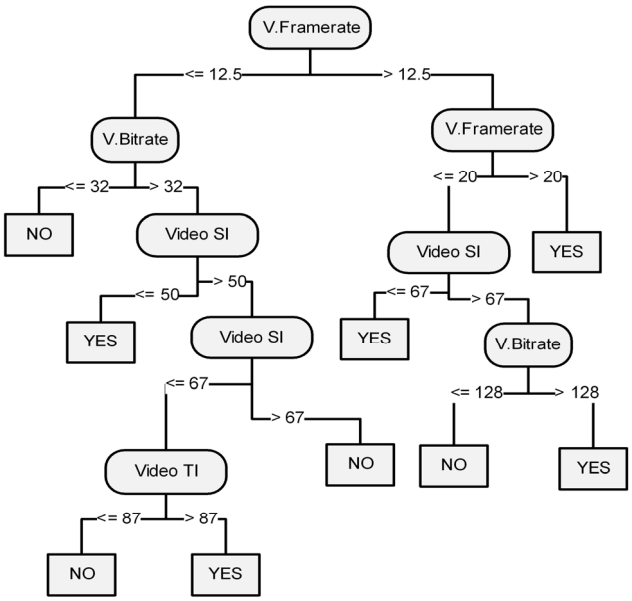


Figure 1. Decision Tree prediction model

In this work, the algorithm that performs best is C4.5 [15]. This is a DT induction algorithm, which builds a DT model from the training data. This DT model (

Figure 1) consists of nodes and leaves. The nodes are associated with splitting rules, based on a single attribute. The leaves of the DT are associated with class values, so that, all the datapoints that fall on a particular leaf are classified with the associated class.

Based on this subjective data, the models classifies as QoE acceptable is "Yes" or "No". Using this model unseen cases with different values of the attributes can be now classified as QoE acceptable or not acceptable.

The models in [12] perform with accuracy of above 90% estimated using the cross-validation technique [16].

III. IMPROVING QOE

Algorithms accurate QoE prediction models can be built using ML, having sufficient subjective data. Estimating the QoE is a crucial step in QoE-aware network management. However for a full implementation of the management loop we need to be able to maintain target QoE. Maintaining a target QoE involves determining the desired conditions that need to be achieved. In this section we are introducing a geometric technique that based on the QoE prediction model estimates the minimum needed changes in the measured stream parameters to improve the QoE.

This technique is enabled by the DT prediction models we use for estimating the QoE. One of the strengths of DT compared to other ML prediction models is their intelligibility. A DT in a way represents a set of rules stacked in a hierarchical way. Simple decision trees commonly define just a few rules that are deduced from the data and used for classification, but when the number of rules grows the size of the DT also grows, and with that, it loses its intelligibility. It is also possible to represent a DT model in the geometric space, defined by the dataset parameters. Consider each of the dataset parameters as a dimension in a hyperspace. Each of the datapoints form the dataset can be represented as a point in this hyperspace. The DT is represented by hyper regions formed by the leaves of the DT (Figure 2). Each node in the DT represents a split or a hyperplane that splits the hyperspace, until we reach a leaf, which carves out a hyper region. These hyper regions (as well as the leaves in the DT) are associated with a class label membership. So every datapoint or point in the hyperspace belongs to one of the regions of that the DT defined in the hyperspace, and as such is classified with the corresponding class label. In our particular case the hyper regions are associated class labels that are the QoE estimates.

In order to automate the QoE remedy estimation approach, we implemented an algorithm (Figure 3) that represents the DT in the hyperspace as follows:

This algorithm implements the DT representation in the dataset's hyperspace by generating a set of hyper regions that represent the tree leaves. Each hyper region contains a set of split rules that define the hyper-surface, which carves out the hyper region. The split rules are either representing an inequality of the type *Parameter1 >= Value1* or of the type *Parameter1 = Value1* depending on whether *Parameter1* is continual or categorical. If the leaf is on the left side of a continual *Parameter1* split then the split inequality will be 'more than or equal to', if it is on the right side the split inequality will be 'less than'.

Having a list of *HyperRegion*-s we can easily determine where each datapoint from the dataset belongs to, by testing the datapoint on the split rules of each hyper region. The hyper region is associated with the same class label as the leaf it represents, so all datapoints that belong to that region are classified as such.

In order to improve the QoE estimation of a particular stream, we need to look at the datapoint that was generated by the monitoring system for that stream. If the datapoint is classified with a QoE value that is not satisfactory, we look at the distance to a set of hyper regions $\overline{\Phi}$ that are associated with a satisfactory QoE value. The distance to each of the desired regions is the difference in parameter values that are needed in order to move the datapoint to the desired regions.

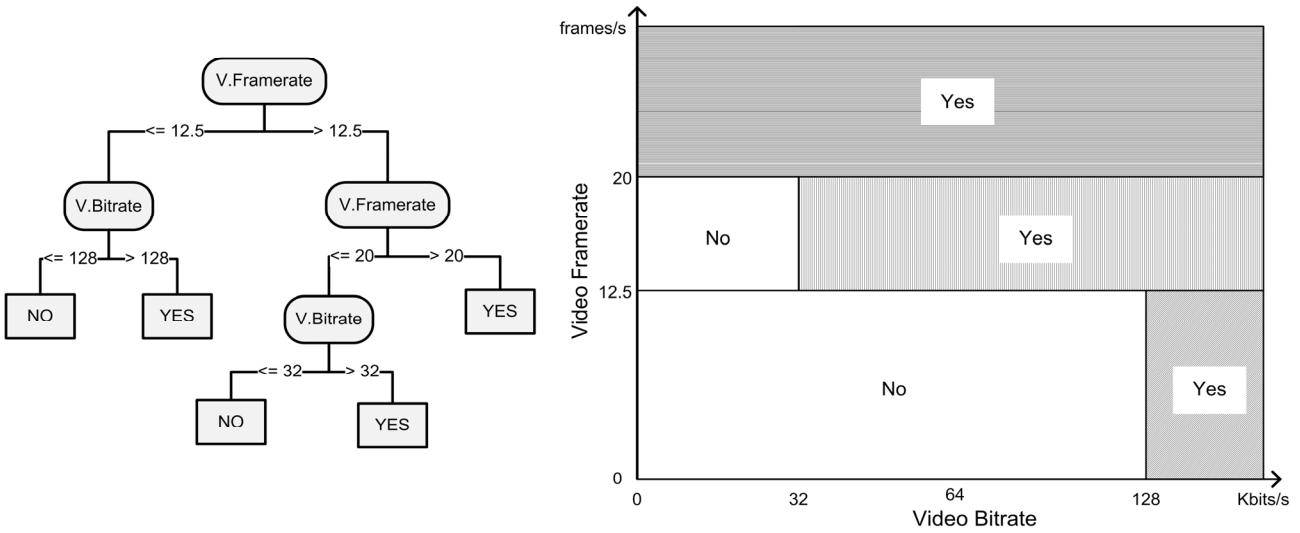


Figure 2. Simple decision tree in 2D space

The output of the algorithm is a set of distance vectors, which define the parameters that need to be changed and their change values.

To illustrate the matter better we can take an example from the laptop dataset from [12]. The prediction model built from this dataset is given in

Figure 1. If we look at the datapoint given in Table 1 we can see that this datapoint will be classified by the model as QoE = No ('Not Acceptable'). Since the V. Framerate is less than 12.5 and the V.Bitrate is less than 32 the datapoint reaches a leaf with 'Not Acceptable' class associated with it.

Now, what is the best way to improve the QoE of this stream?

Start from the root node and call a recursive method *FindLeaves*

FindLeaves:

1) If the node has children
   a) Call *FindLeaves* on each child
   b) Add the SplitRule on each of the Hyper Regions ($\overline{\Phi}$) that are returned
      i) If the leaf split is categorical add a Split Rule: Attribute = 'value'
      ii) If the leaf split is continual add on the leaves from the left side SplitRule: Attribute < value, and on the leaves from the right side Attribute > value
   c) Return the set of Hyper Regions ($\overline{\Phi}$)
2) Else, you are in a leaf
   a) Create an Hyper Region object
      i) Assign the class of the leaf to the Φ
      ii) Return Φ

Figure 3. DT to Hyper Region algorithm

First of all there are parameters that characterize the type of the content such as the Video SI and the Video TI and cannot be changed. In this dataset structure we are looking into increasing the V.Bitrate and V.Framerate. If we increase the V.Bitrate for this particular datapoint by one step to 64kbits/s we can see that the datapoint goes now down the decision tree to one of the bottom leaves, but it is still classified as QoE Acceptable = No. On another hand if we increase the V.Framerate to 15f/s we can see that the datapoint is classified as QoE Acceptable = Yes without adding more bandwidth.

TABLE I. EXAMPLE DATAPOINT

Video SI	Video TI	V. Bitrate	V. Framerate
67	70	32	10

We can deduce a rule from the model that a video with these characteristics needs to have higher V.Framerate for it to be perceived with high quality. However, this rule is not easily evident from only looking at the model. We can also imagine a system with large number of attributes that we can change where tuning this attributes the right way becomes an increasing problem. Further down this line of reasoning, if we want to make a system-wise improvement that will increase the QoE of most streams we cannot easily derive which parameters are best to be increased and by how much.

In the case of the example datapoint the algorithm returns the two possible paths:

- Increasing the Framerate to above 12.5f/s
- Increasing the V. Bitrate to above 32kbits/s and the Video TI to above 87

Since we know that increasing the Video TI is not an option, because this is defining the type of content we can see, then the only option is to increase the frame rate. In a general case, there can be many different paths to a hyper region with the desired class.

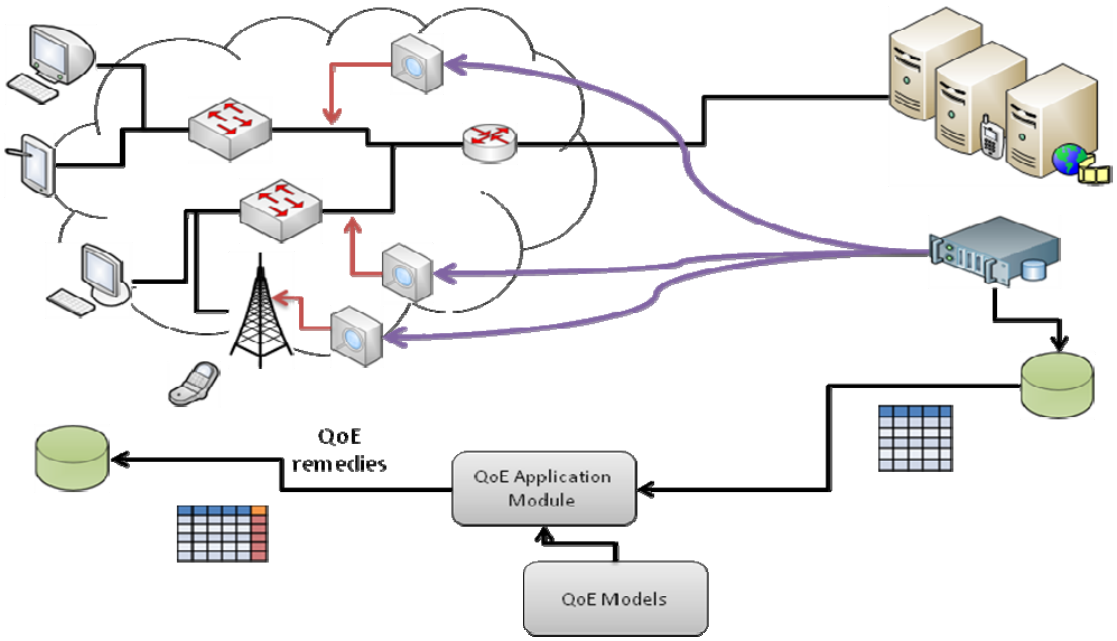


Figure 4. QoE estimation and remedies architecture

To automate the process we can assign cost functions to the change of the attribute values and automatically calculate the cheapest way to reach the desired QoE. In this manner attributes that are not changeable, such as the Video TI, can have infinite value of the cost function.

Given a datapoint and a target label the algorithm produces a set of change vectors. Each of the change vectors applied to the datapoint moves the datapoint to a hyper-region classified with the target label. In other words, each change vector is one possible fix for the datapoint.

$$\overline{\Phi} = FindLeaves(DT, QoE) \quad (2)$$

$$\overline{\Delta\varphi_i} = Distance(\Phi_i, \bar{d}) \quad (3)$$

$$\Delta\varphi_{optimum} = \min_i \left(Cost(\Delta\varphi_i) \right) \quad (4)$$

In (2), $\overline{\Phi}$ is a set of regions with a targeted QoE value. The distance function in (3) calculates the vector of distances for each attribute to the target region in $\overline{\Delta\varphi}$. The optimal distance vector is the one with minimal cost (4) for the given input datapoint $\bar{d}$. The *Cost* function in (4) is dependent on the application. Each system has explicit and implicit costs associated with changes of specific parameters.

IV. SYSTEM ARCHITECTURE

The QoE estimation and remedies module is a software application that inputs monitoring data from the monitoring system in a predefined manner and outputs QoE prediction and remedies (Fig. 4). The monitoring system collects statistics of the multimedia service performance and records systems parameter values. These values are collected in a monitoring database. When the application model detects new monitoring data in the database it loads these records and executes a QoE prediction. The estimated value of the QoE is augmented to the original monitoring data and submitted in the database. Then the estimation is compared to the target QoE value. If the particular measured stream does not meet the target QoE a remedy is calculated. The remedies are a group of target parameter values that if reached will improve the stream QoE up to the target value. There could be more than one parameter value that needs to be changed in a single suggested remedy. More over the application will also suggest other possible groups of parameter values as alternative remedies. If costs are associated with the parameter change then the application can calculate the most effective way of improving the QoE by suggesting the cheapest remedy. However, this is highly system specific and needs to be defined by the service provider them self. One example of such a situation could be choosing between improving the QoE by increasing the bit-rate or lowering the packet loss. Increasing the bit-rate might be possible up to some level, lowering the packet loss may be a matter of using different transport protocol or it might not be possible in the given latency constraints.

Due to the complexities associated with defining the costs and utility of the parameter changes the QoE application module is just an enabling component of a QoE management platform. A significant integration effort needs to be implemented in order for a full QoE management platform to be implemented.

V. SOFTWARE IMPLEMENTATION

The QoE module software is implemented as an extension of the Weka [17] machine learning platform. The Weka platform is provides for a flexible way to build and use ML prediction models. It can load the data from several existing database implementations of from a few text file formats. It has a clearly defined API (application programming interface), which allows for integration with other software components. It provides for extending the library for use adding more functionality and a wide variety of ML algorithm implementations.

The application consists of a main loop which instantiates data listeners. The listeners can be passively waiting for a data file to appear in a directory or actively pooling a database for new data. The main application prior to instantiating the listeners loads the instances of the prediction models. The prediction models are built using the Weka explorer user interface, and then serialized to a local directory.

The listeners now are able to query the models for prediction when new data arrives. They first parse and adapt the data accordingly then call the models to estimate the QoE. This value is recorded and then saved in an extra column in the monitoring database.

If the QoE estimation does not meet the minimum required level the remedy algorithm is executed to create the remedy suggestions.

The remedy algorithm starts by creating a list of HyperRegion (Fig. 5) objects from the give prediction model.

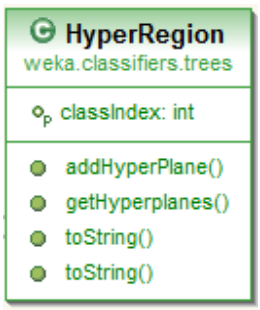


Figure 5. HyperRegion class diagram

For each leaf in the DT the algorithm outputs a HyperRegion object. The object contains a number of hyper planes that define the region in the data space. Each hyper plane corresponds to each split in the decision tree until the leaf is reached. Each of the HyperRegions has a classIndex that corresponds to the QoE level associated for the hyper region and DT leaf. For the implantation of the remedy algorithms we extended the J48 existing code to include a get distance method (Fig. 6).

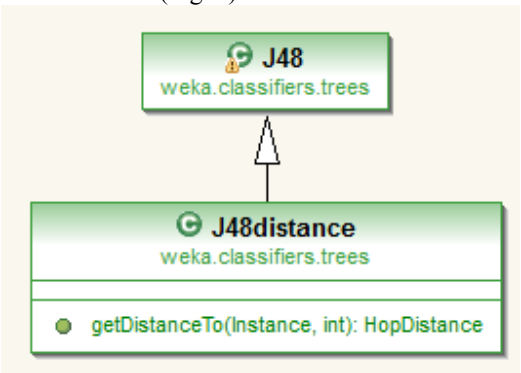


Figure 6. J48 Class diagram

This method is necessary to calculate the distance of a point to a particular hyper region and returns HopDistance object. The remedies algorithm calls the getDistanceTo() method from the model for the particular datapoint and each HyperRegion with QoE value at and above the target level.

Figure 7. HopDistance class diagram

Each of the HopDistance (Fig. 7) objects contains a list of parameter values distances from the current to the target that need to be reached so that the data point can be moved into a HyperRegion with a satisfactory QoE. These HopDistance-s are then adapted into remedies and outputted from the module as suggestions. If the parameter value change costs are defined, then the algorithm can calculate the costs for each HopDistance or remedy and select the most cost effective one for output. At this point the module can be coupled with a management implementer who can execute action that will implement the given remedies and automate the full QoE management process.

VI. CONCLUSION

In this work, we present a user-centric approach and a software module implementation for management of streaming multimedia services. This approach is based on developing DT models with ML tools that can estimate the QoE of the streaming service. Furthermore, we have

developed a geometric approach that represents the DT model in the dataset feature space, which derives a way for estimating the changes needed for improving the quality of specific streams. Effectively we have presented a functional description of a method that can estimate the QoE and find the possible remedies for improving it, if it is not satisfactory. We have analyzed the method through a worked example with data and models from a subjective study of mobile multimedia streaming. These models show that ML tools can build accurate DT prediction models, which can be used to estimate the possible remedies to reach satisfactory QoE. We present a software module implementation that integrates with a streaming service monitoring system and augments it with estimation for the QoE and its remedies based on the subjective feedback of the customers.

This ongoing research will continue towards the quantitative besides qualitative assessment of the method. There is an undergoing plan to deploy the system on an existing communication network in collaboration with network providers. These experiments are expected to give enough evidence about the financial benefits for a provider in deploying our system.

ACKNOWLEDGMENT

The work included in this article has been supported by Telefonica R&D and Indra (Spain). The authors thank María del Mar Cutanda, head of division at Indra, for providing guidance and feedback.

REFERENCES

[1] V. Menkovski, G. Exarchakos, A. Liotta, and A. Sanchez, "Estimations and Remedies for Quality of Experience in Multimedia Streaming," in *Advances in Human-Oriented and Personalized Mechanisms, Technologies and Services (CENTRIC), 2010 Third International Conference on*, pp. 11-15, 2010.

[2] S. Winkler and P. Mohandas, "The Evolution of Video Quality Measurement: From PSNR to Hybrid Metrics," *Broadcasting, IEEE Transactions on*, vol. 54, no. 3, pp. 660-668, 2008.

[3] S. Winkler, "Video Quality Measurement Standards - Current Status and Trends," in *Proceedings of ICICS 2009*, 2009.

[4] A. Takahashi, D. Hands, and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," *Communications Magazine, IEEE*, vol. 46, no. 2, pp. 78-84, 2008.

[5] S. Winkler, *Video Quality and Beyond*. Symmetricom, 2007.

[6] M. Siller and J. Woods, "QoS arbitration for improving the QoE in multimedia transmission," in *Visual Information Engineering, 2003. VIE 2003. International Conference on*, pp. 238-241, 2003.

[7] R. I. ITU-T, "910," *Subjective video quality assessment methods for multimedia applications*, 1999.

[8] F. Agboma and A. Liotta, "Addressing user expectations in mobile content delivery," *Mobile Information Systems*, vol. 3, no. 3, pp. 153-164, Jan. 2007.

[9] F. Agboma and A. Liotta, "QoE-aware QoS management," in *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*, pp. 111-116, 2008.

[10] G. T. Fechner, E. G. Boring, H. E. Adler, and D. H. Howes, *Elements of psychophysics / Translated by Helmut E. Adler ; Edited by David H. Howes [and] Edwin G. Boring ; with an introd. by Edwin G. Boring*. New York :: Holt, Rinehart and Winston, 1966.

[11] W. R. Klecka, *Discriminant analysis*. SAGE, 1980.

[12] V. Menkovski, A. Oredope, A. Liotta, and A. Cuadra Sánchez, "Predicting Quality of Experience in Multimedia Streaming," in *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, pp. 52-59, 2009.

[13] J. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81-106, Mar. 1986.

[14] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[15] J. R. Quinlan, *C4.5*. Morgan Kaufmann, 2003.

[16] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence*, vol. 14, pp. 1137-1145, 1995.

[17] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.

Speech Translation Statistical System for Teaching Environments and Conference Speeches

Jesus Tomás, Alejandro Canovas, Jaime Lloret, Miguel García
Instituto de Investigación para la Gestión Integrada de Zonas Costeras
Universidad Politécnica de Valencia, Spain
{jtomas, alcasol, jlloret, migarpi}@upv.es

Abstract— **The synergic combination of different sources of knowledge is a key aspect in the development of modern statistical translators. The effect and implications of adding additional other-than-voice information in a voice translation system for teaching environments and conference speakers is described in this work. The additional information serves as the bases for the log-linear combination of several statistical models. A prototype that implements a real-time speech translation system from Spanish to English is presented. In the scenario of analysis a teacher, or presenter, as speaker giving its presentation could use a real time translation system for foreign students or participants. The speaker could add slides or class notes as additional reference to the voice translation system. Should notes be already translated into the destination language the system could have even more accuracy. In this paper, first, we present the theoretical framework of the problem, then, we summarize the overall architecture of the system, next, we specify the speech recognition module and the machine translation module, then, we show how the system is enhanced with capabilities related to capturing the additional information, and, finally, we present the performance results of the developed system.**

Keywords-pedagogical tool; adaptation; speech recognition; speech translation; natural language processing.

I. INTRODUCTION

The development of automatic real-time translation systems from voice signals constitutes a long-term objective. However, recent advances in the field of statistical translation increase the possibility of an actual widespread usage in the near future [1, 2].

On one side, an ever-more increasing number of foreign students, interchange programs, and alike in Europe (reaching a 15% of the students in the Higher Polytechnic School of Gandia, Polytechnic University of Valencia), and, on the other side, multi language meetings, workshops and conferences, are requiring more efforts to provide tools and means that help the integration in the learning and speaker presentations processes while the new language skills are getting developed. A tool like the one presented in this article could increase the learning rate provided by the spoken classes and the attendance of foreign-language speakers.

We hence provide a prototype that demonstrates the viability of the real-time speech translation in the pedagogical student-teacher environment. Given the fact that current status-of-the-art products and techniques in the area of automatic real-time translation are far from perfect, we enhance the results by providing beforehand material about the elements of translation, e.g., specific vocabulary, texts, etc. With this purpose, our speech translation system is fed with slides and the class or presentation notes previous to the operation of the system. Often, these sources or information are already translated and handed over to the student. We make use of the offline translation as input to the system as well.

In this paper we explain how to adapt an existing real-time speech translation system to incorporate the usage of the above mentioned additional information, and how this impacts positively in the accuracy results of the translation. Notably, the system improves the alignment using off-line data taken from the tool output. A good environment to apply this system is in teaching, because the teacher's slides and notes could be used as additional information. Another good environment is multi-language meetings, workshops, and conferences. This, together with a very good real time response in the translation recognition, makes our proposal an ideal tool for this type of environments.

This paper is an extension of the paper presented in a conference [3].

The remainder of this paper is organized as follows. Section 2 presents some related works about the speech-to-speech translation systems. Section 3 explains our prototype in order to let the reader know how will work the final product. Statistical Spoken Language Translation used in our system is described in section 4. Section 5 shows the architecture of our system and introduces the modules used in it. The speech recognition module is explained in section 6. Section 7 explains the machine translation module. The adaptation system is described in Section 8. The system evaluation is shown in Section 9. Section 10 concludes the paper and gives our future work.

II. RELATED WORKS

Nowadays there are several lines of research in speech-to-speech translation systems. For example NESPOLE!, which is a speech-to-speech machine translation research project funded jointly by the European Commission and the US NSF [4]. The prototype system developed in NESPOLE! is intended to provide effective multi-lingual speech-to-speech communication between all pairs of four languages (Italian, German, French and English) within broad, but yet

restricted domains. The idea of this project is to allow a communication online client-server on which both parties are expressed in different languages. The transmitter's phrases are translated and heard by the receiver by means of sensitized speech.

Many research projects have addressed speech-to-speech translation technology, such as VERBMOBIL [5], C-STAR [6], BABYLON [7] and S2ST [8]. The last is quite interesting because it is mainly focused on translation between English and Asian languages (Japanese and Chinese). This requires advanced technologies to overcome the drastic differences in linguistic expressions.

The speech translation system used in our work is based on hidden Markov models and n-grams as [9]. Nowadays, the most successful speech translation systems are based on stochastic finite-state networks. This paper was written using the methodologies developed and the data collected in "The EuTrans-I speech translation system" project. This speech translation is accomplished in using a procedure that is similar as the one used in our speech recognition. Stochastic finite-state transducers, which are specific stochastic finite-state networks, have proved very adequate for translation modeling. The acoustic, language and translation models are finite-state networks that are automatically learnt from training samples. Other interface between automatic speech recognition and machine translation are the confusion networks. In [9], the authors obtained next conclusions: "Confusion networks, from one side, permit to effectively represent a huge number of transcription hypotheses, and from the other side, lead to a very efficient search algorithm for statistical machine translation".

In the paper presented in [10], the problem boils down to the question of how to arrive to a suitable interaction between the recognition process and the translation process. In this study they try to combine distinctive features derived from both modules: speech recognition and statistical machine translation. All the features from the speech recognition and the machine translation modules were combined by the log-linear models seamlessly. It is very interesting for us their conclusions derived from speech recognition: likelihood of acoustic and language models, helped to improve the speech translation. The N-best recognition hypotheses are better than the single-best ones when they are used in translation. They show that N-best recognition hypothesis translation can improve speech recognition accuracy of incorrectly recognized sentences. This same approach has been made in [11]. In this paper, they attempt to derive a suitable Bayes decision rule for speech translation and to present suitable implementations. The authors introduce specific modeling assumptions to convert the Bayes decisions into a practical algorithm. We thought that to compare our work with the works in reference [9] and [11] might be interesting, but it is difficult. The tasks are different and the tools used in these tasks are not open source. In the beginning, we considered to work with one of these tools, but this tool does not compute in real time.

Our translation system is based on the Corpus of the European Parliament. We take this corpus as training data for statistical machine translation. In [12], the authors describe the acquisition of the corpus and its application to the statistical machine translation. These training corpuses are usable thanks to the work performed by some research groups such as "The statistical machine translation group" at the University of Edinburgh. They participated in the transcription and translation tasks for five language pairs, using only the supplied corpora, for example in [13].

In [14], the authors present an overview of their current out-of-the-box system. It includes a detailed treatment of models added over the last years, especially a novel lexicalized reordering model. The system employs a phrase-based statistical machine translation model that uses the Pharaoh decoder [15].

III. PROTOTYPE DESCRIPTION

The prototype presented in this paper implements a real-time speech translation system to support a Spanish speaker with English-speaking speech listeners.

First, the speaker provides the slides in a MS PowerPoint format making sure that the notes area of each slide contains an explanation of the slide. The explanation in the notes area should be very close to what the speaker is going to say explaining the slide.

Before starting the speech, the speaker must load the PowerPoint file in the system. At this point, the system gets adapted to reflect the text within the slides, increasing the probability to guess the right words to appear when each slide is presented. The procedure is used all along the duration of the presentation.

When the presenter is speaking, the system recognizes the sentence, translates it, and is able to display the subtitle translation as caption to the slide. An example is seen in Figure 1. The last two lines are superimposed to the projection of the slide. In the bottom, what the presenter said appears written (in Spanish); while the line above it corresponding translation in English is written. In this way, an English-speaking student is able to relate the content with the Spanish representation through the displayed translation.

Figure 1. Slide example.

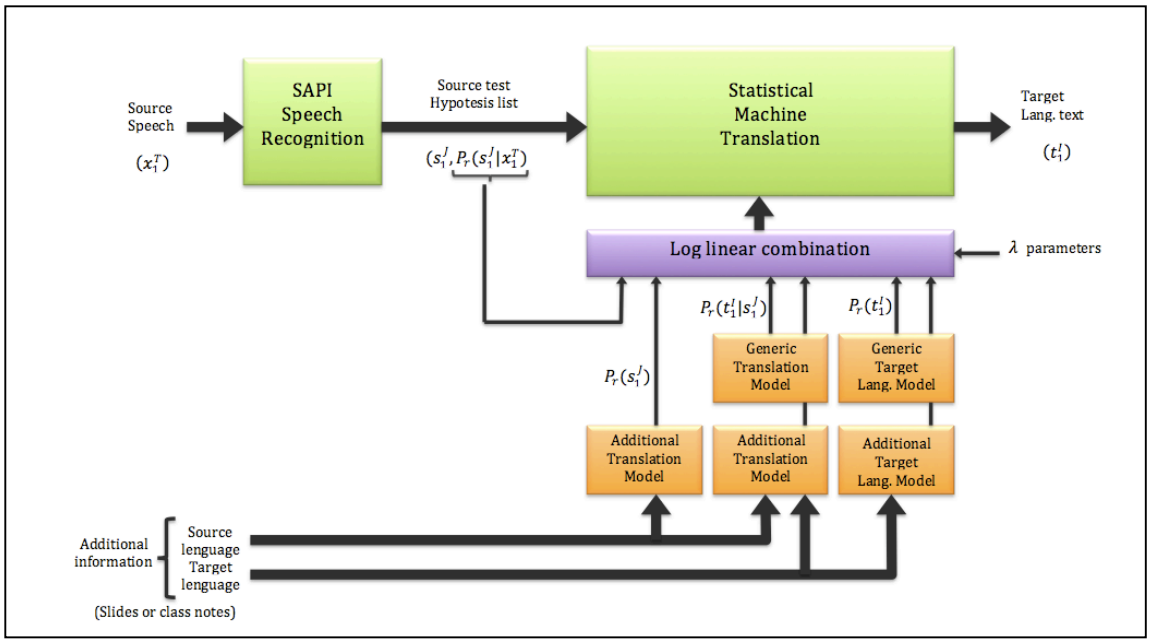


Figure 2. System architecture.

IV. STATISTICAL SPOKEN LANGUAGE TRANSLATION

The goal of Statistical Spoken Language Translation [14, 16] is to translate a given acoustic observation vector $x_1^T = x_1 \dots x_T$ into a target sentence $t_1^I = t_1 \dots t_I$.

The methodology used in [17, 18] is based on the definition of the function $Pr\,(t_1^I|x_1^T)$ that returns the probability that t_1^I is a translation of a given acoustic observation. We can introduce a hidden variable that represents the source sentence, $s_1^J = s_1 \dots s_J$. Then, we can write equation 1.

$$\hat{t}_1^{\hat{I}} = \underset{t_1^I}{\text{argmax}}\, Pr(t_1^I|x_1^T) = \underset{t_1^I}{\text{argmax}} \sum_{s_1^J} Pr(s_1^J, t_1^I|x_1^T) =$$

$$= \underset{t_1^I}{\text{argmax}} \sum_{s_1^J} Pr(s_1^J|x_1^T)\, Pr(t_1^I|s_1^J) \approx$$

$$\approx \underset{t_1^I,\, s_1^J}{\text{argmax}}\, Pr(s_1^J|x_1^T) Pr(t_1^I|s_1^J) \qquad (1)$$

Following the log-linear approach [10, 11], $Pr(t_1^I|s_1^J)$ can be expressed as a combination of a series of feature functions, $h_m(t_1^I, s_1^J)$, that are calibrated by scaling factors, λ_m, as it is shown in equation 2.

$$Pr(t_1^I|s_1^J) = \sum_{m=1}^{M} \lambda_m\, h_m(t_1^I, s_1^J) \qquad (2)$$

This framework allows us a simple integration of several models in the translation system. Moreover, scaling factors let us adjust the relative importance of each model. Bearing in mind this objective, Och and Ney propose a minimum error rate criterion in [19].

V. ARCHITECTURE

Our system architecture is based in two modules, as Figure 2 shows.

A. Speech Recognition Module (SRM)

The Speech Recognition Module (SRM) takes the audio input stream from a microphone and obtains an N-best output text. In the N-best list, each hypothesis is scored according to the equation $Pr(s_1^J|x_1^T)$.

Although there are several open source speech recognition systems available, like Sphinx [20] or HTK [21], we have used the standard system provided by the MS Windows Vista OS, because it is the only one that incorporates acoustic models for Spanish. They are available to non-restricted tasks. The communication with this engine is based on a SAPI interface [22].

In addition, this engine has a few interesting capabilities that makes it well suited for a real-time application like ours. It let us customize its functionality for a specific speaker and task, which allows us to work with multiple output hypotheses simultaneously.

B. Machine Translation Module (MTM)

The Machine Translation Module (MTM) is based on a previous work (described in [23]). Basically, in order to estimate $Pr(t_1^I|s_1^J)$, a log-linear combination of several statistical models is used. In our application, two models are needed, one for the translation itself and one for the target language selected. A new important feature is introduced in this work, the output score of the speech recognition module.

Therefore the machine translation module integrates the following knowledge:

- Translation model. It is based on monotone phrase-base models. Phrase-base models divide the sentence into segments, each one of them composed by a series of words. Now, the translation probabilities relate a sequence of words in a source sentence with another sequence of words in the target sentence. The simplest and fastest formulation in such models is based on monotone models [23]. However, to operate in real time the speed of the translation search is a critical factor. Thus, we have to select a monotone phrase-based model.
- Target language model: It is comprised of two sub-models, a conventional trigram model, $p(t_i|t_{i-2}^{i-1})$, and a five-gram class model: $p(T_i|T_{i-4}^{i-1})$.
- Speech recognition score: It is the output of the speech recognition module.

VI. *SPEECH RECOGNITION MODULE*

Generally, a speech recognition application involves the detection of the user's voice and the interpretation of what he/she has said. In our case, the speech recognizer is a fundamental tool of the application. Once the recognizer detects and interprets what the user says, the information is passed to the translation module and, then, it is displayed on the application screen.

The voice detection of our prototype is performed by the SAPI voice recognition engines. Specifically, we use the recognition engine included in Spanish Windows SDK for Windows Vista. Figure 3 shows a diagram of the structure of the speech recognition module. The figure describes and explains how the speech recognition module is integrated into our system.

As it is shown in the previous scheme, the information provided by the speaker's speech flows from the top to the bottom, but the configuration information is from the bottom to the top. For the text to the speech translation scheme, the procedure is more or less the same, but starting from the information based on applications and ending at the speakers.

The high-level interface is implemented in Microsoft COM objects in order to call in a simple way to a low-level interface. It is essential to load the libraries of Microsoft speech recognition in order to access these objects. On the other hand, the low-level interface is implemented by the TTS and SR engines. SAPI interface allows the exchange of these engines without needing to reprogram the application. Therefore, an application can choose between different voice recognition engines, such as the speech recognizer Windows SDK, IBM or the Dragon, among others.

The main interfaces for speech recognition in ISAPI are ISpRecoContext, ISpRecognizer and ISpGrammar. What follows is a brief explanation of a thread of different interfaces in the SAPI speech recognizer.

A. Speech recognition interface

1) ISpRecoContext

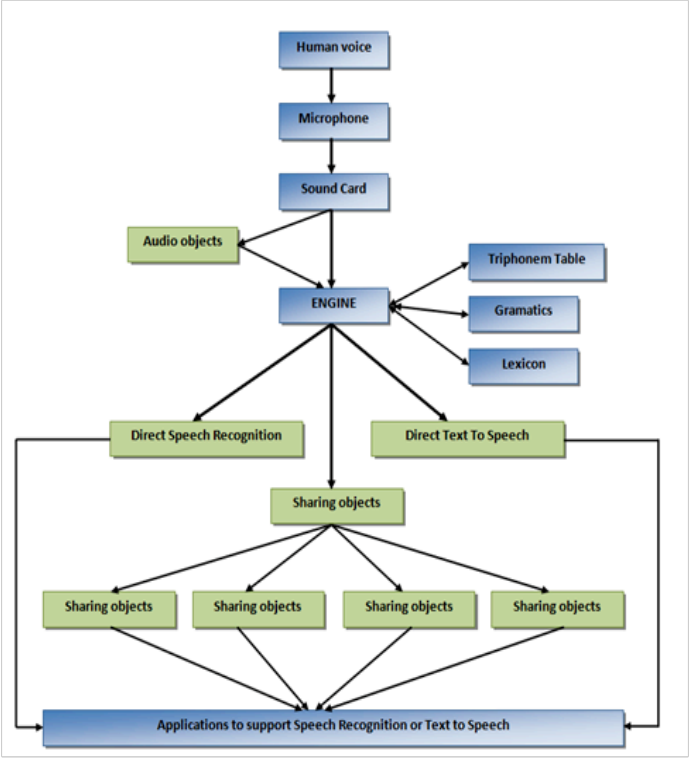


Figure 3. Speech recognition module.

ISpRecoContext is the main interface in the Windows Vista OS speech recognition.

ISpRecoContext interface allows the application to create different functional points of views or context of the SR (SpeechRecognition) engine. ISpRecoContext, as well as ISpVoice, is ISpEventSource. ISpRecoContext is used by the voice application to receive notifications from different events during the voice recognition. Therefore, this object allows an application to start and stop the recognition, receive results from the recognition, analyze the words and sentences pronounced by the user and other events. For example, it will not be the same for an application to use the world "close" in a dictate, being referred to close a door, that "close" to close a desktop application. Both belong to different contexts.

An application could have different contexts. Those words that are not inside the context could be included.

In order to convert the acoustic entrance into a stream in SAPI, first we call the object ISpRecognizer and create a RecoContext using the method CreateRecoContext. Therefore, when we create an ISpRecoContext from an ISpRecognizer, SAPI is converting the voice to text.

2) ISpRecognizer

ISpRecognizer interface allows the application to control some the voice recognition engine (SR Engine) and its audio input. Each ISpRecognizer interface represents a unique SREngine.

There are two possible implementations of the ISpRecognizer in SAPI. One is for the “in-process” (InProc) recognition, where only our application could connect to the recognizer. It is used in the situations in which a maximum performance, lower response time or high quality recognition

is being searched. And another implementation is "shared-recognizer", where all voice applications work simultaneously using a shared engine connected to the same recognizer. In this way, when a user talks, the recognition engine will decide which context is the most convenient between all possibilities.

3) ISpRecoGrammar

ISpRecoGrammar interface allows the application to manage the words and sentences recognized by the voice recognition engine.

A SpRecognizer object can have several SpRecoContext objects associated to it and a SpRecoContext object could have several SpRecoGrammar associated to it. This let us create voice recognition applications with several grammars. Therefore, a SpRecoGrammar object could have a contact free grammar and a dictate grammar simultaneously.

B. Voice recognition process execution thread.

In order to create an ISpRecoContext object, first we create an ISpRecognizer "in-process" object from the application. Then, we call ISPRecognizer::SetInput to activate the audio input and an ISpRcognizer::CreateRecoContext to obtain an ISpRecoContext object. Next, we activate the notifications of the events generated by the voice recognition. After that, we create an ISp_Recognition object. This object will inform us if the ISpRecognizer has recognized a voice.

Last, a voice application should create, load and activate a ISpRecoGrammar. This object gives us information basically about the model, that is, if it is a dictate or control commands. In order to create this interface, we use the following call ISpRecoContext::CreateGrammar. Then, the application loads the appropriate grammar, through calling an ISpRecoGrammar::LoadDictation for dictate or an ISpRecoGrammar::LoadCmdxxx for control commands. In order to activate these grammars or models, the application calls to ISPRecoGrammar::SetDictationState for a dictate or to ISPRecoGrammar::SetRuleState or ISpRecoGrammar::SetRuleIdState for control commands.

It is important to highlight that the applications based in voice synthesis use grammars. They should be specified and loaded when the application is starting. The dictate grammar is freer because it allows using higher number of language words. However, the grammar word list of control commands is limited.

Figure 4 summarizes the process previously explained.

It is important to say that voice recognition only happens when a RecognizeStream event is produced by the recognition engine, that is, when a stream is processed by the engine. Then, the result of this event is sent to SAPI. This reply is not sent until the stream recognition is fully finished. This synchronization between SAPI and the recognition engine is controlled by the engine, because until there is not a reply from the engine, there is no recognition.

C. Several events in the application execution.

One of the issues to take into account during the development of our project has been the analysis performed of the produced events when the application is being executed. The application follows the execution thread shown in figure 5. We can see in this figure that the first produced event is sent by the engine when a sound is detected (*On sound Start*). If the sound input set is recognized by the engine, it will be processed as a *stream* until a sentence is formed, creating the state *recognizer State Change* in the recognition. If it the sounds are not recognized, the engine will send an *interference event* in the recognition.

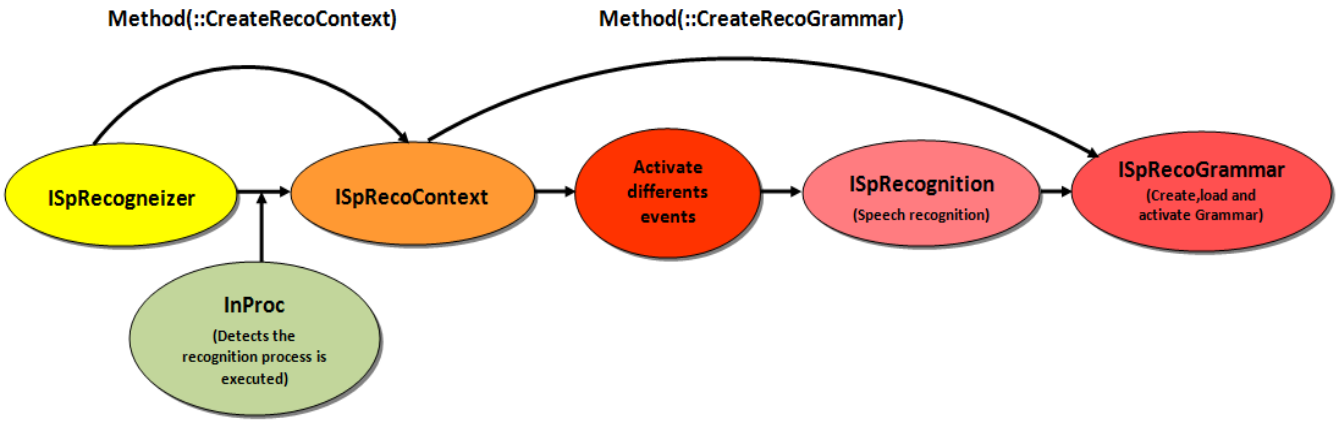


Figure 4. Windows interface execution thread.

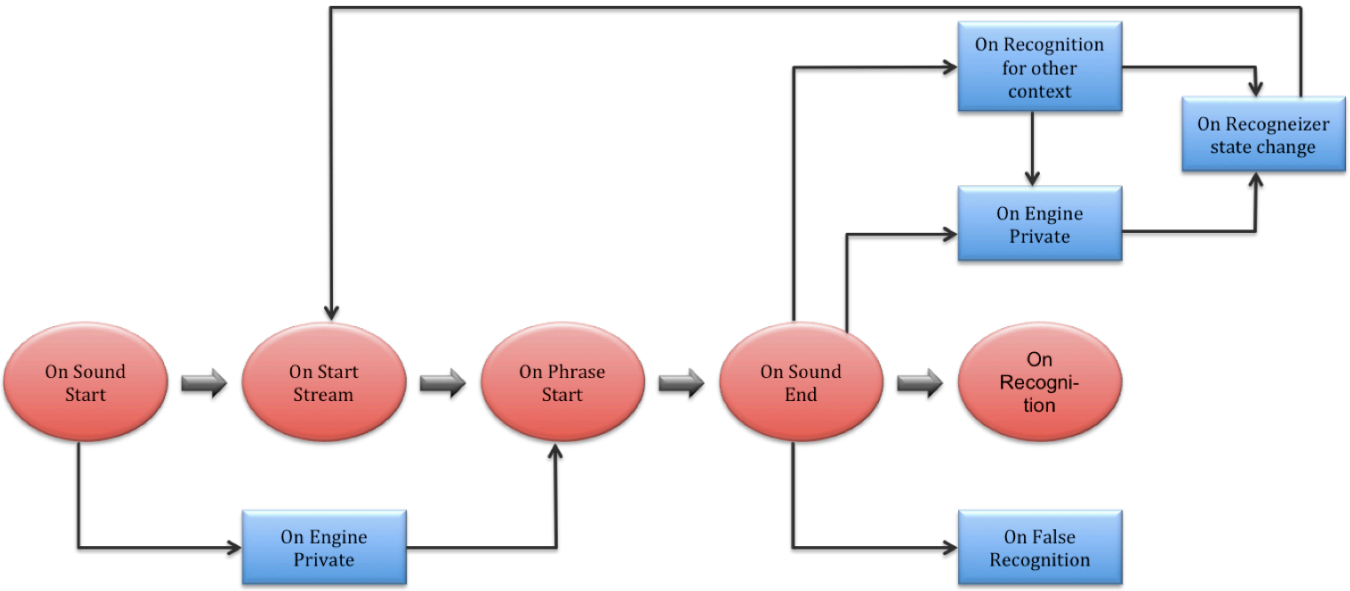


Figure 5. Application execution thread.

When one of these events has been produced, the system re turns an end of sound event and the recognition result. This result can be basically classified in a correct recognition, in a false recognition, or may be the recognition is not considered coherent or may belong to another context.

VII. *Machine Translation Module*

Machine translation module is based in Statistical Machine Translation (SMT). The goal of SMT is to translate a given source language sentence, $s_1^J = s_1 \dots s_J$, into a target sentence $t_1^I = t_1 \dots t_I$. In order to achieve this goal, a function $Pr\left(t_1^I | s_1^J\right)$ is defined. It estimates the probability of obtaining t_1^I as a translation of a given s_1^J. Following the log-linear approach [19], this function can be expressed as a combination of a series of feature functions, $h_m(t_1^I, s_1^J)$, as it is shown in equation 3.

$$\hat{t}_1^I = \underset{t_1^I}{\operatorname{argmax}} Pr\left(t_1^I | s_1^J\right) = \underset{t_1^I}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m\left(t_1^I, s_1^J\right) \quad (3)$$

A. Phrase-Based Models

For many SMT systems, the most important feature function in equation 3 is the PB model. The main characteristic of this model is that it attempts to calculate the translation probabilities of word sequences (phrases) rather than only single words. These methods explicitly estimate the probability of a sequence of words in a source sentence (s_1^J) to be translated as another sequence of words in the target sentence (t_1^I).

To define the PB model, we segment the source sentence s_1^J into K phrases $(\tilde{s}_1^K)$ and the target sentence t_1^I into K phrases $(\tilde{t}_1^K)$. A uniform probability distribution over all possible segmentation is assumed. If we assume a monotone alignment, the target phrase in k position is produced only by the source phrase in the same position [24]. Then we get equation 4.

$$Pr\left(t_1^I | s_1^J\right) \propto \max_{K, \tilde{t}_1^K, \tilde{s}_1^K} \prod_{k=1}^{K} p\left(\tilde{t}_k | \tilde{s}_k\right) \quad (4)$$

where the parameter $p(\tilde{t}|\tilde{s})$ estimates the probability of translating the phrase $\tilde{s}$ into the phrase $\tilde{t}$. A phrase can be comprised of a single word (but empty phrases are not allowed). Thus, the conventional word-to-word statistical dictionary is included. If we permit the reordering of the target phrases, a hidden phrase level alignment variable, α_k, is introduced. In this case, we assume that the target phrase in k position is produced only by the source phrase in position α_k. Then, we obtain equation 5.

$$Pr\left(t_1^I | s_1^J\right) \propto \max_{K, \tilde{t}_1^K, \tilde{s}_1^K, \alpha_1^K} p\left(\alpha_1^K\right) \prod_{k=1}^{K} p\left(\tilde{t}_k | \tilde{s}_{\alpha_k}\right) \quad (5)$$

where the distortion model $p(\alpha_1^K)$ establishes the probability of a phrase alignment. Usually a first order model is used, assuming that the phrase-based alignment depends only on the distance of a phrase to the previous one [19].

There are different approaches for the parameter estimation. The first one corresponds to a direct learning of the parameters, in equation 4 or equation 5, from a sentence-aligned corpus using a maximum likelihood approach [25,

21]. The second one is heuristic, and tries to use a word-aligned corpus [26, 14]. These alignments can be obtained from single-word models [17] using the available public software GIZA++ [27]. The phrase-based models used in our project have been trained using the second approach.

VIII. ADAPTATION SYSTEM

If available, we make use of additional information, such as text closely related to the speech we are going to translate. This text can be written in the source language, in the target language or in both languages.

The SRM adaptation is performed via the SAPI interface [22]. SAPI adaptation uses specific calls to the SAPI interface. Specifically, we extract each word from the source adaptation data and we use it with the SAPI call *AddVocabulary* to extend the SRM vocabulary.

The MTM adaptation is hence performed as follows: first we train an additional source language model using the source language text, and, then, we train a second additional target language model using the target language text. Finally, we train additional statistical models using both source and target text.

These new models are then incorporated to the system using the loglinear framework. In this framework, each model needs a scaling factor parameter that is estimated by using a minimum error rate criterion [19]. A development corpus is needed for this propose. We can create these adaptation models from a corpus of labeled samples. They will have greater or lesser weight in the system according to the error rate configured for each model.

From a practical point of view, in a first step, in the system adaptation, we consider the choice of the subject on which the presentation will be taught, and what material we can use to improve the training process. The material that can be introduced is classified into three levels:

a) Texts of the slides: The speaker has to provide the slides, that are going to be used during the presentation, to the system. In this case, we used the Power Point format, so the application was programmed to be able to extract text from each slide.

b) Class Notes: Power Point application allows the speaker to introduce in each slide text notes on what is going to be explained. If these notes are introduced, the system will read and use them to guide the system. For this reason, we recommend to include them using a text as equal as possible to the one used in the presentation. For example, if the teacher starts the class telling "welcome to this speech", this sentence should be added in the first slide.

c) General information on the subject: The speaker can include the notes offered to the speech participants or students, reference books and other materials related to the subject. This third level of information has to be introduced to the system using plain text files.

This information may be supplied in the source language (Spanish) and in the target language (English). In addition, for levels a) and b), the system let us introduce this information in a bilingual format. That is, both the slides and the lecture notes can be translated and added into the system.

Additional information is used to adapt both modules of our system. The main adaption is performed in the machine translation module. New statistical models are trained and combined with the models described before using this new information and the log-linear approach [19].

A. Speech recognition module adaptation

1) Add exclusivity to the recognition in our application

One of the problems we encountered during the development of this project is the conflict created between the recognition commands connected to Windows and the ones related to our application. In other words, if we say a word that corresponds to a system command such as "start", then a conflict appears because it is recognized as a command and it is executed.

In order to solve this problem we have added the code shown in figure 6 in the developed application. It gives us exclusivity to our grammar and thus does not recognize any voice-recognition command in Windows Vista.

2) Add new words to the recognizer

In this sub-section we describe how we can add a new word recognizer from the application process. First, the application creates a token object. It assigns several constants and, then, a user lexicon object is created. Secondly, if the user wants to add a word, the application creates an object from the token and the user lexicon object. Finally, the application adds the new word to the user lexicon using the *AddPronuntation*. There are two ways to add new words to the dictionary of the speech recognition engine using the user interface (UI). One is making a call from the same application UI object (running again the screen setup), and the other one is to go directly to the option of "Speech Dictionary" IU recognition engine and select "add a new word". The code to add a new word or phrase into the SR engine in Delphi language is shown in figure 7.

First we added the elements that we need from the palette of elements of the development environment Delphi 7. These objects are SpSharedRecoContext and SpLexicon. They belong to the SAPI interface found in ActiveX. SpLexicon object is used with the *AddPronunciation* method. The way we used *AddPronunciation* method is shown in figure 8. The constant indicating the type of unknown word is *SPSUnknown*. In this case, the pronunciation of the word is added with phonemes.

The procedure in figure 7 let us add the words from the speaker's presentation notes to the user lexicon dictionary in order to improve the recognition. *GetPronunciations* method let us know the words added to the user lexicon.

```
10 var GramarFija:TOleEnum;
20 begin
30    GramarFija:=SGSExclusive;
40    SRGrammar.State:= GramarFija;
50 end;
```

Figure 6. Added code for having exclusivity.

```
10 Procedure AddPronunciation(const FileName:string;
             SpLexicon:TSpLexicon);
20 var
30   Ref,fic1,fic2,salida: textFile;
40   sAdap_Ref,FileNameSalida,fileWav,palabra:string;
50   SREF: TStringList;
60   l,posPalabra:integer;
70 begin
80   numRep:=0;sRef:=TStringList.Create;
90   assignFile(Ref,FileName); reset(Ref);
100  while not eof (Ref) do begin sRef.Free;
110   readSoundReferences(Ref,fileWav,sAdap_Ref,sRef);
120    posPalabra:=1;
130    repeat
140        palabra:=readFromStr(sAdap_Ref, posPalabra,
                    ' ,.:;/{}()\"+=');
150        if palabra <>'' then
160        // We will read every word of the file.
170        // txt (sAdap_Ref) and adding to the
180        // dictionary SR motor using the method
190        // AddPronunciation()
200        SpLexicon.AddPronunciation(palabra, 3082,
                      SPSUnknown, '');
210    until palabra='';
220   end;
230 end;
```

Figure 7. Code to add a new word into the SR engine.

```
10 SpObjectToken1.AutoConnect:=true;
20 SpLexicon1.AddPronunciation('new word',
                          3082, SPSUnknown,'');
```

Figure 8. *SpLexicon* object with the *AddPronunciation* method.

```
10 SpeechRecoContext.SetAdaptationData(
                       AdaptationString As String)
```

Figure 9. *setAdaptationData* with *SpeechRecoContext* method.

```
10  procedure  AdaptationData(const FileName: string;
        iteration: integer;
        SpSharedRecoContext: TSpSharedRecoContext);
30  var Ref,fic1,fic2,salida:textFile;
40      sAdap_Ref:string;
50      sRef: TStringList;
60      n,numRep:integer;
70      FileNameSalida,fileWav,srclang,trglang:string;
80      valorEdit:string;
90  begin
100   numRep:=0;
110   nivelFichTraza:=0;
120   sRef:=TStringList.Create;
130   assignFile(Ref,FileName); reset(Ref);
140   while not eof (Ref) do begin
150      sRef.Free;
160      readSoundReferences(Ref, fileWav, sAdap_Ref,
                                          sRef);
170      for n :=0 to iteration do begin
180         SpSharedRecoContext.SetAdaptationData(
                                       sAdap_Ref);
190      end;
200   end;
210 end;
```

Figure 10. Code to add a new phrase into the SR engine.

3) Adapt phrases to the recognizer

We utilized *setAdaptationData* method to improve the application. This method is used in the interface ISpeechRecoContext which recognizes different contexts. Generally, setAdaptationData method sends a data string to the voice recognition engine. It is often used to improve the recognition of words (or groups of unfamiliar words), by training the SR engine. Its structure is shown in figure 9.

The code used in the program to adapt different phrases is shown in figure 10.

B. Machine Translation Module Adaptation

In order to adapt our system to a specific topic, additional information must be supplied. As we have described at the beginning of this section, the information is structured in three levels: texts of the slides (level a), presentation notes (level b) and general information on the subject (level c). The following lines explain how we have added the statistical models in the MTM.

1) Specific matter source language model: A trigram language model that is trained using the source training text which has been supplied in level c. It is expressed in equation 6.

$$\Pr_{\mathrm{m}}(s_1^J) \propto \prod_{j=1}^{J} p\left(\mathrm{s_j}\middle|\mathrm{s_{j-2}^{j-1}}\right) \tag{6}$$

This model may seem redundant because it is also defined in the SRM. However, our experiments show that it helps the recognizer to have better performance.

2) Specific target language matter model: A trigram language model that is trained using the target training text which has been supplied in level c. It is given by equation 7.

$$\Pr_{\mathrm{m}}(t_1^I) \propto \prod_{i=1}^{I} p\left(\mathrm{t_i}\middle|\mathrm{t_{i-2}^{i-1}}\right) \tag{7}$$

This model is vital to guide the translator in selecting the most likely output.

3) Slides source language model: A bigram language model that is trained using the text from the PowerPoint file (levels a and b) in the source language.

$$\Pr_{\mathrm{s}}(s_1^J) \propto \prod_{j=1}^{J} p\left(\mathrm{s_j}\middle|\mathrm{s_{j-1}}\right) \tag{8}$$

It is similar to $\Pr_{\mathrm{m}}(s_1^J)$, but it is trained on texts that have high probability of being delivered (transparencies and presentation notes).

4) Slides target language model: A bigram language model that is trained using the text from the PowerPoint file (levels a and b) in the target language.

$$\Pr_{\mathrm{s}}(t_1^I) \propto \prod_{i=1}^{I} p(\mathrm{t_i}|\mathrm{t_{i-1}}) \tag{9}$$

It is similar to $\Pr_{\mathrm{m}}(t_1^I)$, but trained with slides and lecture notes.

5) Slides translation model: A phrase based translation model that is trained using the bilingual text from two PowerPoint files: the source language and the target language.

$$\Pr_s\left(t_1^I \middle| s_1^J\right) \propto \max_{K,\tilde{t}_1^K,\tilde{s}_1^K} \prod_{k=1}^{K} p(\tilde{t}_k \,|\tilde{s}_k) \quad (10)$$

In order to train this model we need a bilingual corpus (source and target languages). In our case we have used Spanish-English corpus. For this corpus the presenter has to create two PowerPoint files with the same information in each one of them. The number of sentences in each one of the slides or presentation notes should be equal. In our application, if the presenter loads the slides with different number of sentences, the system displays an error message. The presenter must indicate which file is in Spanish and which one is in English.

IX. SYSTEM EVALUATION

The system described in this paper was assessed through a series of experiments stressing the system in different situations and using three different speakers. The experiments were carried out in a scenario that reproduces the regular conditions of a university class.

The subject selected for the experiments was "Programming", which was taught in the first year of the "Technical Engineer in Telecommunications" degree (at the Higher Polytechnic School of Gandia, Polytechnic University of Valencia). This course describes the principles of C++ object-oriented programming language. This choice was motivated because this subject is being taught in several groups and has a lot of specific information in several languages that allow us to adapt the system. Specifically we used several programming books in both languages and the teacher notes of the subject in Spanish.

In the scenario, a teacher provided a 20 minutes class supported with projected slides and class notes which where beforehand translated into Spanish and English. The class was recorded in an empty room without students for the sake of comparing output results with the same background noise conditions. Sentences from the recording in Spanish were then segmented, transcribed and translated into English.

The sentences obtained where divided into two parts: the test corpus (made by 240 sentences) and the development corpus (made by 120 sentences). The sentences from the test corpus were also recorded later by two additional speakers. Table I represents the different quality features for each speaker.

The generic models of MTM were initially trained by the Europarl corpus [24]. Slides and class notes have been used to train the additional models of MTM. The developed corpus has been used to estimate the lambda parameters using the minimum error rate criteria.

A. Speech Recognition Module Evaluation

First, we performed a set of experiments to assess the SRM module individually. In this set of experiments, we analyzed the influence of the speaker in each type of environment, the type of computer used and the different adaptation methods applied to the SRM.

1) Speaker and speech rate

Three different speakers were used in the test phase. In all three cases the sentences were the same. The first one spoke spontaneously. The other two read a transcript of the class. One of these speakers made an adaptation to his pronunciation. Table I shows the most important features of the three speakers.

Table II shows the speech recognition performance comparison for three test speakers. It is obvious that the speaker adaptation capabilities are crucial to obtain good speech recognition rates. They can be used to analyze the words error rate recognized by the system (WER).

2) Used Hardware

During the development phase we noted that the type of computer used has a very important factor for proper operation of the SRM. In order to evaluate the influence of the hardware at this stage, we performed the test using different hardware equipment. Their details are shown in Table III. Table IV shows the WER obtained from different experiments. We can state that if we have a system with more memory and faster processing capacity, the recognition is improved significantly. The remaining of experiments carried out in this paper were performed using the server.

3) Type of adaptation in SRM

In section VIII we proposed different methods for the SRM adaptation. Its evaluation was performed through a series of experiments. We used the presentation slides and class notes in Spanish as the information provided for the adaptation. The results are shown in Table IV. When new vocabulary words (*AddPronunciation*) are added, the results are significantly improved. Statistical significance is calculated using paired bootstrap [29]. It is better than the baseline with a confidence of 99%. Otherwise, to retrain internal language models (*SetAdaptationData*) does not show any improvement. We have seen that if we repeat this process several times (10 and 100 times), we obtained a slight improvement. But if we use our language model, the results are considerably better. This leads us to the conclusion that the implementation of the SAPI *SetAdaptationData* method is not satisfactory.

B. Machine Translation Module Evaluation

In Table V, different adaptation mechanisms have been compared. In the baseline case, there was no adaptation. The SAPI adaptation mechanism uses specific calls to the SAPI interface. Specifically, we extract each word from the source slides and the class notes to extend the SRM vocabulary. To evaluate the MTM adaptation we considered two cases of sources of knowledge. The first one was only the slides, and the second one was the slides with class notes. Moreover, in each case we tested it just using the source language, and using both source and target language.

TABLE I. SYSTEM QUALITY FOR THREE SPEAKERS

	spontaneous speech	speaker adaptation	gender
speaker 1	yes	No	male
speaker 2	no	Yes	male
speaker 3	no	No	female

TABLE II. SPEECH RECOGNITION PERFORMANCE FOR THREE SPEAKERS

	Speech Recognition (WER)
speaker 1	30.75
speaker 2	15.38
speaker 3	34.5

TABLE III. FEATURES OF THE COMPUTERS USED IN EXPERIMENTS

Model	MacBook Pro	Server
Processor	Intel(R) Corel(TM)2 Duo	Intel® Core™2 Quad Processor Q9400
Processing Speed	2.40 GHz+2.40 GHz	4x2.66 GHz
Operating System	Windows Vista Enterprise (32 bits). Service Pack 1	Windows 7 Ultimate (32 bits)
Memory (RAM)	2,00 GB	4,00 GB

TABLE IV. WER OBTAINED FOR DIFFERENT METHODS OF ADAPTATION AS A FUNCTION OF TYPE OF COMPUTER

Adaptation method	MacBook Pro	Server
Base-line	20.2	17.5
AddPronunciation	19.3	16.7
AddPronunciation+SetAdaptationData (10 times)	19.2	16.8
AddPronunciation+SetAdaptationData (100 times)	19.0	16.4
AddPronunciation + external LM	18.3	15.3

TABLE V. ADAPTATION RESULTS FOR SPEAKER 1

	speech Recognition (WER)	Machine Translation (WER)	(BLEU)
Base line	17.5	54.2	34.8
+ SRM adaptation	16.5	53.8	35.1
+$\mathbf{Pr_m}$ (s_1^J)	15.3	53.3	35.6
+ $\mathbf{Pr_m}$ (t_1^I)	15.4	42.1	45.7
+ $\mathbf{Pr_m}$ (t_1^I)	9.7	48.4	40.1
+ $\mathbf{Pr_s}$ (t_1^I)	9.7	35.0	56.4
+ $\mathbf{Pr_s}$ ($t_1^I|s_1^J$)	9.6	34.8	56.5

In order to measure the quality of the translation machine, we analyzed both, the well-recognized words and the well translated, in WER and BLEU measures respectively. We can see in this table that there is an important difference between translation WER and recognition WER. These results depend on the type of task.

The experiments demonstrate how the use of an additional information source really improves significantly the overall results (by a 12%). Particularly, it is most improved when we make use of the class notes in both languages before we start the system. In this case, the accuracy rate increases by a 35%. To provide translations is obviously an extra effort for the teachers, but is often worth doing it when the number of foreign students in the class is high.

X. CONCLUSIONS AND FUTURE WORK

A real-time speech translation system specific to pedagogical environments has been presented in this paper. The main innovation of this work is the way in which additional sources of knowledge are used to improve the accuracy of the system, while remaining practical.

To train the system with other text sources related to the class helps very much the translation (even if they are not the exact notes to the slides). But, we should provide texts related to the same concepts developed in the speech (e.g. reference books of the topic that is going to be presented).

Finally we listed a series of benefits and drawbacks associated with this system. In contrast, we show that a powerful PC microprocessor is required in order to have a good tool performance. Moreover, in order to carry out the training a data collection process is necessary. This data collection process is somewhat laborious for the teacher. This is why we developed a tool to perform this process in real time. Only small breaks between phrases are needed to collect data. This was one of the main objectives, and it has been achieved successfully.

The system described in this work can be used in any pair of languages allowing their translation.

As future work we will improve the system by using models of confusion networks as interfaces between the automatic speech recognition and machine translation modules. Moreover, we are going to add a third language to the system in order to have a system with real-time translation to several languages.

ACKNOWLEDGMENT

This work has been partially supported by the Generalitat Valenciana (Project GV/2009/038) and by the Universidad Politécnica de Valencia.

REFERENCES

[1] Loof, J., Gollan, C., Hahn, S., Heigold, G., Hoffmeister, B., Plahl, C., Rybach, D., Schluter, R., and Ney, H.: The rwth 2007 tc-star evaluation system for europeanenglish and spanish. Interspeech, Antwerp, Belgium (2007). Pp. 2145–2148.

[2] Casacuberta, F., Ney, H., Och, F.J., Vidal, E., Vilar, J.M., Barrachina, S., Garca-Varea, I., Llorens, D., Martnez, C., Molau, S., Nevado, F., Pastor, M., Pic´o, D.,Sanchis, A., and Tillmann, C.: Some approaches to statistical and finite-state speechto-speech translation. Computer Speech and Language 18 (2004) 25–47.

[3] A. Canovas, J. Tomás, J. Lloret, M. García, Speech Translation Statistical System Using Multimodal Sources of Knowledge, The Fifth International Multi-Conference on Computing in the Global Information Technology (ICCGI 2010), Valencia (España), september 20–24, 2010.

[4] Lavie, A., et al. 2001. Architecture and Design Considerations in NESPOLE!: a Speech Translation System for E-Commerce Applications. In Proceedings of the Human Language Technology Conference (HLT 2001), San Diego, CA. (2001).

[5] Verbmobil website, at http://verbmobil.dfki.de/overview-us.html [Last access January 31, 2011]

[6] Hervé Blanchon, and Christian Boitet. Speech Translation for French within the C-STAR II Consortium and Future Perspectives. Sixth International Conference on Spoken Language Processing (ICSLP 2000). Beijing, China. October 16-20, 2000. Vol. 4/4. Pp. 412-417.

[7] Baylon Sppech Engine Website. http://babylon-speech-engine.fyxm.net/ [Last access January 31, 2011]]

[8] Nakamura, S., et al.: The ATR multi-lingual speech-tospeech translation system. IEEE Transactions on Speech and Audio Processing, Vol. 14, No. 2, (2006) Pp. 365–376.

[9] F. Casacuberta, D. Llorens, C. Martíınez, S. Molau, F. Nevado, H. Ney, M. Pastor, D. Picó, A. Sanchis, E. Vidal, and J. M. Vilar: Speech-to-speech Translation Based on Finite-State Transducers. Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, Salt Lake City, UH, (2001) Pp. 613–616.

[10] Zhang, R., Kikui, G., Yamamoto, H., Watanabe, T., Soong, F., Lo, W. K.: A Unied Approach in Speech-to-Speech Translation: Integrating Features of Speech Recognition and Machine Translation. The 20th International Conference on Computational Linguistics (COLING 2004), Geneve, Switzerland, August 23-27, (2004).

[11] Ney, H. Speech translation: Coupling of recognition and translation. In Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, Phoenix, AR (1999) Pp. 517-520.

[12] Brown, P., Chen, S., Pietra, V. D., Pietra, S.D., Keller, A., and Mercer, R.: Automatic speech recognition in machine translation. Computer Speech and Language 8. (1994) Pp. 177–187.

[13] J. Schroeder and P. Koehn: The University of Edinburgh System Description for IWSLT 2007, Proc. of the International Workshop on Spoken Language Translation, Trento, (2007).

[14] Koehn, P., Axelrod, A., Birch, A., Callison-Burch, C., Osborne, M., and Talbot, D.. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. International Workshop on Spoken Language Translation (IWSLT 2005), Pittsburgh, PA, USA, (2005).

[15] Koehn, P. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. Sixth Conference of the Association for Machine Translation in the Americas. Pp. 115-124. 2004.

[16] J.C. Amengual, J.M. Benedl, F. Casacuberta, A. Castellanos, V.M. Jimenez, D. Llorens, A. Marza, M. Pastor, F. Prat, E. Vidal, J. M. Vilar. The EuTrans-I speech translation system. Machine Translation. vol. 15, Pp. 75 –103, 2000.

[17] Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., and Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics 19 (1993) Pp. 263–311.

[18] Tomás, J., Lloret, J., and Casacuberta, F.: Phrase-based alignment models for statistical machine translation. In: Pattern Recognition and Image Analysis. Volume 3523 of Lecture Notes in Computer Science. Springer-Verlag (2005). Pp. 605–613.

[19] Och, F.J., Ney, H.: Discriminative training and maximum entropy models for statistical machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, PA (2002).

[20] Sphinnx Website. At http://cmusphinx.sourceforge.net/ [Last access January 31, 2011]]

[21] HTK Hidden Markov Model Toolkit home page. At http://htk.eng.cam.ac.uk/ [Last access January 31, 2011]

[22] Hao Shi, A.M.: Speech-enabled windows application using microsoft sapi. International Journal of Computer Science and Network Security 6 (2006) Pp. 33–37.

[23] Tomás, J., Vilar, J., and Casacuberta, F.: The ITI statistical machine translation system. Proceedings of the TC-Star Speech to Speech Translation Workshop, Barcelona, Spain (2006) Pp. 49–55.

[24] Tomás, J., Casacuberta, F.: Monotone statistical translation using word groups. Proceedings of the Machine Translation Summit VIII, Santiago, Spain (2001) Pp. 357-361.

[25] Marcu, D., Wong, W.: A Phrase-Based, Joint Probability Model for Statistical Machine Translation. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2002), Philadelphia, PA (2002) Pp. 6 -7.

[26] Zens, R., Och, F. J., Ney, H.: Phrase-Based Statistical Machine Translation. Lecture Notes in Computer Science, Volume 2479, Pp. 35-56. (2002)

[27] Och, F. J., Ney, H.: Improved statistical alignment models. Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (2000).

[28] Koehn, P.: Europarl: A parallel corpus for statistical machine translation. Proceedings of the Machine Translation Summit X, Phuket, Thailand (2005) Pp. 12-16.

[29] Bisani, M., Ney, H.: Bootstrap estimates for confidence intervals in asr performance evaluation. In: IEEE International Conference on Acustics, Speech, and Signal Processing. Volume 1., Montreal, Canada (2004) 409-412

Identifying Potentially Flawed Items in the Context of Small Sample IRT Analysis

Panagiotis Fotaris, Theodoros Mastoras, Ioannis Mavridis, and Athanasios Manitsaris
Department of Applied Informatics
University of Macedonia
Thessaloniki, Greece
{paf, mastoras, mavridis, manits}@uom.gr

***Abstract*—Although Classical Test Theory has been used by the measurement community for almost a century, Item Response Theory has become commonplace for educational assessment development, evaluation and refinement in recent decades. Its potential for improving test items as well as eliminating the ambiguous or misleading ones is substantial. However, in order to estimate its parameters and produce reliable results, IRT requires a large sample size of examinees, thus limiting its use to large-scale testing programs. Nevertheless, the accuracy of parameter estimates becomes of lesser importance when trying to detect items whose parameters exceed a threshold value. Under this consideration, the present study investigates the application of IRT-based assessment evaluation to small sample sizes through a series of simulations. Additionally, it introduces a set of quality indices, which exhibit the success rate of identifying potentially flawed items in a way that test developers without a significant statistical background can easily comprehend and utilize.**

Keywords - item response theory, computer aided assessment, item quality, educational measurement, learning assessment evaluation, e-learning, psychometrics.

I. INTRODUCTION

Following the recent advances in Information and Communications Technology (ICT) as well as the growing popularity of distance learning, an ever-increasing number of academic institutions worldwide have embraced the idea that the educational process can be greatly enhanced through the use of Computer Aided Assessment (CAA) tools [1][2][3][4]. Some of the benefits of these tools include the potential for generating rapid individualized feedback [5][6], the reduction of the marking load on staff [7], the ability to include multimedia elements in test items [8], the availability of administrative and statistical data [9], and, most importantly, the assessment of the examinees' knowledge [10].

Self-assessment tests, while commonly portrayed as the most popular technique to enhance learning [11] and evaluate the learning status of each examinee [12], have been criticized extensively on account of their perceived lack of reliability. Moreover, both research and experience show that a substantial number of test items (questions) are flawed at the initial stage of their development. As a result, test developers can expect nearly 50% of their items to fail to perform as intended, thus leading to unreliable results with respect to examinee performance [13]. It is, therefore, of the utmost importance to ensure that the individual test items are of the highest possible quality, since an inferior item could have an inordinately large effect on scores and consequently pose a serious threat to overall test effectiveness.

Among the dominant methodological theories in item evaluation using item response data are Classical Test Theory (CTT) [14] and Item Response Theory (IRT) [15]. The former is essentially a loose collection of techniques for analyzing test functionality, including but not limited to indices of score reliability, item difficulty, item discrimination, and the distribution of examinee responses across the available range of responses [16]. Many of these techniques were generated in the 19th and 20th centuries by Pearson, Spearman, Thurstone, and others [17]. CTT is built around the idea that the observed score an examinee attains on a test is a function of that examinee's "true score" and error. Although its relatively weak theoretical assumptions make CTT easy to apply in many testing situations [18], this approach has a number of well-documented shortcomings. These include (a) the use of item indices whose values depend on the particular group of examinees with which they are obtained, and (b) examinee ability estimates that depend on the particular choice of items selected for a test [19]. Additionally, CTT is not as likely to be as sensitive to items that discriminate differentially across different levels of ability (or achievement), it does not work as well when different examinees take different sets of items, and it is not as effective in identifying items that are statistically biased [20][21].

On the other hand, IRT is more theory-grounded and models the probabilistic distribution of the examinees' success at item level. As its name indicates, IRT primarily focuses on the item-level information in contrast to the CTT's primary focus on test-level information [22]. Based on nonlinear models that plot the measured latent variable and the item response, it enables independent estimation of item and person parameters and local estimation of measurement error [23]. The IRT framework encompasses a group of models, with the applicability of each model in a particular situation depending on the nature of the test items and the viability of different theoretical assumptions about them. Models that use a single ability to describe quantitative differences among examinees and among items are referred to as unidimensional (UIRT), whereas those for multiple abilities are called multidimensional (MUIRT).

For test items that are dichotomously or binary scored (i.e., having only two possible outcomes), there are three IRT models, known as three- [24], two-, and one-parameter logistic models [25]. IRT analysis yields three estimated

parameters for each item, α, b and c respectively. The α parameter, also known as item slope, is analogous to CTT's item-test biserial correlation and measures the discriminating power of the item, while the b parameter is an index of item difficulty; consequently, the latter increases in value as items become more difficult. In contrast to the p-value used in CTT, b is theoretically not dependent on the ability level of the sample of examinees tested. Finally, the c parameter, commonly called the guessing or the pseudo-guessing parameter, is defined as the probability of a very low-ability test-taker answering the item correctly [26]. All three parameters are present in the following equation called Item Response Function (IRF) that defines the three-parameter logistic model (3PL) for dichotomous data. IRF gives the probability of a correct response to item i by an examinee with ability θ. When displayed graphically, it is called the Item Response Curve (IRC).

$$P_i(\theta) \equiv P_i(X_i = 1 \mid \theta) = c_i + \frac{(1 - c_i)}{1 + e^{-Da_i(\theta - b_i)}},\ i = 1, 2, ..., n. \quad (1)$$

In Equation (1), X_i is the score for item i, with $X_i = 1$ for a correct response and $X_i = 0$ for an incorrect response. θ is the numerical value of the trait that reflects the examinee's level of ability, achievement, skill, or other major characteristic, which is measured by the test. α_i, b_i, and c_i are item parameters, and D is a scaling constant. In the two-parameter logistic model (2PL) the c parameter is fixed at a specific value rather than estimated, with both the α and c parameters fixed at specific values in the one-parameter logistic model (1PL).

Theoretically, IRT overcomes the circular dependency of CTT's item / person parameters and its models produce item statistics independent of examinee samples, and person statistics independent of the particular set of items administered [27]. This invariance property of IRT's item and person statistics has been widely accepted within the measurement community, resulting in the widespread application of IRT models to large-scale educational testing programs.

However, the aforementioned models have not proven popular outside these programs due to the complex nature of the item parameter estimation associated with them. To ensure that all IRT parameters are correctly estimated, every single item needs to be tested on a large number of examinees so as to define its properties [28]. Unless this condition is fulfilled, the benefits of using IRT might not be attained, i.e., the success of IRT applications requires a satisfactory fit between the model and the data. It is generally accepted that a minimum sample size of 20 items and 200 examinees is sufficient to fit the one parameter logistic model [29][30], while much larger sample sizes (e.g., 60 items and 1,000 examinees) are needed for the three parameter logistic IRT model [31].

Although a number of researchers have proposed smaller sample sizes than those specified above [32][33], it remains difficult for educators who teach small- or medium-sized classes to find a satisfying number of test-takers. So far, little research has been done to investigate whether the potential advantages of an IRT model can still be achieved in such an environment. Fotaris et al. [34] introduced a methodological and architectural framework for extending an LMS with IRT–based assessment test calibration. By defining a set of validity rules, test developers are able to set the acceptable limits for all IRT parameters before administering the test. The enhanced LMS subsequently applies these rules to the parameters produced by the IRT analysis in order to detect potentially flawed items and report them for reviewing. Since this system has only been used on a pilot basis, the present study is focused on exploring the benefits, limitations and accuracy of incorporating IRT analysis given limited sample sizes before its adoption in actual academic courses.

The next sections of this paper will introduce two types of quality indices and describe the simulation study design and process. The results of the study will be presented in the final section, together with a discussion about the impact of the examinees' ability distribution on the performance of the IRT analysis.

II. ASSESSMENT QUALITY INDICES

As previously mentioned, it is necessary to estimate parameters α, b, and c for each test item when using a three-parameter logistic model. This procedure, also called item calibration, is typically performed by software that employs joint maximum likelihood (JML), conditional maximum likelihood (CML), or marginal maximum likelihood (MML) methods [35]. Although its level of accuracy affects the success rate of potentially flawed items detection, the latter is not directly obtained from the item fit indices used in the various goodness-of-fit studies [36][37][38][39]. Therefore, the present paper introduces new indices to describe this exact success rate in a way that test developers without a profound statistical background will be able to fully comprehend and utilize.

Let τ_i denote the true value of one IRT parameter (α, b, or c) for item i, and $\hat{\tau}_i$ its corresponding estimate. Accordingly, let $A\{\tau_i\}$ denote the set of the assessment test's true parameter values, with $(A, \leq)$ being totally ordered, and $\hat{A}\{\hat{\tau}_i\}$ the set of its corresponding estimates, with $(\hat{A}, \leq)$ being totally ordered, as well. Sets N and $\hat{N}$ contain the indices i of the elements in A and $\hat{A}$ respectively.

$$N\{i : \tau_i \in A\},\quad i \in \{1, 2, ..., n\}, \quad (2)$$

$$\hat{N}\{i : \hat{\tau}_i \in \hat{A}\},\quad i \in \{1, 2, ..., n\}, \quad (3)$$

Thus, given a test with n dichotomous items, all of the aforementioned sets have the same cardinality n:

$$|\hat{A}| = |A| = |N| = |\hat{N}| = n. \quad (4)$$

Sets $N_{\leq(q)}$ and $\hat{N}_{\leq(q)}$ contain the indices *i* of items whose parameter values are equal to or lower than a threshold value *q*; they can be described as follows:

$$N_{\leq(q)} \{i : \tau_i \in A,\ \tau_i \leq q\},\quad i \in \{1,2,...,n\}, \tag{5}$$

$$\hat{N}_{\leq(q)} \{i : \hat{\tau}_i \in \hat{A},\ \hat{\tau}_i \leq q\},\quad i \in \{1,2,...,n\}, \tag{6}$$

with $N_{\leq(q)}$ referring to the true parameter values and $\hat{N}_{\leq(q)}$ to the estimates, respectively. In a similar manner, the sets containing the indices *i* of items whose parameter values are equal to or greater than a threshold value *q*, can be described thus:

$$N_{\geq(q)} \{i : \tau_i \in A,\ \tau_i \geq q\},\quad i \in \{1,2,...,n\}, \tag{7}$$

$$\hat{N}_{\geq(q)} \{i : \hat{\tau}_i \in \hat{A},\ \hat{\tau}_i \geq q\},\quad i \in \{1,2,...,n\}, \tag{8}$$

with $N_{\geq(q)}$ referring to the true parameter values, and $\hat{N}_{\geq(q)}$ to the estimates, respectively. Let $N_{L(r\%)}$ be a subset of *N* with its cardinality being a percentage (*r*%) of the cardinality of *A*. $N_{L(r\%)}$ contains the indices *i* of items whose parameter values belong to the lower set of $(A, \leq)$ (9). Likewise, let $\hat{N}_{L(r\%)}$ be a subset of $\hat{N}$ with its cardinality being a percentage (*r*%) of the cardinality of $\hat{A}$. $\hat{N}_{L(r\%)}$ contains the indices *i* of items whose parameter values belong to the lower set of $(\hat{A}, \leq)$ (10).

$$N_{L(r\%)} \{i : \tau_i \in A,\ \tau_j \in A,\ (\forall \tau_i, \tau_j)\left(i \in N_{L(r\%)} \wedge \tau_j \leq \tau_i \rightarrow j \in N_{L(r\%)}\right),\ \left|N_{L(r\%)}\right| = \left(|A| r\%\right)\},\quad i, j \in \{1,2,...,n\} \tag{9}$$

$$\hat{N}_{L(r\%)} \{i : \hat{\tau}_i \in \hat{A},\ \hat{\tau}_j \in \hat{A},\ (\forall \hat{\tau}_i, \hat{\tau}_j)\left(i \in \hat{N}_{L(r\%)} \wedge \hat{\tau}_j \leq \hat{\tau}_i \rightarrow j \in \hat{N}_{L(r\%)}\right),\ \left|\hat{N}_{L(r\%)}\right| = \left(\left|\hat{A}\right| r\%\right)\},\quad i, j \in \{1,2,...,n\} \tag{10}$$

The same logic is used to denote the sets $N_{U(r\%)}$ and $\hat{N}_{U(r\%)}$, with the only difference being that they contain the indices *i* of items whose parameter values belong to the upper set of $(A, \leq)$ and $(\hat{A}, \leq)$, respectively.

$$N_{U(r\%)} \{i : \tau_i \in A,\ \tau_j \in A,\ (\forall \tau_i, \tau_j)\left(i \in N_{U(r\%)} \wedge \tau_j \geq \tau_i \rightarrow j \in N_{U(r\%)}\right),\ \left|N_{U(r\%)}\right| = \left(|A| r\%\right)\},\quad i, j \in \{1,2,...,n\} \tag{11}$$

$$\hat{N}_{U(r\%)} \{i : \hat{\tau}_i \in \hat{A},\ \hat{\tau}_j \in \hat{A},\ (\forall \hat{\tau}_i, \hat{\tau}_j)\left(i \in \hat{N}_{U(r\%)} \wedge \hat{\tau}_j \geq \hat{\tau}_i \rightarrow j \in \hat{N}_{U(r\%)}\right),\ \left|\hat{N}_{U(r\%)}\right| = \left(\left|\hat{A}\right| r\%\right)\},\quad i, j \in \{1,2,...,n\} \tag{12}$$

Finally, the new quality indices $g_{\leq(q)}$, $g_{\geq(q)}$, $g_{L(r\%)}$, $g_{U(r\%)}$, are defined as follows:

$$g_{\leq(q)} = \begin{cases} \sqrt{\dfrac{\left|\left(\hat{N}_{\leq(q)} \cap N_{\leq(q)}\right)\right|^2}{\left|\hat{N}_{\leq(q)}\right|\left|N_{\leq(q)}\right|}}, & \text{for } \left|\hat{N}_{\leq(q)}\right| \neq 0 \wedge \left|N_{\leq(q)}\right| \neq 0 \\ 0, & \text{for } \left|\hat{N}_{\leq(q)}\right| = 0 \oplus \left|N_{\leq(q)}\right| = 0 \\ \text{undefined}, & \text{for } \left|\hat{N}_{\leq(q)}\right| = 0 \wedge \left|N_{\leq(q)}\right| = 0 \end{cases} \tag{13}$$

$$g_{\geq(q)} = \begin{cases} \sqrt{\dfrac{\left|\left(\hat{N}_{\geq(q)} \cap N_{\geq(q)}\right)\right|^2}{\left|\hat{N}_{\geq(q)}\right|\left|N_{\geq(q)}\right|}}, & \text{for } \left|\hat{N}_{\geq(q)}\right| \neq 0 \wedge \left|N_{\geq(q)}\right| \neq 0 \\ 0, & \text{for } \left|\hat{N}_{\geq(q)}\right| = 0 \oplus \left|N_{\geq(q)}\right| = 0 \\ \text{undefined}, & \text{for } \left|\hat{N}_{\geq(q)}\right| = 0 \wedge \left|N_{\geq(q)}\right| = 0 \end{cases} \tag{14}$$

$$g_{L(r\%)} = \frac{\left|\left(\hat{N}_{L(r\%)} \cap N_{L(r\%)}\right)\right|}{\left|\hat{N}_{L(r\%)}\right|},\ \text{for } \left|\hat{N}_{L(r\%)}\right| \neq 0 \tag{15}$$

$$g_{U(r\%)} = \frac{\left|\left(\hat{N}_{U(r\%)} \cap N_{U(r\%)}\right)\right|}{\left|\hat{N}_{U(r\%)}\right|},\ \text{for } \left|\hat{N}_{U(r\%)}\right| \neq 0 \tag{16}$$

Their values are in the range [0, 1], with 1 indicating a successful parameter estimation by the IRT model. The following example demonstrates how those indices can be utilized to evaluate the success rate when attempting to detect potentially flawed items. Let us assume that the "true" values of the *b* parameter of a 100-item assessment test are *Ab* = {-1.085, 0.802, 0.101, -2.112, 0.03… -0.449}, and the estimates derived from the IRT analysis are $\hat{A}b$ = { -1.15959, 0.936761, 0.190408, -2.47219, 0.094549, …, -0.14912}. When setting *q* = 1.7 as the threshold value in order to identify the questions with the highest degree of difficulty, the indices *i* of the true difficult questions are included in the set $Nb_{\geq(1.7)}$ = {37, 89, 49, 24}. Similarly, the set $\hat{N}b_{\geq(1.7)}$ = {89, 49, 24, 74} contains the indices *i* of the difficult questions as estimated by the IRT model. Compared to the true data, the IRT estimation falsely

identified question no. 74 as difficult, and failed to flag question no. 37 for revision. According to (14), the $gb_{\geq(1.7)}$ index can be calculated as follows: $gb_{\geq(1.7)} = \sqrt{3^2 / 4 \times 4} = \sqrt{9/16} = \sqrt{0.5625} = 0.75$, which shows that the IRT estimation detected 75% of the actual difficult questions, with only 75% of those estimates being genuinely difficult.

Given the same 100-item assessment test, sets $Nb_{U(10\%)}$ = {42, 53, 74, 81, 21, 72, 37, 89, 49, 24}, and $\hat{N}b_{U(10\%)}$ = {86, 54, 72, 21, 37, 81, 89, 49, 24, 74} contain the indices *i* of the true and the estimated 10 hardest questions, respectively. A comparison of the two sets reveals that the IRT estimation failed to detect questions no. 42 and no. 53, i.e., only 8 out of the 10 flagged questions were correctly identified as difficult. As a result, the $gb_{U(10\%)}$ index value that denotes this exact success rate is 0.8.

III. SIMULATION STUDY DESIGN

This study explored the application of IRT analysis in a very specific context – assessing whether it would produce accurate results in the detection of potentially flawed items given limited sample sizes. For this reason, it was necessary to compare the item parameter estimates with the true item parameter values. However, since the latter cannot be known *a priori*, the investigation was carried out by using simulated data. Taking into account the vast number of features provided by the freeware computer program WinGen2 [40] (including support for various IRT models, generation of IRT model parameter values from various distributions, and an intuitive and user friendly interface), the latter was used to generate the true item parameter values from various distributions. Additionally, WinGen2 begot data sets of realistic dichotomous item response data, which were subsequently sent to the open-source IRT analysis tool ICL (IRT Command Language) [41] for IRT parameter estimation. In fact, ICL is a set of IRT estimation functions (ETIRM) embedded into a fully-featured programming language called TCL ("tickle") [42] that allow relatively complex operations.

Both true and estimated parameter values were employed by the quality indices $g_{\leq(q)}$, $g_{\geq(q)}$, $g_{L(r\%)}$, and $g_{U(r\%)}$, as a means of calculating the success rate when attempting to detect potentially flawed items. Considering that the latter are described by low discrimination (α), high / low difficulty (*b*), or high guessing (*c*), the acceptable threshold values of each item's IRT parameters were set to $\alpha \geq 0.5$, $-1.7 \leq b \leq 1.7$, and $c \leq 0.2$ [26][43]. As a result, the quality indices used in this study were $g\alpha_{\leq(0.5)}$, $gb_{\leq(-1.7)}$, $gb_{\geq(1.7)}$, and $gb_{\geq(0.2)}$. The threshold-independent indices $g\alpha_{L(10\%)}$, $gb_{L(10\%)}$, $gb_{U(10\%)}$, and $gc_{U(10\%)}$, were also employed as an alternative way to evaluate the goodness-of-fit of the IRT analysis.

The simulation study employed 9 test lengths (*i* = 20, 30, 40, 50, 60, 70, 80, 90, and 100 items), 50 sample sizes (*N* = 20 to 1,000 examinees, with a step of 20), and 3 groups of examinees with ability levels of differing distributional characteristics. The first group comprised of medium ability level examinees, while the majority of examinees in the second and third group were of low and high ability level, respectively (Fig. 1).

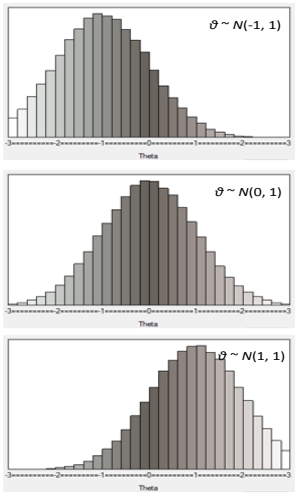


Figure 1. The ability level distributions for the three groups of the simulation study.

The values for the item difficulty parameter (*b*) were randomly selected from a standard normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$, $b \sim N(0, 1)$. As for the values for the discrimination (α) and the guessing (*c*) parameters, these were randomly sampled from a lognormal distribution $\alpha \sim log\text{-}N(0, 0.5)$, and a beta distribution $c \sim B(2, 19)$, respectively. The value for each quality index was computed as the average over 10 iterations of the IRT analysis, with all parameters being re-estimated every time. Since a highly accurate estimation of those indices was not the primary goal of the present study, the aforementioned amount of iterations was considered adequate. The total number of performed IRT analyses was 13,500 (3 groups x 10 iterations x 50 sample sizes x 9 test lengths).

IV. SIMULATION STUDY PROCESS

The methodology used in the simulation study can be divided into three steps (Fig. 2).

Step 1: WinGen2 simulated data sets of realistic ability parameters θ for each one of the three groups of 1,000 examinees. These values were randomly selected from a standard normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$, $b \sim N(0, 1)$ for the first group (Fig. 3), $\mu = -1$, $\sigma = 1$, $b \sim N(0, 1)$ for the second one, and $\mu = 1$, $\sigma = 1$, $b \sim N(0, 1)$ for the third. The generated ability data, serving now as the "true" data, were then stored in a text file comprised of 1,000 lines and 2 columns, with the first column containing each examinee's index number and the second its corresponding true value of the ability θ. Subsequently, WinGen2 generated a test of 100 random test items whose item parameter values α, *b*, and *c* were randomly sampled from the following distributions: $\alpha \sim \log\text{-}N(0, 0.5)$, $b \sim N(0, 1)$, $c \sim B(2, 19)$ (Fig. 4). The resulting data were stored in a 100 lines long and 6 columns wide text file. Its first column contained each item's index number and the last three ones

the true values of the *α*, *b*, and *c* parameters, respectively. Finally, WinGen2 produced and saved the results of the previous test in a new text file with a size of 1000 lines (number of examinees) x 100 columns (total test items) (Fig. 5). Since the IRT model used in the simulation was dichotomous, the only possible answers to the test items were 0 (wrong) and 1 (correct).

Step 2: With the "true" values of the IRT parameters already at hand, the next step was to create their corresponding estimates. For that purpose, a custom Visual Basic program executed ICL for a total number of 1,350 times (3 groups x 50 sample sizes x 9 test lengths), feeding it each time with the simulated response data from WinGen2. Accordingly, ICL performed dichotomous 3PL IRT analysis on the aforementioned data using the following script:

```
output -no_print↵
allocate_items_dist <items>↵
read_examinees out_examinees_<items>.dat
{@11 <items>i1}↵
starting_values_dichotomous↵
set fileID [open out_items__<items>
_examinees_<examinees>_results.par w] ↵
write_item_param_channel $fileID -format
%.5e↵
close $fileID↵
release_items_dist↵
```

Step 3: The resulting file ("`out_items_<items>_examinees_<examinees>_results.par`") was finally sent to Microsoft Excel in order to calculate the quality indices ($g\alpha_{\leq(0.5)}$, $gb_{\leq(-1.7)}$, $gb_{\geq(1.7)}$, $gc_{\geq(0.2)}$, $g\alpha_{L(10\%)}$, $gb_{L(10\%)}$, $gb_{U(10\%)}$, $gc_{U(10\%)}$) for that particular pair of items and examinees.

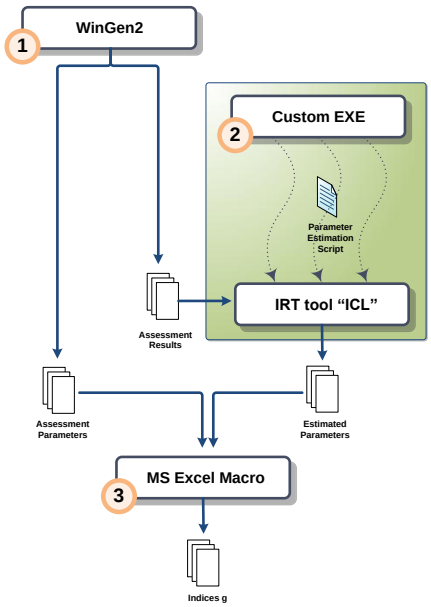


Figure 2. Simulation study process

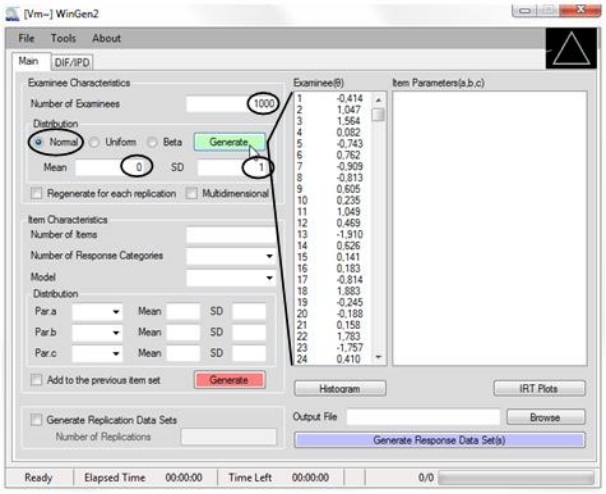


Figure 3. Generating the first group of 1,000 examinees, $\theta \sim N(0, 1)$.

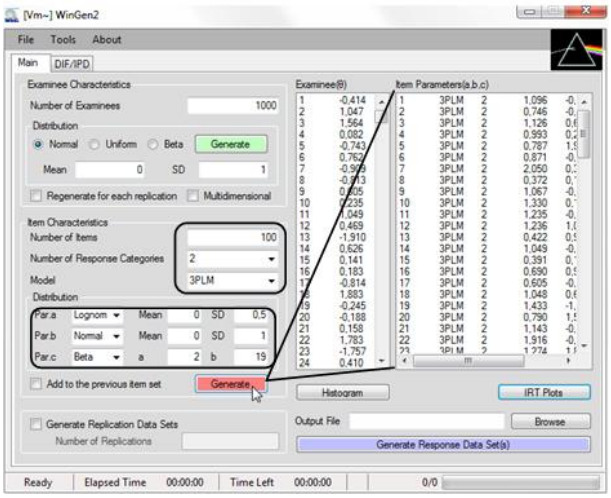


Figure 4. Generating 100 test items, $\alpha \sim log\text{-}N(0, 0.5)$, $b \sim N(0, 1)$, $c \sim B(2, 19)$.

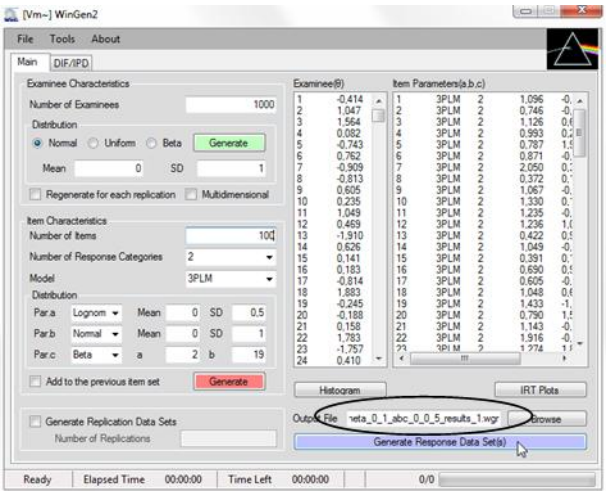


Figure 5. Generating the results of 1,000 examinees in a 100 item test.

α <= (0.5)

Examinees \ Items	20	40	60	80	100
40	0,17	0,25	0,34	0,33	0,30
80	0,18	0,30	0,45	0,42	0,41
120	0,37	0,45	0,51	0,48	0,45
160	0,25	0,49	0,53	0,51	0,50
200	0,21	0,46	0,54	0,51	0,50
240	0,20	0,50	0,55	0,56	0,53
280	0,19	0,51	0,61	0,56	0,54
320	0,21	0,50	0,59	0,56	0,54
360	0,32	0,52	0,60	0,57	0,56
400	0,26	0,55	0,62	0,58	0,57
440	0,23	0,55	0,61	0,60	0,57
480	0,24	0,57	0,63	0,60	0,58
520	0,27	0,58	0,62	0,59	0,57
560	0,48	0,57	0,61	0,60	0,59
600	0,49	0,57	0,60	0,60	0,58
640	0,49	0,58	0,61	0,59	0,56
680	0,55	0,59	0,62	0,59	0,57
720	0,60	0,59	0,63	0,60	0,57
760	0,61	0,63	0,62	0,59	0,56
800	0,62	0,61	0,61	0,58	0,57
840	0,55	0,59	0,60	0,58	0,56
880	0,68	0,61	0,60	0,58	0,56
920	0,66	0,60	0,62	0,57	0,56
960	0,74	0,62	0,61	0,58	0,58
1000	0,81	0,64	0,62	0,60	0,58

$g\alpha_{\leq(0.5)}$

α Lower (10%)

Examinees \ Items	20	40	60	80	100
40	0,25	0,23	0,25	0,29	0,29
80	0,30	0,38	0,43	0,44	0,40
120	0,45	0,45	0,48	0,46	0,50
160	0,40	0,53	0,52	0,53	0,50
200	0,45	0,45	0,52	0,50	0,49
240	0,45	0,53	0,53	0,51	0,56
280	0,45	0,58	0,50	0,60	0,64
320	0,50	0,58	0,57	0,63	0,62
360	0,55	0,63	0,63	0,64	0,65
400	0,55	0,63	0,63	0,64	0,64
440	0,50	0,65	0,65	0,65	0,68
480	0,55	0,60	0,60	0,64	0,65
520	0,55	0,58	0,62	0,63	0,68
560	0,65	0,63	0,65	0,65	0,66
600	0,65	0,65	0,67	0,69	0,69
640	0,65	0,65	0,67	0,65	0,70
680	0,70	0,75	0,68	0,69	0,73
720	0,65	0,70	0,68	0,65	0,70
760	0,65	0,70	0,68	0,70	0,73
800	0,70	0,68	0,63	0,70	0,73
840	0,75	0,70	0,72	0,69	0,73
880	0,75	0,73	0,70	0,71	0,75
920	0,70	0,73	0,70	0,70	0,73
960	0,75	0,73	0,70	0,70	0,73
1000	0,80	0,68	0,70	0,71	0,75

$g\alpha_{L(10\%)}$

b <= (-1.7)

Examinees \ Items	20	40	60	80	100
40	0,42	0,37	0,36	0,53	0,52
80	0,56	0,47	0,55	0,66	0,70
120	0,50	0,53	0,58	0,73	0,67
160	0,65	0,55	0,54	0,66	0,64
200	0,65	0,57	0,57	0,70	0,71
240	0,68	0,61	0,63	0,69	0,69
280	0,50	0,64	0,62	0,60	0,63
320	0,65	0,53	0,74	0,78	0,65
360	0,60	0,57	0,60	0,75	0,68
400	0,65	0,59	0,58	0,75	0,68
440	0,50	0,59	0,60	0,77	0,70
480	0,50	0,58	0,58	0,75	0,76
520	0,36	0,64	0,73	0,79	0,79
560	0,50	0,56	0,68	0,76	0,78
600	0,50	0,73	0,68	0,78	0,76
640	0,50	0,63	0,64	0,76	0,74
680	0,50	0,73	0,64	0,77	0,77
720	0,50	0,73	0,68	0,79	0,77
760	0,50	0,73	0,65	0,76	0,76
800	0,50	0,71	0,63	0,76	0,74
840	0,50	0,71	0,68	0,76	0,76
880	0,50	0,71	0,68	0,79	0,77
920	0,36	0,58	0,70	0,79	0,77
960	0,32	0,58	0,65	0,78	0,78
1000	0,46	0,58	0,61	0,78	0,78

$gb_{\leq(-1.7)}$

b Lower (10%)

Examinees \ Items	20	40	60	80	100
40	0,45	0,58	0,60	0,63	0,61
80	0,60	0,70	0,63	0,69	0,70
120	0,80	0,75	0,67	0,75	0,73
160	0,80	0,78	0,70	0,73	0,74
200	0,85	0,80	0,70	0,78	0,75
240	0,85	0,75	0,68	0,76	0,76
280	0,85	0,75	0,68	0,80	0,78
320	0,80	0,78	0,68	0,80	0,77
360	0,80	0,75	0,72	0,78	0,77
400	0,80	0,75	0,72	0,78	0,79
440	0,80	0,83	0,70	0,75	0,77
480	0,75	0,80	0,72	0,80	0,81
520	0,75	0,83	0,70	0,76	0,78
560	0,80	0,78	0,72	0,79	0,80
600	0,80	0,80	0,73	0,79	0,80
640	0,80	0,80	0,73	0,80	0,80
680	0,80	0,75	0,70	0,80	0,79
720	0,75	0,75	0,70	0,81	0,81
760	0,75	0,73	0,72	0,80	0,80
800	0,80	0,73	0,70	0,81	0,80
840	0,75	0,73	0,68	0,79	0,80
880	0,75	0,73	0,70	0,79	0,80
920	0,75	0,73	0,67	0,79	0,79
960	0,70	0,75	0,68	0,76	0,78
1000	0,75	0,78	0,67	0,76	0,79

$gb_{L(10\%)}$

b >= (1.7)

Examinees \ Items	20	40	60	80	100
40	0,25	0,37	0,33	0,41	0,41
80	0,34	0,56	0,40	0,56	0,56
120	0,43	0,61	0,41	0,60	0,60
160	0,47	0,44	0,43	0,55	0,58
200	0,71	0,63	0,47	0,63	0,63
240	0,47	0,44	0,48	0,62	0,66
280	0,47	0,55	0,40	0,65	0,67
320	0,43	0,42	0,37	0,63	0,65
360	0,57	0,59	0,46	0,71	0,69
400	0,57	0,46	0,51	0,64	0,72
440	0,85	0,56	0,48	0,69	0,72
480	0,85	0,59	0,52	0,74	0,75
520	0,85	0,62	0,60	0,78	0,75
560	0,85	0,59	0,51	0,77	0,76
600	0,85	0,58	0,57	0,76	0,77
640	0,85	0,71	0,46	0,71	0,72
680	0,85	0,66	0,52	0,76	0,79
720	0,85	0,66	0,60	0,78	0,81
760	0,85	0,69	0,62	0,80	0,83
800	0,85	0,66	0,62	0,83	0,83
840	0,85	0,66	0,62	0,85	0,82
880	0,85	0,66	0,60	0,79	0,84
920	0,85	0,69	0,60	0,83	0,83
960	0,85	0,82	0,62	0,82	0,82
1000	0,85	0,81	0,70	0,79	0,83

$gb_{\geq(1.7)}$

b Upper (10%)

Examinees \ Items	20	40	60	80	100
40	0,55	0,53	0,48	0,55	0,57
80	0,50	0,63	0,48	0,56	0,59
120	0,60	0,63	0,62	0,66	0,64
160	0,65	0,68	0,65	0,69	0,67
200	0,60	0,68	0,65	0,68	0,66
240	0,60	0,75	0,68	0,69	0,67
280	0,65	0,73	0,67	0,73	0,71
320	0,60	0,78	0,63	0,71	0,67
360	0,65	0,78	0,63	0,71	0,72
400	0,65	0,75	0,70	0,73	0,73
440	0,70	0,75	0,72	0,76	0,77
480	0,70	0,78	0,77	0,78	0,80
520	0,65	0,75	0,77	0,78	0,78
560	0,65	0,78	0,77	0,78	0,79
600	0,70	0,80	0,75	0,78	0,80
640	0,70	0,80	0,77	0,75	0,82
680	0,70	0,78	0,77	0,76	0,82
720	0,70	0,80	0,80	0,76	0,79
760	0,60	0,83	0,78	0,78	0,79
800	0,70	0,80	0,80	0,78	0,80
840	0,65	0,83	0,80	0,83	0,82
880	0,65	0,83	0,83	0,78	0,80
920	0,60	0,80	0,78	0,80	0,81
960	0,65	0,83	0,78	0,79	0,79
1000	0,65	0,83	0,78	0,80	0,81

$gb_{U(10\%)}$

c >= (0.2)

Examinees \ Items	20	40	60	80	100
40	0,26	0,30	0,27	0,26	0,26
80	0,31	0,28	0,27	0,26	0,26
120	0,25	0,26	0,27	0,27	0,28
160	0,30	0,28	0,31	0,29	0,28
200	0,28	0,29	0,28	0,28	0,28
240	0,33	0,28	0,30	0,30	0,32
280	0,35	0,29	0,34	0,33	0,32
320	0,28	0,27	0,32	0,31	0,30
360	0,32	0,33	0,34	0,32	0,32
400	0,34	0,35	0,32	0,31	0,32
440	0,31	0,31	0,33	0,33	0,32
480	0,30	0,30	0,31	0,31	0,30
520	0,30	0,30	0,32	0,34	0,29
560	0,28	0,31	0,34	0,32	0,31
600	0,27	0,31	0,35	0,34	0,35
640	0,28	0,32	0,37	0,33	0,33
680	0,29	0,30	0,36	0,35	0,34
720	0,31	0,27	0,33	0,34	0,32
760	0,34	0,29	0,34	0,33	0,33
800	0,30	0,27	0,35	0,30	0,31
840	0,34	0,24	0,34	0,30	0,30
880	0,23	0,36	0,36	0,32	0,32
920	0,24	0,36	0,35	0,30	0,32
960	0,10	0,37	0,33	0,29	0,30
1000	0,10	0,30	0,32	0,32	0,28

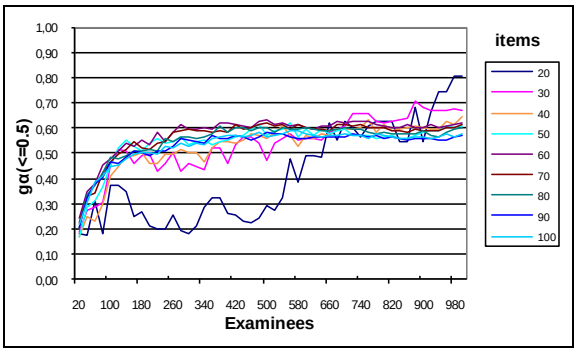
$gc_{\geq(0.2)}$

c Upper (10%)

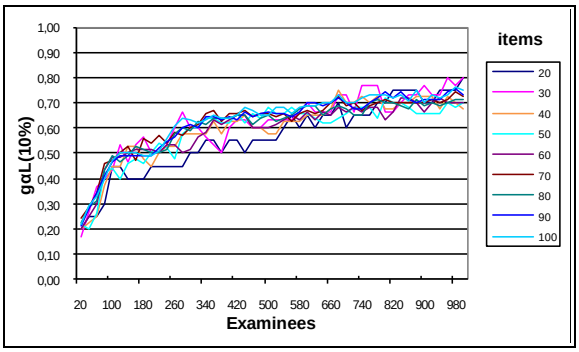
Examinees \ Items	20	40	60	80	100
40	0,20	0,13	0,12	0,15	0,16
80	0,25	0,25	0,20	0,16	0,18
120	0,20	0,25	0,18	0,21	0,20
160	0,30	0,25	0,27	0,25	0,25
200	0,30	0,28	0,28	0,20	0,18
240	0,25	0,30	0,28	0,25	0,24
280	0,15	0,30	0,32	0,29	0,29
320	0,10	0,28	0,32	0,26	0,28
360	0,15	0,18	0,27	0,30	0,26
400	0,25	0,20	0,33	0,30	0,27
440	0,20	0,25	0,30	0,30	0,28
480	0,20	0,28	0,27	0,26	0,25
520	0,10	0,20	0,25	0,24	0,25
560	0,20	0,25	0,22	0,24	0,26
600	0,20	0,20	0,28	0,25	0,24
640	0,20	0,15	0,32	0,24	0,26
680	0,10	0,18	0,27	0,24	0,26
720	0,20	0,18	0,28	0,25	0,25
760	0,20	0,13	0,35	0,25	0,22
800	0,20	0,23	0,28	0,25	0,23
840	0,20	0,20	0,33	0,26	0,26
880	0,20	0,18	0,35	0,30	0,26
920	0,10	0,25	0,30	0,25	0,28
960	0,05	0,23	0,30	0,25	0,28
1000	0,10	0,23	0,30	0,26	0,25

$gc_{U(10\%)}$

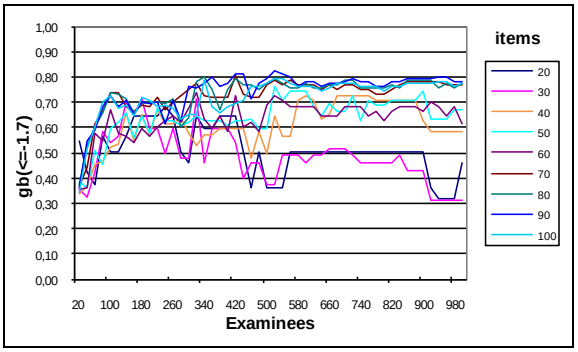
Figure 6. Simulation results for the group of examinees with ability parameters $\theta \sim N(0, 1)$.



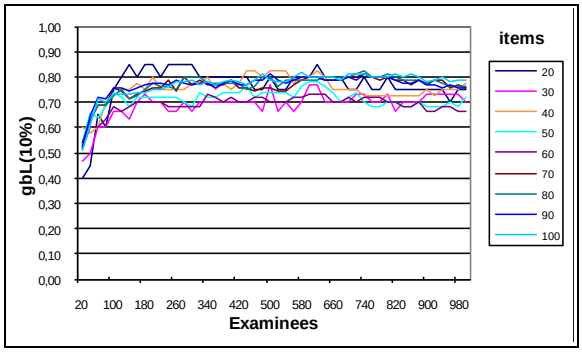
$g\alpha_{\leq(0.5)}$



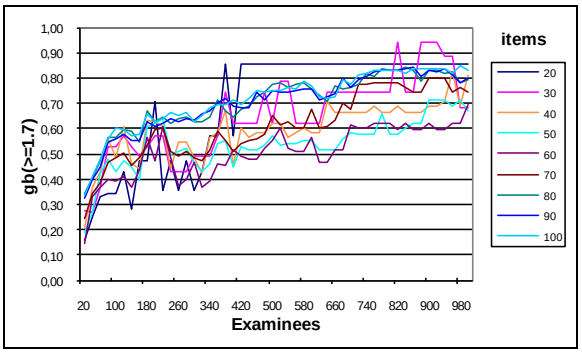
$g\alpha_{L(10\%)}$



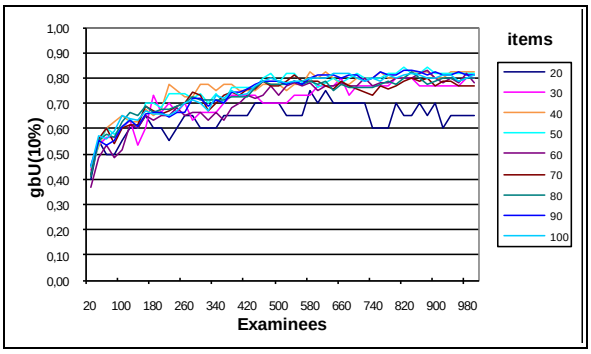
$gb_{\leq(-1.7)}$



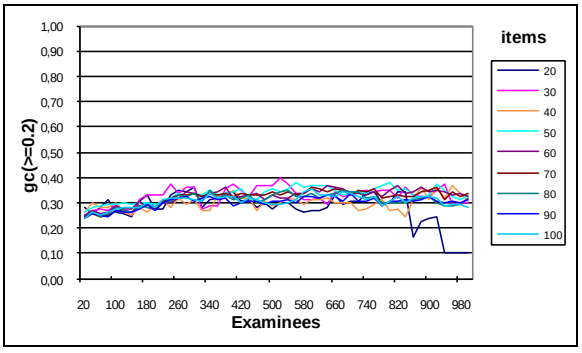
$gb_{L(10\%)}$



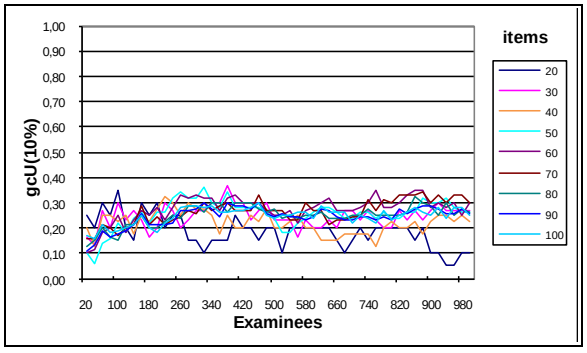
$gb_{\geq(1.7)}$



$gb_{U(10\%)}$

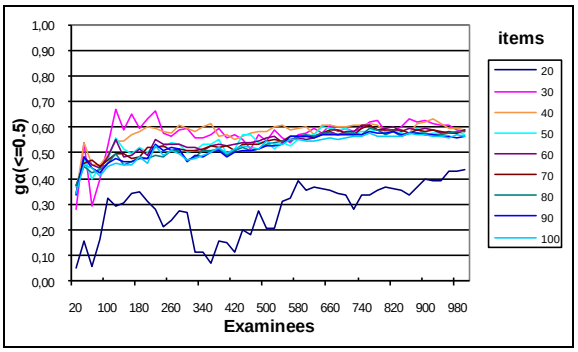


$gc_{\geq(0.2)}$

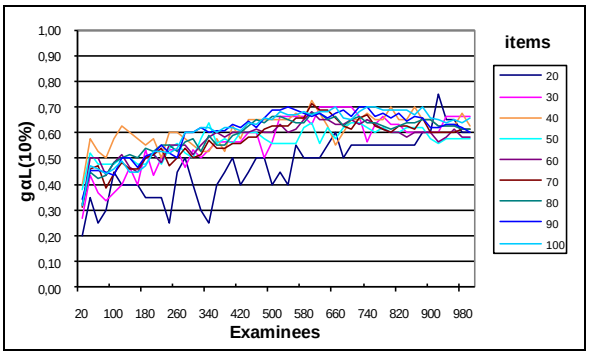


$gc_{U(10\%)}$

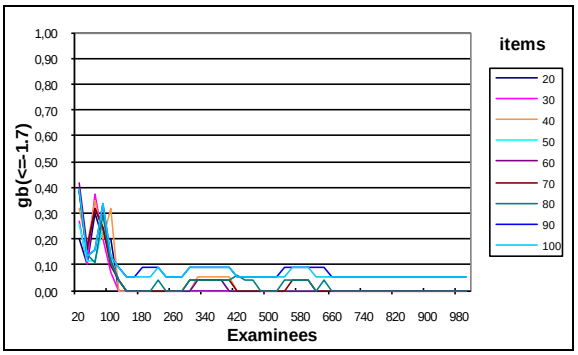
Figure 7. Simulation results graphs for the group of examinees with medium ability parameters, $\theta \sim N(0, 1)$.



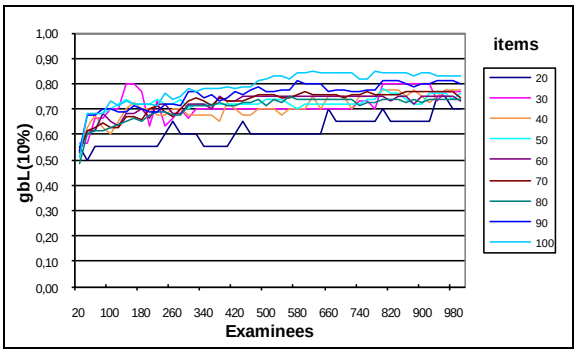
$g\alpha_{\leq(0.5)}$



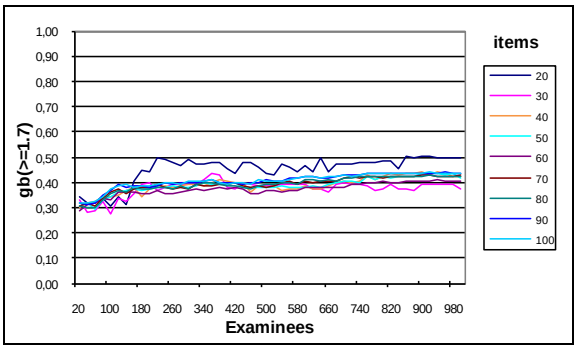
$g\alpha_{L(10\%)}$



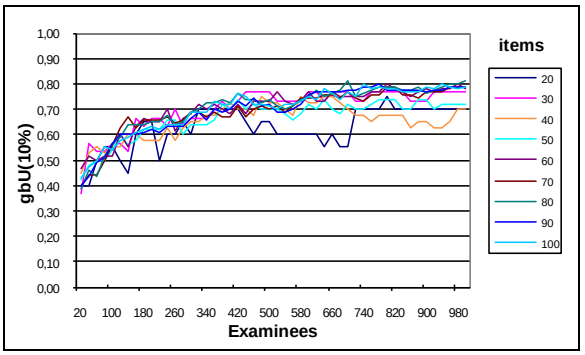
$gb_{\leq(-1.7)}$



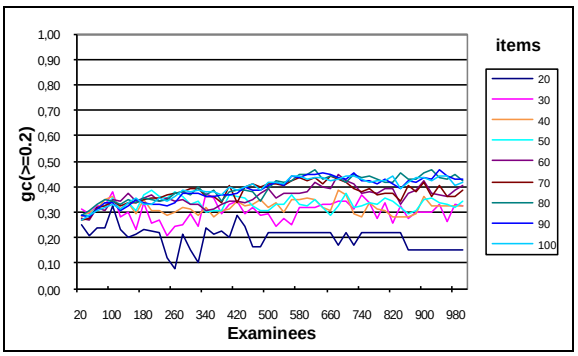
$gb_{L(10\%)}$



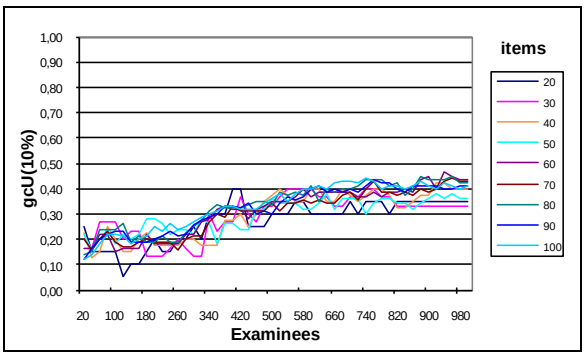
$gb_{\geq(1.7)}$



$gb_{U(10\%)}$

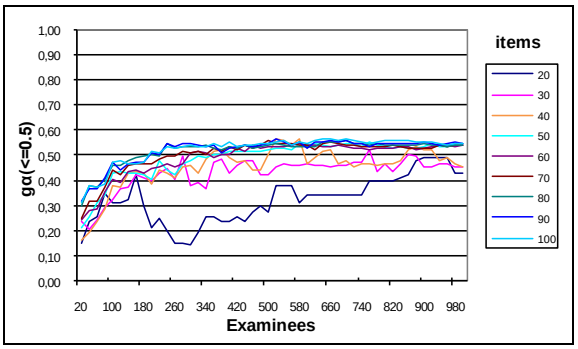


$gc_{\geq(0.2)}$

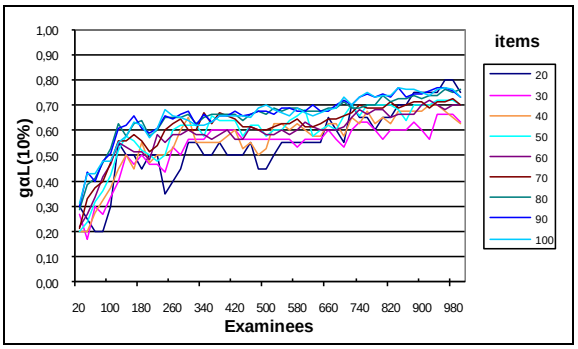


$gc_{U(10\%)}$

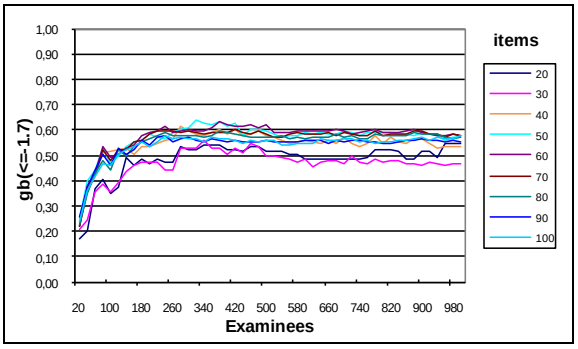
Figure 8. Simulation results graphs for the group of examinees with low ability parameters, $\theta \sim N(-1, 1)$.



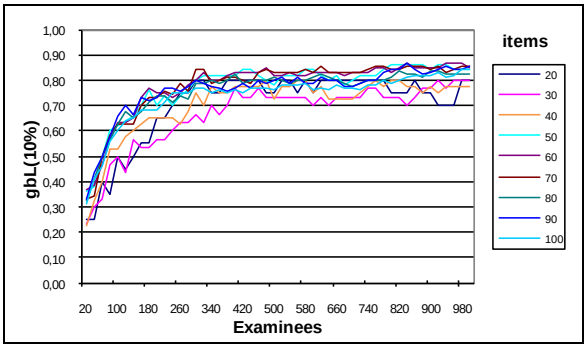
$g\alpha_{\leq(0.5)}$



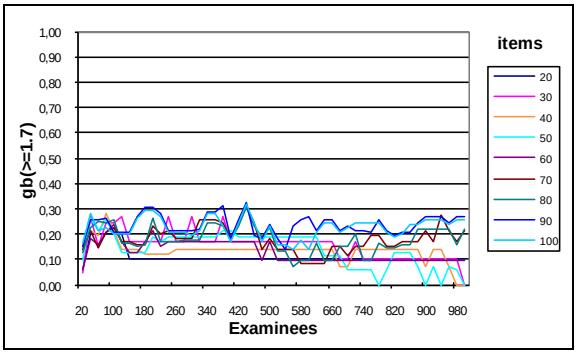
$g\alpha_{L(10\%)}$



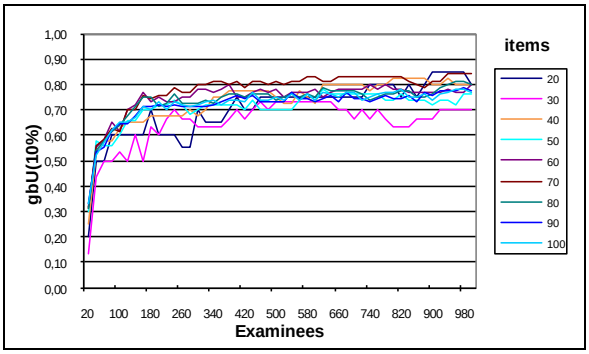
$gb_{\leq(-1.7)}$



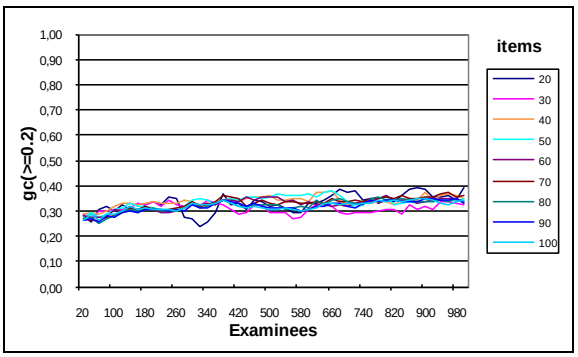
$gb_{L(10\%)}$



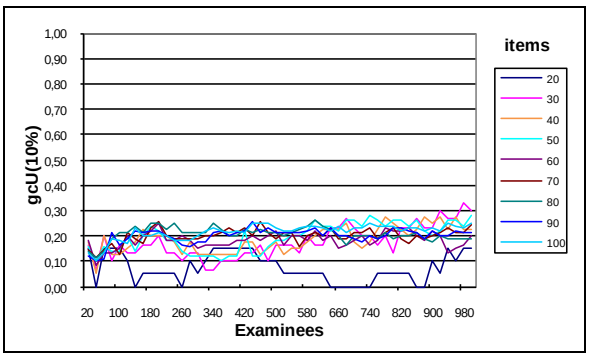
$gb_{\geq(1.7)}$



$gb_{U(10\%)}$



$gc_{\geq(0.2)}$



$gc_{U(10\%)}$

Figure 9. Simulation results graphs for the group of examinees with high ability parameters, $\theta \sim N(1, 1)$.

V. RESULTS - DISCUSSION

Indices $g\alpha_{\leq(0.5)}$, $gb_{\leq(-1.7)}$, $gb_{\geq(1.7)}$, $gc_{\geq(0.2)}$, $g\alpha_{L(10\%)}$, $gb_{L(10\%)}$, $gb_{U(10\%)}$, and $gc_{U(10\%)}$ can be practically treated as indicators of the IRT analysis success rate. For instance, according to the first group simulation (examinees with ability parameters $\theta \sim N(0,1)$), the value for index $gb_{L(10\%)}$ in a sample of 30 items and 160 examinees is 0.70. This can be interpreted as follows: when performing an IRT analysis to detect the 3 lowest difficulty items (i.e., 10% of the 30 items), only 2 of the results (i.e., 66% ≈ 70%) will be among the actual items with the lowest difficulty.

In the same manner, index $gb_{\geq(1.7)}$ receives the value of 0.71 in a sample of 80 items and 360 examinees (Fig. 6). In practice this means that, if the IRT analysis returns 5 items when asked to identify which ones have the highest difficulty level ($b \geq 1.7$), only 4 of them (71% of the 5 items = 3.55 ≈ 4) will, in fact, be among the ones with the highest difficulty.

As can be seen in Fig. 10a, the best fit between estimated and actual values for parameter b in a 100-item assessment test is achieved when the ability level of the examinees is medium, i.e., $\theta \sim N(0, 1)$. The measured Root Mean Square Error (RMSE) is considerably larger in the case of examinees with high ability levels $\theta \sim N(1, 1)$, and increases even further when the majority of examinees are of low ability ($\theta \sim N(-1, 1)$). Nevertheless, the probability of an item identified by the IRT analysis as being very difficult to be among the actual items with the highest degree of difficulty remains virtually the same for all three cases, regardless of the examinees' ability level (Fig. 10b).

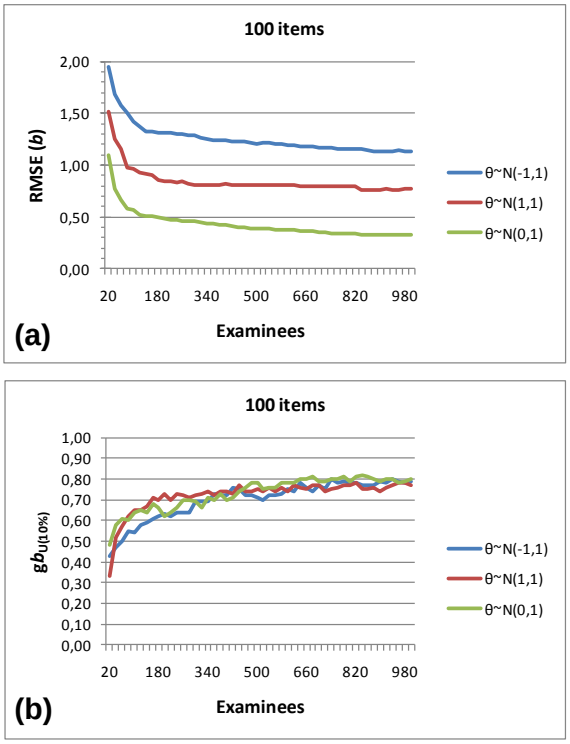


Figure 10. A comparison between (a) the fit index Root Mean Square Error (RMSE), and (b) the proposed quality index $gb_{U(10\%)}$ for the b parameter.

Consequently, despite the fact that the estimated values for parameter b are cohort-dependent, the ranking of those values is independent of cohort type and remains unchanged. Based on the above observations, it can be concluded that the proposed indices represent a practical and reliable means of assessing the quality of the IRT analysis results.

According to Fig. 8, IRT analysis fails to produce satisfactory results when attempting to detect items with low level of difficulty ($b \leq -1.7$) in the case of the low-ability cohort ($\theta \sim N(-1, 1)$), i.e., the success rate denoted by index $gb_{\leq(-1.7)}$ is considerably low. However, this finding was to be expected, since most low ability examinees may experience considerable problems when trying to answer all questions, including low difficulty ones. In spite of this, index $gb_{L(10\%)}$ appears unaffected by the cohort's ability level and, consequently, does not change. The explanation lies in the fact that there is still a match in the order of the actual and their corresponding estimated parameters, despite the poor item fit caused by small sample sizes.

Similarly, IRT analysis produces poor results when attempting to detect items with a high level of difficulty ($b \geq 1.7$) in the case of the high-ability cohort ($\theta \sim N(1, 1)$), i.e., the success rate provided by index $gb_{\geq(1.7)}$ is low (Fig. 9). This finding was equally unsurprising since most high ability examinees may answer most, perhaps all, questions correctly regardless of their actual difficulty level. Once again, the corresponding index $gb_{U(10\%)}$ appears unaffected by the cohort's ability level and does not change.

Overall, the IRT analysis results seemed unaffected by the different distributions associated with examinees' ability levels; the only exception to this rule was a decrease in the performance of indices $gb_{\leq(-1.7)}$ and $gb_{\geq(1.7)}$ for groups with $\theta \sim N(-1, 1)$ and $\theta \sim N(1, 1)$, respectively. This, in conjunction with the fact that indices $g\alpha_{L(10\%)}$, $gb_{L(10\%)}$, $gb_{U(10\%)}$, and $gc_{U(10\%)}$ perform better than $g\alpha_{\leq(0.5)}$, $gb_{\leq(-1.7)}$, $gb_{\geq(1.7)}$, and $gc_{\geq(0.2)}$ suggests that it would be best to base the methods used to detect flawed items not on specific parameter values, but on ranges near boundaries.

Finally, the values displayed in Fig. 6 are largely dependent on the distributions of both θ and the item parameters as generated by WinGen2. These parameters were specifically selected with the aim of simulating realistic item response data so as to produce reliable results. However, further experiments exceeding the scope of the present study have revealed that an increase in the mean of the distribution of the guessing parameter c would also result in a considerable decline in performance.

VI. CONCLUSIONS AND FUTURE WORK

Even though research focused on IRT sample size effects suggests that more than 1,000 examinees are needed to obtain accurate results when using the 3PL model [44], the simulated data depicted in Fig. 6 show that a sample of only 100 examinees can produce relatively satisfying results ($gb_{L(10\%)} = 0.7$) when trying to detect defective items with a low level of difficulty ($gb_{L(10\%)}$). In cases of assessments with more items, this limit can be lowered further to 60

examinees. Attempts to detect items with a high level of difficulty ($gb_{U(10\%)}$) have proven equally encouraging, with success rates exceeding 70% for sample sizes of 260 examinees and above.

Nevertheless, as the number of examinees is reduced from 200 to 100 and finally down to 20, the success rate of potentially flawed items detection drops dramatically (Fig. 7, 8, 9). Since all indices perform rather poorly for a sample of 50 examinees (< 30%), it becomes obvious that a smaller sample will produce even less adequate results. Therefore, addressing the scenario of a set comprised of less than 20 examinees deemed unnecessary for the purpose of the present study.

As expected given the small size of the samples described above, parameter *b* appears to be estimated more accurately than parameters *α* and *c*. However, in order to achieve an acceptable degree of success when trying to detect items with low discrimination (parameter *α*) the required number of examinees exceeds the 660 mark. In addition, even a sample size as large as 1,000 seems insufficient to produce reliable estimates for parameter *c*. These findings indicate that in academic contexts where the sample size can roughly exceed 120 examinees, IRT-based assessment could in practice be used only to identify inappropriate items according to their level of difficulty. In any other case, this procedure has the risk of losing the measurement precision, as well as other advantages of IRT. Further investigation into the impact of sample size on IRT assessment is needed and will be undertaken by performing a greater number of simulations in the near future.

REFERENCES

[1] P. Fotaris, T. Mastoras, I. Mavridis, and A. Manitsaris, "Performance Evaluation of the Small Sample Dichotomous IRT Analysis in Assessment Calibration," Proc. Fifth International Multi-conference on Computing in the Global Information Technology ICCGI 2010, Valencia, Spain, Sep. 2010, pp. 214-219.

[2] V. S. Anantatmula and M. Stankosky, "KM criteria for different types of organisations," International Journal of Knowledge and Learning, vol. 4, no. 1, 2008, pp. 18-35, doi:10.1504/IJKL.2008.019735.

[3] H. Rego, T. Moreira, F. Garcia, and E. Morales, "Metadata and knowledge management driven web-based learning information system," International Journal of Technology Enhanced Learning, vol. 1, no. 3, 2009, pp. 215-228, doi:10.1504/IJTEL.2009.024868.

[4] S. Virtanen, "Increasing the self-study effort of higher education engineering students with an online learning platform," International Journal of Knowledge and Learning, vol. 4, no. 6, 2009, pp. 527-538, doi:10.1504/IJKL.2008.022886.

[5] G. Baggott and R. Rayne, "Learning Support for Mature, Part-time, Evening Students: Providing Feedback via Frequent, Computer-Based Assessments," Proc. Fifth International Computer Assisted Assessment Conference, Loughborough University, Jul. 2001, pp. 9-20.

[6] J. Dalziel, "Enhancing Web-Based Learning with Computer Assisted Assessment: Pedagogical and Technical Considerations," Proc. Fifth International Computer Assisted Assessment Conference, Loughborough University, Jul. 2001, pp. 99-107.

[7] P. Davies, "Computer Aided Assessment MUST be more than Multiple-Choice Tests for it to be Academically Credible?" Proc. Fifth International Computer Assisted Assessment Conference, Loughborough University, Jul. 2001, pp. 143-150.

[8] G. Lambert, "What is Computer Aided Assessment and how can I use it in my teaching," 2004, [online], Available: http://www.canterbury.ac.uk/Support/learning-teaching-enhancement-unit/Resources/Documents/BriefingNotes/Blackboard.pdf, [Accessed 15th January 2011].

[9] U. P. Singh and M. R. de Villiers, "Establishing the current extent and nature of usage of Online Assessment Tools in Computing-related Departments at South African Tertiary Institutions," Proc. SACLA 2010, Zebra Country Lodge, Jun. 2010.

[10] G. Brown, J. Bull, and M. Pendlebury, Assessing Student Learning in Higher Education, London: Routledge, 1997.

[11] D. Boud, Enhancing learning through self assessment, London: Rutledge, 1995.

[12] J. Wood and M. Burrow, "Formative Assessment in Engineering Using "TRIADS" Software," Proc. of the Sixth International Computer Assisted Assessment Conference, Loughborough University, 2002, pp.369-380.

[13] T. M. Haladyna, Developing and Validating Multiple-Choice Test Items, 2nd ed., Mahwah, NJ: Lawrence Erlbaum Associates, 1999.

[14] SCOREPAK: Item Analysis, unpublished.

[15] F. M. Lord, Applications of Item Response Theory to Practical Testing Problems, Hillsdale, NJ: Lawrence Erlbaum Associates, 1980.

[16] R. E. Bennett and D. H. Gitomer, "Transforming K–12 assessment: Integrating accountability testing, formative assessment, and professional support," in Educational assessment in the 21st century, C. Wyatt-Smith and J. Cumming, Eds. New York: Springer, 2009, pp. 43–61.

[17] C. Spearman, "General intelligence: Objectively determined and measured," American Journal of Psychology, vol. 15, 1904, pp. 201–293.

[18] R. K. Hambleton and R. W. Jones, "Comparison of Classical Test Theory and Item Response Theory and their Applications to Test Development," Educational Measurement: Issues and Practices, vol. 12, 1993 pp. 38-46, doi:10.1111/j.1745-3992.1993.tb00543.x

[19] R. K. Hambleton, "Principles and selected applications of item response theory," in Educational measurement, 3rd ed., R. L. Linn, Ed., New York: Macmillan, 1989, pp. 147-200.

[20] R. K. Hambleton and H. Swaminathan, Item Response Theory: Principles and Applications, Boston, MA: Kluwer-Nijhoff Publishing, 1987.

[21] C. B. Schmeiser and C. J. Welch, "Test Development," in Educational Measurement 4th ed., in R. .L. Brennan, ed. Westport, CT: Praeger Publishers, 2006.

[22] X. Fan, "Item response theory and classical test theory: An empirical comparison of their item/person parameters," Educational and Psychological Measurement, vol. 58, 1998, pp. 357–381.

[23] Š. Progar and G. Sočan, "An empirical comparison of Item Response Theory and Classical Test Theory," Horizons of Psychology, vol. 17, no. 3, 2008, pp. 5–24.

[24] A. Birnbaum, "Some Latent Trait Models and their Use in Inferring an Examinee's Ability,", in Statistical Theories of Mental Test Scores, F. M. Lord and M. R. Novick, Eds. Reading: Addison-Wesley, 1968.

[25] G. Rasch, Probabilistic models for some intelligence and attainment tests, Copenhagen, Denmark: Danmarks Paedogogische Institut, 1960.

[26] F. B. Baker, Item Response Theory: Parameter Estimation Techniques, New York:Marcel Dekker, 1992, doi:10.2307/2532822

[27] R. K. Hambleton, H. Swaminathan, and H. J. Rogers, Fundamentals of item response theory. Newbury Park, CA: Sage, 1991.

[28] X. Lin, H. Chen, R. Mather, and H. Fletcher, "Adaptive Assessment – A Practice of Classification on Small-Size Training Sets," Proc. Society for Information Technology & Teacher Education International Conference (SITE 2009), Chesapeake, VA: AACE., Mar. 2009, pp. 3168-3181.

[29] S. M. Downing, "Item response theory: applications of modern test theory in medical education," Medical Education, vol. 37, no. 8, 2003, pp. 739-745, doi:10.1046/j.1365-2923.2003.01587.x.

[30] B. D. Wright and M. H. Stone, Best Test Design, Chicago: MESA Press, 1979.

[31] M. D. Reckase, "Unifactor latent trait models applied to multi-factor tests: Results and implications," Journal of Educational Statistics, vol. 4, 1979, pp. 207-230.

[32] B. B. Reeve and P. Fayer, "Applying item response theory modelling for evaluating questionnaire item and scale properties," in Assessing quality of life in clinical trials, 2nd ed., P. Fayers and R. Hays, Eds. Oxford: Oxford University Press, 2005.

[33] J. M. Linacre, "Sample size and item calibration stability," Rasch Measurement Transactions, vol. 37 no. 4, 1994, p. 328.

[34] P. Fotaris, T. Mastoras, I. Mavridis and A. Manitsaris, "Extending LMS to Support IRT-Based Assessment Test Calibration," in Technology Enhanced Learning. Quality of Teaching and Educational Reform, M. D. Lytras et al., Eds. vol. 73, Berlin Heidelberg: Springer, pp. 534-543.

[35] W. M. Yen and A. R. Fitzpatrick, "Item Response Theory," in Educational Measurement 4th ed., in R. .L. Brennan, ed. Westport, CT: Praeger Publishers, 2006.

[36] M. Orlando and D. Thissen, "New item fit indices for dichotomous item response theory models," Applied Psychological Measurement, vol. 24, 2000, pp. 50-64.

[37] M. Orlando and D. Thissen, "Further investigation of the performance of S-X2: An item fit index for use with dichotomous item response theory models," Applied Psychological Measurement, vol. 27, 2003, pp. 289-298, doi:10.1177/0146621603027004004.

[38] W. M. Yen, "Using simulation results to choose a latent trait model," Applied Psychological Measurement, vol. 5, 1981, pp. 245-262, doi:10.1177/014662168100500212.

[39] R. L. McKinley and C. N. Mills, "A comparison of several goodness-of-fit statistics," Applied Psychological Measurement, vol. 9, 1985, pp. 49-57, doi:10.1177/014662168500900105.

[40] K. T. Han, "WinGen: Windows Software That Generates Item Response Theory Parameters and Item Responses," Applied Psychological Measurement, vol. 31, no.5, Sept. 2007, pp. 457-459, doi: 10.1177/0146621607299271.

[41] B. A. Hanson, IRT Command Language (ICL), unpublished.

[42] B. B. Welch, K. Jones, and J. Hobbs, Practical programming in Tcl and Tk, 4th ed., Upper Saddle River, NJ: Prentice Hall, 2003.

[43] R. Flaugher, "Item Pools," in Computerized Adaptive Testing: A Primer, 2nd ed., H. Wainer, Ed. Mahwah, NJ: Lawrence Erlbaum Associates, 2000.

[44] H. Swaminathan and J. A. Gifford, "Estimation of Parameters in the Three-parameter Latent Trait Model," in New Horizons in Testing, D. J. Weiss, Ed. New York: Academic Press, 1983.

A Ubiquitous System for Secure Management of Critical Rescue Operations

Suleyman Kondakci
Faculty of Engineering & Computer Sciences
Izmir University of Economics,
Izmir, Turkey
Email: suleyman.kondakci@ieu.edu.tr

***Abstract*—Application of real world sensor networks to critical rescue and aid operations is still a challenging research and development area for scientists and engineers. A ubiquitous monitoring, aid, and rescue management system, its requirements, architecture, communication protocols, algorithms, and design details are presented here. The system is designed for use by various operational purposes, organizations, individuals, and operation teams, especially by critical rescue and aid forces. With its light weight and secure communication abilities, it can be used in a variety of critical operations, ranging from disaster management to anti–terror operations. The proposed system consists of three interoperating subsystems; ubiquitous agents, intermediate message passing agents, and command control centers. The paper considers first the basic requirements and design of the overall system, and then presents a prototype implementation consisting of the three subsystems.**

***Keywords*-WSN application; cryptography; security protocol;ubiquitous system.**

I. INTRODUCTION

Historically, it has been witnessed that controlling natural disasters threatening human life has not been considered as series as it should deserve. Especially, populations do steadily proliferate in underdeveloped countries, which makes it even harder to overcome critical situations. This is mainly due to the limitation in economical resources and lack of technologies needed to safely manage critical operations. Especially in such countries, developing inexpensive systems needed to ubiquitously monitor and manage subjects is a challenge area for many researchers. It is still an open problem to design adequate mobile systems needed to rapidly identify critical situations and communicate related data with management points in order to efficiently manage dynamically growing emergency situations.

By going through the history of disaster outbreaks throughout the world, numerous reasons can be listed for employing science and technology as one of the primary aids in the management of disasters and rescuing of human lives. Thus, there is a growing demand for developing effective measures to manage critical public safety and disaster management operations, environmental monitoring, and various object tracking operations. In order to elaborate the management of such critical events, we have designed a ubiquitous monitoring system (UBIMOS), [1], which monitors real-time environmental conditions (gas, smoke, humidity, temperature), locations of subjects (e.g., humans, animals, and system) as well as their vital conditions known as heart rate, oxyhemoglobin saturation [2], thermal stress factors such as the body temperature, [3]. Vital human conditions are noninvasively measured by wearable sensors. As the main unit, a wearable micro-computer collects and processes sensor data, and transmits the sensor data to a remote center for the coordination of necessary emergency operations. An important contribution of UBIMOS is such that the results gained by its prototype implementation will allow us to refine assumptions and requirements made when designing hardware, software, protocols and mechanisms for critical ubiquitous operations.

As will be detailed later, one of the major application areas of UBIMOS is the secure wireless sensor network (WSN) operation, e.g., management of anti-terror operations. Related to this, we have incorporated a secure communication protocol for secure authentication of the nodes and for cryptographic data exchange between the nodes. There is an increase in terrorist actions undertaken in several countries. UBIMOS also aims at mitigating the potential damage from such terrorist attacks by two ways: (i) by equipping public transportation vehicles and other terrorist targets with the early detection and warning units of UBIMOS, and (ii) by equipping anti–terror team members with wearable UBIMOS units that monitor and exchange data about locations, environmental conditions, and vital body conditions of the team members located at the field of operation. Terrorism, policies, and anti–terrorism legislation issues have been seriously considered throughout th world, and a vast amount of organizations, e.g., Council of Europe Convention on the Prevention of Terrorism [4].

A. Outline of the Paper

In the following, Section II gives a brief overview of related work, Section III outlines the major requirements of UBIMOS, Section IV presents the overall architecture and subsystems of the presented system, Section V details the structure and operation modes of the UBIMOS subsystems. Section VI deals with the construction of UBIMOS domains, their configurations, routing algorithm, Section VII considers

secure communication and structure of the transport protocol. Section VIII reviews the prototype implementation of UBIMOS, Section IX concludes the paper.

II. Related Work

The work presented here contains the application of several new topic areas such as WSNs, topology control and routing in WSNs, security, interconnection between IP and sensor networks, programming, and embedded sensor hardware design and integration. WSNs [5], as the fundamental communication technology of the UBIMOS project, have a broad application spectrum with an enormous variety of designs [6]. A location aware WSN architecture called Disaster Aid Network dealing with real-time patient localization is presented in [7]. Another rescue related work considering the application of WSNs for fire rescue operations together with its requirements and challenges is given in [8].

In recent years, ubiquitous computing has become very popular as it delivers truly useful solutions in pervasive application domains. Especially, WSNs, mobile wireless technologies based on General Packet Radio Service (GPRS), Digital RF, WLAN 802.15.4, and Global Positioning System (GPS) are becoming today's must-have technologies for a variety of ubiquitous applications.

As already known, security threats [9], [10] to WSNs and countermeasures against those threats are of many faceted. Implementing security in small hand-held devices is a challenging task, because of the strengthened requirements such as the requirement for extreme low power consumption, high–mobility, and real–time operations. There exist a number of hardware and software design proposals for providing effective communication approaches. For security critical operations add on security functionality is a must. Adding security functionality to existing communication protocols (e.g., the IEEE 802.15.4 and TCP/IP protocol suits) is a demanding issue, and can even become a speed burden on real-time operations. In conjunction with this we have designed a secure transport protocol, based on the RSEP protocol [11], for use by UBIMOS's secure communications. An architecture for secure communication in mobile wireless networks is presented in [12]. An implementation and analysis of a lightweight cryptographic algorithm suitable for WSNs is presented in [13], and a reliable synchronous transport protocol for wireless image sensor networks is considered in [14].

A typical wireless sensor node has limited protection against radio jamming. The situation becomes worse if energy-efficient jamming can be achieved by exploiting knowledge of the data link layer. Encrypting the packets may help to prevent the jammer from taking actions based on the content of the packets, but the temporal arrangement of the packets induced by the nature of the protocol might unravel patterns that the jammer can take advantage of, even when the packets are encrypted. Several jamming attacks that allow the jammer to jam S-MAC, LMAC, and B-MAC are discussed in [15], where the algorithms are described in detail and simulated for the analysis of energy efficiency. Another work dealing with the denial of sleep attacks is presented in [16].

Another important aspect of WSNs is the power consumption [17]. In order to reduce the power consumption in sensor nodes, the nodes must go into a sleep mode during idle periods. Adaptive approaches to the solution of power consumption problems are important, which provide dynamic relocations of sensors and resizing of sensor networks. To achieve higher scalability and adaptability, a network architecture composed of self-organizing entities is presented in [18].

Design of energy-efficient wireless sensor networks with censoring and on-off sensors is considered in [19]. Recall that censoring is a statistical analysis method used for reliability testing of systems. A framework for the study of power consumption and bit error rate performance of non-coherent impulse radio ultra wideband correlation receivers conforming to the IEEE 802.15.3a is considered in [20]. The work from [21] examines the performance of differential positioning using both the GPS and GLONASS satellite systems for vehicle positioning. Another work, [22], addresses the problem of GPS signal tracking processes in low signal-to-noise ratio (SNR) and multi–path interference environments.

Routing [23] is a relatively more energy consuming process in WSNs, which is also a challenging research area growing with a great interest. Two different energy– and security–aware routing approaches for the real-time communication in WSNs are presented in [24] and [25]. Finally, an extensive work dealing with the estimation of mobile user's trajectory in mobile wireless networks is presented in [26].

III. Requirements

UBIMOS is mainly a domain-specific communication infrastructure designed to exchange critical event information for the management of critical operations. Exchange of the critical event data is performed via ubiquitous agents (UAs) and other agents having higher communication capabilities, which are called intermediate message passing nodes (IMPs). The management of the critical operations is coordinated by command control centers (3Cs). These centers also provide functionalities for long-range (RF and satellite) communications among different UBIMOS domains. Obviously, the UBIMOS infrastructure contains three different types of subsystems each with different technical and procedural capabilities, which require consistent policies for building mechanisms that make the nodes interoperate efficiently and securely. Thus, the policy of the UBIMOS infrastructure consists of the following major aspects:

Power consumption. Configurable modes of operations are necessary both for different operational circumstances

and for different modes of operations. For example, security operations require more energy sustainability. Thus, if a node operates in secure mode then precautions are needed for the provision of redundancy in the power source and reduced coverage both in the *distance* among the nodes having sensors (UAs and IMPs) and in the *size* of the operational domains. The size depicts the number of non–idle nodes in a given domain. The distance between two UA-nodes is depicted by the number of active hops between them, while the distance between a UA and an IMP, a UA and a 3C, and an IMP and a 3C is given by the time delay a packet takes to travel to its destination. As a result of the reduction in the domain size and shortened node distances, the number of instructions required by security and routing algorithms will decrease, which will substantially reduce the energy consumption. Secure exchange of data always requires larger packet sizes compared to insecure data exchanges. This is due to the secure node authentication and cryptographic data contents. Therefore, improvising on the energy constraints of wireless sensor networks is crucial. Considering the aspect of system availability, WSNs differ considerably from other existing networked systems. Due to extreme system availability and energy constraints, the design of WSNs requires a proper understanding of the interplay between network protocols, energy-aware design, signal-processing algorithms, embedded and distributed programming techniques, [27]. By dynamically configuring coverage of a domain, we can maximize the domain's lifetime instead of minimizing the energy consumption or maximizing the residual energy. There have been proposed similar models to minimize energy consumption of sinks in a WSN, [28] and [7]. With the UBIMOS policy, relocating of the nearest multi-hop nodes is periodically done so that the effective route path of messages diminish to a minimum level. Besides, in the secure mode, security functionality of the IEEE 802.15.4 will be bypassed by the secure transport protocol of UBIMOS. This will substantially eliminate the required power consumption in the physical layer.

Security. Achieving secure data exchange between small wireless systems is a challenging issue, which requires the design of interoperable, robust, time–efficient, easily applicable, scalable, and configurable systems. Based on the operation type, each node may change its operation mode to be secure. Especially, operations dealing with confidential operations must perform encrypted data exchange and secure authentications. There also exist situations where sensor data are transmitted securely to command control centers over insecure channels (e.g. Internet) or via dedicated RF channels. For example, one may continuously monitor, oil pipes, bridges, and critical passage points used for smuggling of drugs or immigrants. There are several security threats if the data from such nodes are transmitted in plain form. For example, critical data may be intercepted, hijacked, changed, and transmitted to unauthorized principals during a confidential operation or during the observation and exchange of real-time data. Launching the so called "man in the middle attack" can help adversary to intercept data transmitted in plain form, and the intercepted data can be disclosed, misused, and redirected to unauthorized principals. Security related problems and requirements for adequate solutions are considered separately in the Secure Communication section.

Scalability. A WSN node, regardless of the spread of geographical positions among its nodes, should be easily relocatable and scalable for configuring different coverage spaces by applying techniques for optimum relocation, shrinking, and augmenting. To achieve higher scalability and adaptability, design of a ubiquitous network should contain the composition of self-organizing elements. Adaptive approaches to the solution of power consumption problems in sensor networks are important means for effective scalability. Adaptive solutions may provide dynamic relocations in the organization of sensors while they are mobile and hence can drop out of the coverage or they may run out of power. These problems are considered under routing (VI-A) and self-optimization (VI-B).

Self-optimization. Nodes are arranged to apply dedicated routing algorithms, intelligent clustering approaches, and algorithms for sleep/idle mode operations. These algorithms vary from node to node depending on the node type. For examples, UAs are intensively active and require frequent use of intelligent clustering and modified multi-path routing algorithm. These are effective approaches needed for reducing the power consumption and diminished WSN traffic. A self-optimization algorithm comprised of the intelligent clustering, a modified multi-path routing algorithm, and idle period management algorithm is used to minimize the average consumed energy for active sensors during the data transmission and sensor data processing. Details are given in Section VI-B: Optimization Strategy.

Self-organization. During the initial setup and also periodically under operations, nodes should learn their relative distances to available sinks, mainly the nearest IMP domains. Here, the nearest neighbor algorithm applying the Minkowski metric is used to cluster UAs into routing groups in order to build a table on each node that contains shortest paths within its domain. Further, for locating and effectively communicating with the nearest IMP, each UA will be dynamically clustered around its nearest IMP using the hierarchical clustering algorithm. Since the requirements for scalability, self-optimization, and self-organization are closely related to the dynamics of the communication of the nodes, they are commonly

considered in Section VI, Setting Up a UBIMOS Domain.

IV. Overall Architecture

UBIMOS is a mobile operation management system with many actors composed of a secure communication infrastructure, mobile human operators carrying wearable computers and sensors, and stationary computer nodes with server capabilities that can securely communicate over long range RF and satellite communication channels. These elements can be configured to manage various types of emergency operations, ranging from natural disaster managements to anti-terror operations. Therefore, the entire system is organized around a secure communication infrastructure, which can serve many types of operational architectures each designed for a specific type of operation. For example, an architecture for a medical care center can be designed to aid ambulatory operations within a geographical region. In this domain patients are equipped with UA's sensors for monitoring and exchanging vital body parameters with the medical center. Another architecture can be setup to manage critical border security of a part of a country. Tracking of individual sports activities exercised under harsh conditions can also be managed by UBIMOS. For example, an operation for rescuing buried mountaineers by a snow avalanche or an accident occurred during a rafting sports activity.

The design of an operational architecture consists mainly of a secure communication protocol, back-end services, and a variety of wireless agents used by operation teams on the field. The secure communication protocol enables users to set up domains (operational architectures), create communication channels among the UBIMOS nodes and make them securely communicate with each other. The back–end services, comprised of computer servers and human operators, perform tasks related to the emergency response management, logging of sensor and communication data, and management of remote operations carried out by rescue/operation teams. Agent nodes, implemented as wearable computers with sensors, are responsible for collecting and transmitting sensor data both from team members and environments to emergency response locations (3Cs) within a given WSN segment (UBIMOS domain). The communication between an agent node and a 3C point is carried out in a full duplex form so that instantaneous conditions of the team members and interventions from the 3C point are accomplished safely.

As already mentioned, UBIMOS is designed to conduct operations in a number of domains varying from individual rescue operations to anti–terror missions. Members of the operations are equipped with UBIMOS agents, while the communication of the systems are organized as wireless sensor networks each with a specific operational domain. Some of the operational domains can be defined as civil defense, public transportations, maritime security, railroad security, traffic control, rescuing and anti–terror operations, and management of disasters (e.g., flooding, earthquake, fire). Accordingly, the domains are organized in WSNs, which can be geographically dispersed depending on the operation type.

Figure 1 shows the general architecture of the operational domains, where each domain consists of three major elements, command control centers, first level agents (IMPs), and second level agents (UAs). Both the IMPs and UAs are

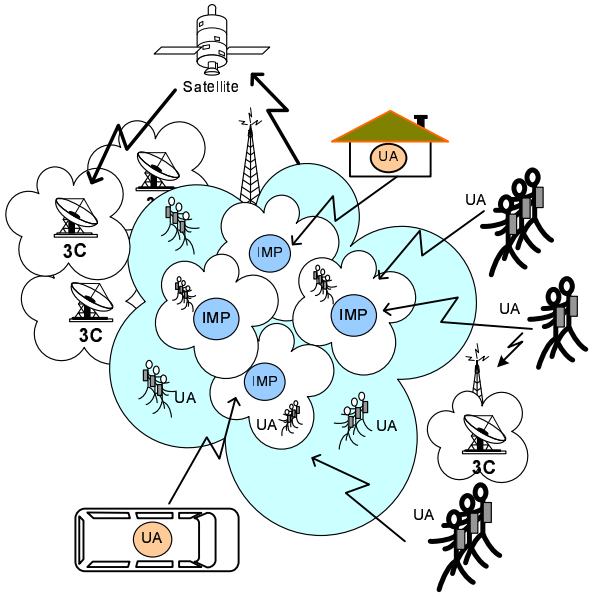


Figure 1. General architecture and components of the UA and IMP nodes

mobile embedded nodes each equipped with sensors and a wearable computer. The main difference between an IMP and UA is such that the IMP has much higher functionality for communication and routing. In addition to all functions a UA has, the IMP acts also as a gateway between its domain and the dedicated command control center. An IMP can also function as a bridging unit (and a sink also) between the UAs of its own domain and external UBIMOS domains. For this reason, depending on the size of a domain, fewer IMPs are harnessed in a given domain. Because, the UA nodes operate in much shorter communication ranges performing most of the intensive operations on the field. On the other hand, the IMP nodes can be configured to operate with reduced mobility taking care of mostly gateway operations among the UAs and the related command control center. At least one IMP must be assigned to a domain, otherwise long range communications between UAs and their command control center can be inefficient, or completely interrupted depending on the geographical distance. As shown in Figure 1, each cloud denotes an operation domain (actually a WSN), where each domain can work independently of others as well as cooperating with some other WSNs as required. As will be detailed later, each WSN is organized as a hierarchical communication system, in which the UA nodes

gather data and route the data to IMPs, which further forward the data to a command control center, and vice versa.

A. Organizing a UBIMOS Domain

UBIMOS domains are constructed according to the type of operations required, each varying from the single disaster management (e.g., mountaineer) to squadron based anti-terror operation. Even with fewer rescue members UBIMOS can be used to aid rescue squadrons if a domain contains all functional parts.

Implementation of hardware and software parts of the UBIMOS infrastructure are described here. Functionally, such an infrastructure is composed of three main subsystems: (i) agents (UAs), (ii) intermediate message passing units (IMPs), (iii) command control centers (3Cs). Both the agents and IMPs are mobile units carried by humans, mounted on vehicles or carried by other subjects, as necessary. On the other hand, 3Cs are stationary decentralized systems that remotely conduct all operations of a given domain and coordinate emergency response units/forces that are mobile on the field.

Based on a major technical requirement, nodes of a UBIMOS network are designed to run effectively on ARM–based small systems. An agent is, indeed, a powerful small computer (or a microcontroller) that remotely collects and processes sensor data from individuals and operation environments and transports the aggregated data to a command control center. That is, agents collect and process sensor data and broadcast them to other UAs and IMPs, which than forward the data to the emergency center. In general, an agent is a mobile client system that monitors some specific parameters of a given subject and transmits them via IMPs to a remote emergency center and obeys instructions received from that center, i.e., the command control center.

Software and hardware modules of both IMPs and UAs are designed to run on embedded systems, having small sizes, but powerful enough for collecting, processing, exchanging the necessary sensor data, and communicating securely with other nodes. Figure 2 shows the structural organization of an ARM-based node specifically designed for the UA and IMP nodes. The UBIMOS runtime environment is designed as a virtual machine with limited functionality, which manages all SW modules that receive sensor data, process the data, and prepare data packets for the transportation. Most obviously, the virtual machine performs scheduling and dispatching of processes needed for all type of data processing and communication software. The virtual machine is also responsible for system diagnoses, debugging, maintenance, and for the execution of UBIMOS processes on the tiny Linux kernel. The sensor interface module is responsible for collecting the physical sensor data and passing them to the sensor data processing module. The transport and security module is a light weight transport

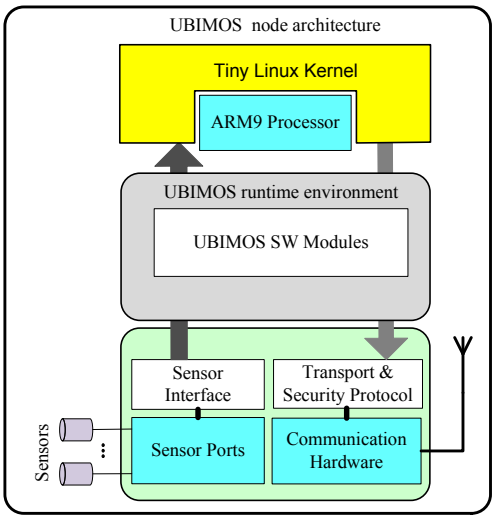


Figure 2. The structural organization of the wearable computer used as UAs and IMPs

protocol implemented with data encryption, decryption, and secure node authentication functions.

Both the UA and IMP nodes are small hand-held devices that are embedded with sensors, while the 3C nodes are general purpose stationary computer systems with server capabilities. A 3C node receives the sensor data from UAs and IMPs, analyses them, displays the locations of UAs and IMPs on a map. Furthermore, vital conditions of the remote users and environmental conditions of the operation field are also analyzed and necessary actions are taken by coordinating necessary instructions with the operation forces. The command control centers also encrypt and store the communication and log information about all events into a database.

In short, a UBIMOS agent works as a mobile system observing, collecting, and distributing vital body conditions of its user and environmental conditions wherever the user and/or objects might be ubiquitously located. As mentioned later, IMPs operate both as an agent and a gateway, while the command control center coordinates and manages the domain operations of a given set of UBIMOS domains. Indeed, an IMP works as a sink that collects the sensor data form agents of a given domain and forwards the data to a dedicated command control center. UAs also do routing, but not as comprehensive as IMPs do. UAs use simple hop–by–hop multi–path routing that only select and flood domain specific sensor data while discarding unrelated, duplicate, and obsolete sensor data packets.

B. Sensor Data Processing

In order to facilitate prioritizing in routing and effectively processing of sensor data, a task based classification of the

sensor data is designed. Sensor data are aggregated and packed following a special packet format shown in Figure 3. The first field of the packet holds an urgent flag, the second field contains a four–bit code identifying 16 different sensor types. The third field contains the sensor data to be queued for broadcasting, the Memory address field contains the address of the first memory location of the sensor data. Finally, the last field contains the length of the sensor data residing in the memory. The urgent flag can be used by

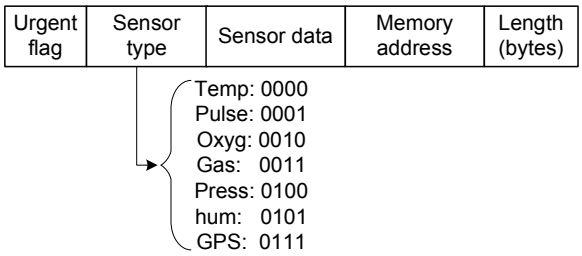


Figure 3. Sensor data format for the process queue

sensor applications, where depending on the application type, each application can explicitly set the urgent flag. For example, some of the ambient data or vital body signs, e.g., temperature or blood pressure may increase unduly, then the related sensor reading code will set its urgent flag. The sensor data processing module checks first whether the urgent flag is set, if so the related sensor data are packed and put in front of the queue for an immediate processing.

There are several reasons to set the urgent flag, (i) if an interrupt is received from the urgent button (hardware interrupt), (ii) the value read from a sensor exceeds its predefined normal range, (iii) the node goes into sleep mode for power saving mode, (iv), the node has reached minimum power level, (v) the node voluntarily changes its state from operative to idle in order to facilitate the self-optimization of the domain it belongs to. If a node announces itself as being idle, then the neighboring nodes delete the respective entry in their routing table in order to reduce the power consumption. This action is taken by the self-optimization algorithm invoked periodically. During the data aggregation from the sensors, excessive values are detected and packed accordingly with the urgent flag set, and then put in front of the process queue for broadcasting. The pseudo–code shown in Algorithm 1 describes the polling of sensor ports, data aggregation from the sensors, and queuing and broadcasting of sensor data.

As observed, we use the notation of object–oriented programming paradigm, where Obj $\rightarrow$ Val denotes a referral to parameter Val of object Obj. As usual, an object defines a conglomerate data structure associated with a given class. For example, Sensor $\rightarrow$ Value := readPort (SP) means that the algorithm reads sensor port SP and assigns the value read to Value parameter of the sensor object Sensor. Thus,

Algorithm 1: Sensor data aggregation and packet generation

```
Algorithm Readsensors()
input : Port addresses of sensors (SP) from 1 to 7
output: Sensor data packets into broadcast queue
for SP ← 1 to 7 do
    if Ready(SP) then
        Sensor := createSensor(SP) ;
        Sensor → Value := readPort(SP) ;
        Sensor → Type := assignType(SP) ;
        Sensor → Urgent := isUrgent(Sensor → Value) ;
        Enqueue(Sensor) ;
    end if
end for

Algorithm Broadcast()
input : Sensor data queue
output: Broadcast sensor data via RF module
while (¬ eof(sensorQueue)) do
    Sensor toSend := Dequeue(sensorQueue);
    Packet Pb := makeBroadcastPacket(toSend);
    Send(RFport, Pb);
end while
```

the above code first generates a sensor object for each sensor, reads and stores the sensor data into the associated sensor objects, checks the urgent flag, and puts the sensor data into a processing queue. This queue is then traversed, sensor data objects (packets) are converted to transport packets, and queued on the RF port for broadcasting. Since most sensors return analog data, the readPort() function reads the analog quantity, converts it to binary form, and stores the result into a memory location. The isUrgent() function reads the binary sensor data from the associated memory location and makes a quick computation (threshold masking) to determine the current value of the urgent flag.

V. Structural View of the UBIMOS Subsystems

As described below, the ubiquitous emergency management system realized so far is designed in three separate subsystems: ubiquitous agent (UA), intermediate message passing (IMP) subsystem, and command control center (3C). Based on internationally recognized standards, the implementation of these subsystems confirms to emerging technologies regarding the software formats and capabilities of small-sized hardware.

A. Agent Subsystem

An agent subsystem is responsible for collecting sensor data, preprocessing the data, and transmitting them to the nearest IMP that has bridging capabilities between the sensor agents and a 3C node. An agent is also able to receive instructions from command control centers and act upon the contents of the instructions. Several agents and IMPs together with a 3C node are organized into a WSN (so called UBIMOS domain) in order to facilitate the application of the multi–path routing algorithm with controlled power constraints.

Each agent subsystem is implemented in five dependable modules: Communication, Sensor, System diagnose, Control, and Security module. The Communication module of UAs is responsible for transmitting preprocessed sensor data from the environment and/or from the team member to its base station, 3C, via other UAs and IMPs that are reachable throughout its routing path. The data exchange is accomplished by UBIMOS's secure transport protocol, while routing of the sensor packets are carried out by a special hop–by–hop multi–path routing algorithm.

Since the UA nodes are constrained for minimizing the energy consumption and data transfer bandwidth, the routing algorithm must, in addition to the classification of the nodes in a WSN, consider the rejection of duplicate packets. The classification of the sink nodes within a WSN is defined by a flag (*DST_type*) in the transport protocol header. When an IMP node sees a *DST_type* flag set (*DST_type = 1*) it disables flooding the related packet, instead, it forwards the packet to its 3C node. By default *DST_type = 0* on UAs and IMPs in a domain, i.e., multi-path routing is enabled regardless of an unreachable sink node.

Sensor module of an agent subsystem is responsible for collecting and processing data available at sensors' ports. Since the sensors are connected to hardware ports (analog and digital) they are addressed by the respective port addresses, which are denoted as SP to identify a given port.

Security module of the agent subsystem is responsible for providing secure authentication, confidentiality, integrity of data, and availability of its services. Secure authentication, confidentiality, and integrity functions are provided by using a light weight implementation of the RSEP protocol, [11], however availability of services is done by self-organization algorithms and by the frequency hopping technique [29], [30], which is a built in functionality of the RF transceiver module. As already noticed, the RF communication module of the UA subsystem is responsible for the communication between UAs, IMPs, and with the related 3C node. It is clear that the self-optimization algorithm can also enable higher availability of the operational nodes, since the amount of flooding and the number of packets during routing are dynamically reduced. As considered later in this paper, the power consumption is also reduced by periodically running the self-optimization algorithm.

System diagnose module is responsible for probing the sensors and other system parts (including the software modules) whether they are properly functioning. The diagnose module is always invoked once during the system startup, however, it can be executed whenever an overall system diagnose is required by its user.

Control module of the agent subsystem manages all other software modules mentioned above. It is mainly involved in process scheduling and dispatching of the system modules. Checking of urgent flags and priority handling of the sensors are also done by this module.

B. IMP Subsystem

Intermediate Message Passing subsystem is an extended agent unit having some additional features, such as routing to 3C nodes via a satellite, long range RF link, or a GSM link. It uses exactly the same modules for gathering and broadcasting of the sensor data as that of the agent subsystem. Though an IMP can be configured to function as a sink in a predefined UBIMOS WSN, it provides bridging between each UA and the control center (3C) within a given WSN. Thus, it has an additional module for the 3C communication to bridge UAs of a WSN to their 3C nodes and to other WSNs as necessary. Since the modules of the IMP subsystem are extended replicas of the UA subsystem, we omit detailed description of them here, and rather refer to the description of the UA subsystem.

C. 3C subsystem

As mentioned earlier, a 3C node receives the sensor data form UAs and IMPs, analyses them, displays the locations of UAs and IMPs on a map. Furthermore, vital conditions of the remote users and environmental conditions of the operation field are also analyzed and necessary actions are taken to intervene critical situations. Moreover, it encrypts and stores the communication data and log information about all events during an operation into a secure database. Hence, the 3C subsystem is responsible for decision–making operations related to the contents of sensor data received from UAs and IMPs. The decision–making operations include processing of the sensor data, creating and submitting alerts, managing emergency situations, coordinating the operational domains, logging, and securely saving the processed sensor data. Data records for the remote team members, alerts, UAs, and IMPs are kept in an encrypted database. In short, overall management of a given set of operational domains is carried out by a 3C node. Figure 4 illustrates a 3C console, where the sample domain contains two different operation teams, Team 1 and Team 2. The locations shown on the map are built using the GPS data received from the team members. The screen also illustrates the vital body information and some environmental data (e.g., pressure, humidity, and ambient temperature) gathered from the remote users (UAs and IMPs).

Regarding the software components, the 3C subsystem is composed of five software modules, Security, Monitoring, Communication, Logging, and Control module. Additionally, it maintains a database to store event information in an encrypted database.

In cooperation with the Communication module, the Security module of 3C subsystem performs secure authentication and encrypted data exchange with its domain nodes. The Communication module of 3Cs is also responsible for data exchange among the 3C nodes and other systems over the Internet. It differs from the corresponding modules of UAs and IMPs in a way that the 3C nodes can also communicate

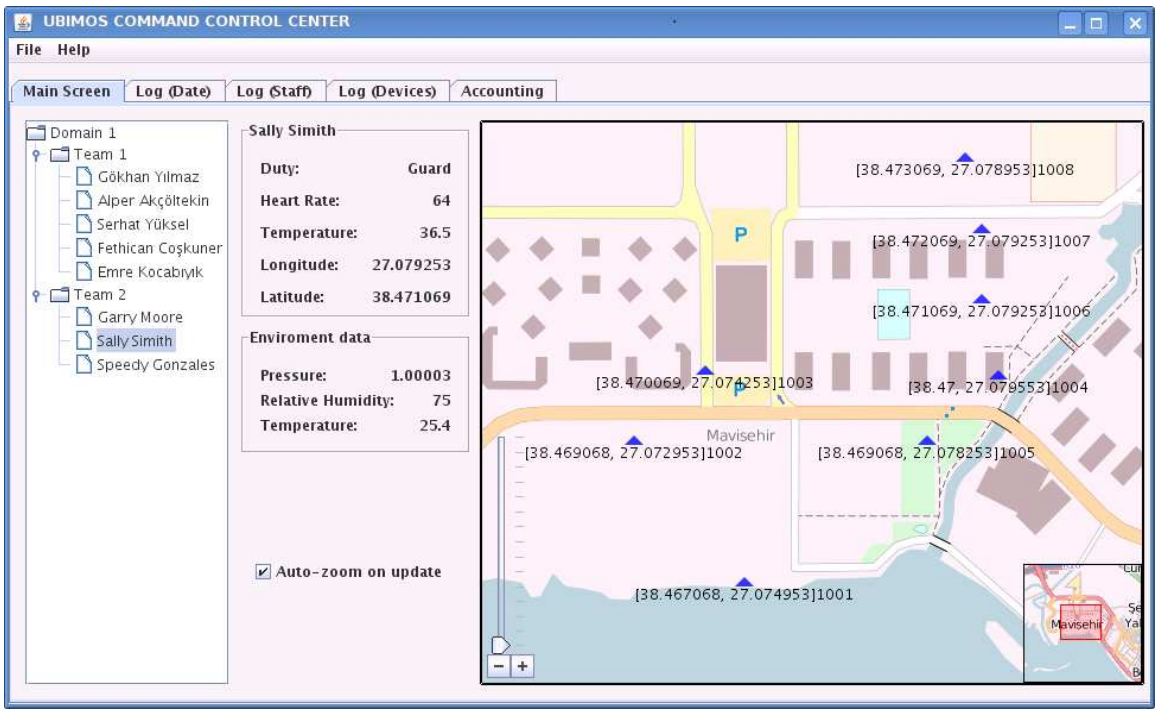


Figure 4. A screenshot of the command control center managing the domain Domain 1

with IMPs via satellite and other long range RF and GSM links.

The Logging module of the 3C subsystem is responsible for securely logging of every single event in detail during the communication. The event data logged so far will also be used to backtrack and compose accountability information on the team members and other subjects found in a given domain. Operations such as insertion, search, and retrieval of the event data are also accomplished by the Logging module in cooperation with the Control module.

The Monitoring module manages real–time monitoring of the remote activities and alerts. For example, the screenshot shown in Figure 4 is processed and displayed by the Monitor module. This module also provides the necessary data to the related command control center needed for its procedural operations such as team organization, coordination, and cooperation of operational domains.

Similar to UAs and IMPs, the Control module of 3C subsystem is responsible for providing a unified interface to underlying modules in order to manage the remaining software modules in a multi–threaded process execution environment.

VI. Setting Up a UBIMOS Domain

For each ubiquitous operation a WSN must be setup by a dedicated command control center, where depending on the type of the operation, either secure or non-secure mode can be initially chosen. For the sake of reliability of the dynamic relocation of the domain nodes, each domain must first specify at least one IMP for relying of data from its UAs. If only one IMP node is available, then the IMP node must have a high availability feature. We can also define some UAs to operate as IMPs if the operation allows. Initial setup of a UBIMOS domain is completed in three main steps:

1) neighbor discovery and authentication of neighbors,
2) self-optimization: *k*–nearest neighbor clustering for preventing nodes from duplicate packet flooding to heavily loaded paths and farther destinations,
3) self-organization: hierarchical clustering of nodes for the detection and configuration of IMPs and 3Cs in a hierarchical task structure.

Figure 5 shows a sample topology of three UBIMOS WSNs organized for a specific operation. Here, basically, the hop–by–hop multi–path routing algorithm is issued with some modifications. The modifications are applied to processing of the duplicate packets and using IMP and 3C nodes as sinks when routing the data packets to their ultimate destinations within a communication pathway. Within these domains, an agent (UA) can only route using flooding within its own domain, while IMPs and 3Cs can perform inter–domain routing. Since IMPs have also limited distance coverage compared to 3C nodes, they do best-effort routing during the data exchange with external domains, either directly to a peer (UA or IMP) or to a 3C node.

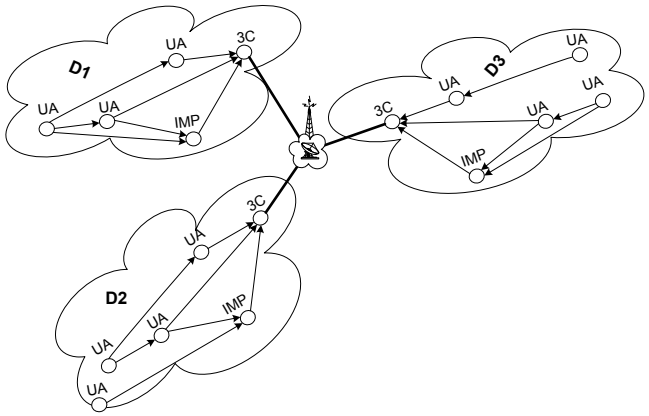


Figure 5. A topological view of three UBIMOS domains configured to interoperate

That is, prior to an operation a UBIMOS domain is constructed (clustered) around at least one IMP node and one 3C node. Clustering of the nodes (UAs, IMPs, and 3Cs) is based on node IDs, functionality, and distances that are measured as time delays for the delivery of data packets. Each node stores a list of node IDs and their cryptographic hash values in a simple data structure. Using the list of these node IDs, each node runs the nearest neighbor clustering (based on Delaunay Triangulation) algorithm for discovering and clustering neighbors, and for assigning appropriate IMPs and 3Cs to their domains, see Figure 6. Details about other localization algorithms can be found in [31].

Figure 6 (a) shows that a clustering of UAs around higher level nodes (IMPs and 3Cs) is realized first and then the IMPs cluster themselves around the 3C nodes, Figure 6 (b). It is important to note that if a UA finds a 3C node, it immediately connects itself to that node without making a distinction between the IMP and 3C. This case is illustrated in Figure 6 (a), where for example, in domain D1 two UAs cluster themselves around a 3C, even though they have an IMP in their domain. Initially, the nodes always search for higher level nodes with shorter distances in the hierarchy for finding an appropriate path.

A. Routing

Power constraints, self-organization, and self-optimization are the main aspects considered when designing the routing policy of UBIMOS. Intensive mobility, long distances between the nodes, and physical obstacles always degrade signal quality of WSNs, which in turn, causes packet delays and frequent packet drops at the nodes. This leads to extreme use of both the channel bandwidth and power consumption at the nodes. To cope up with these and several other known constraints, routing policy for such critical systems should be carefully designed and applied. For example, one item of the UBIMOS roting policy states that an agent (UA node) can only route within its own domain, while IMPs and 3Cs can additionally perform inter–domain routing. This policy prevents a UA from flooding its packets to exterior domains, and hence, significantly reduces the power consumption both for itself and for the flooded nodes. With the inter–domain routing the nodes can exchange sensor data among diverse domains and involve in operations of other domains, which are coordinated by any authorized 3C. Agents are only responsible for collecting and flooding the sensor data within a given operation domain. The part of the algorithm taking care of packet duplicates and hop–by–hop packet forwarding to a specific type of nodes is described by the pseudo–code given in Algorithm 2.

Algorithm 2: Hop-by-hop multi-path routing at the UA nodes

```
Algorithm UAreceive(Packet received)
input : Received packet
output: UA Hop-by-hop Routing
if (received → DSTaddr = thisAddr) then
    Enqueue(processQueue, received) ;
    exit ;
end if
if (received → hopCount > MaxHop) ∨
(Duplicate(received)) then
    drop(received);
    exit ;
end if
if (received → DSTtype = IMP) ∨
(received → DSTtype = CCC) then
    Packet P := repackTosink(received) ;
    Hash h := computeHash(P → ID, P → seqNo) ;
    save(hashTable,h) ;
    Broadcast(P,DSTtype) ;
    exit ;
else
    Hash h := computeHash(received → ID, received →
    seqNo) ;
    save(hashTable,h) ;
    send(RFport,received);
end if
```

```
Algorithm Duplicate(Packet P)
input : Received packet
output: Check the duplication of packets
Hash h := computeHash(P → ID,P → seqNo) ;
if search(h,hashTable) then
    return true ;
else
    return false ;
end if
```

B. Optimization Strategy: Self-Organization for Energy Optimization

During any operation UAs may become quite hectic while exchanging data with each other under various physical conditions. Overall efficiency of an operation mainly depends on faster and reliable communication among the nodes. Periodic self-optimization is required to reduce network traffic bandwidth by decreasing distances and time delays during the data exchange among the active nodes. The self-optimization

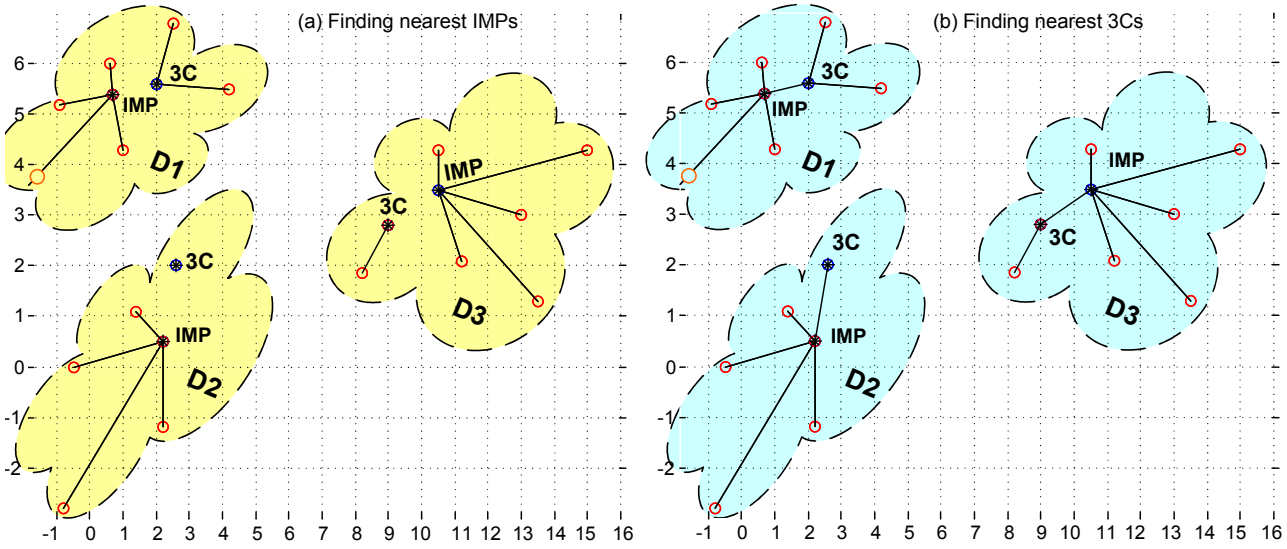


Figure 6. Initialization of UAs and IMPs: UAs cluster around nearest IMP and 3C nodes, while IMPs cluster around the nearest 3C nodes only

policy is mainly realized by dynamic organization/relocation of the active nodes. At the startup of the UA nodes, a self-organization algorithm is run to colonize (insert) nodes into one of the active neighborhoods governed by an IMP. This is generally performed in $O(n \log n)$ running time if a Delaunay Triangulation algorithm is used. Here n denotes the number of nodes traversed for the triangulation. During an active operation, the nodes periodically recompute their new locations in order to dynamically insert themselves into more effective communication paths. The effectiveness is measured by the degree of packet delays among the nodes, rate of packet loss, and size of the convex hull of the domain under consideration. If these constraints can be adequately managed then we can achieve highly effective packet routing and hence substantially reduced energy consumption.

Regarding the periodic relocation, during an operation some UAs may become idle, fall outside of the communication coverage of its operational domain, or switch off (go into sleep mode). In these cases, the domain must reduce its routing coverage and hence flooding space in order to preserve more energy. The nodes are configured to periodically run the optimization algorithm for clustering around *k*–nearest neighbors. The idle nodes set their *IDL* flag in the transport protocol header (next section) to indicate the out of operation status, while other ”unheard” nodes are assumed to be idle if within a given time duration they have not shown any activity. Obviously, *the number of IDL flags that are unset within the newly computed domain boundary* gives us the number of the nearest k nodes for the new cluster size. Thus, clustering with this threshold size gives a new active set of operational domains. Figure 7 illustrates relocating of active nodes around the domains D1, D2, and D3. Blue dots (UA nodes) are assigned to D1, black nodes to D2, and red nodes are assigned to D3. Encircled nodes are clustered around the related IMPs, while the others are distant nodes, which are also able to communicate with their domains if they are in the communication coverage. However, some of the previously active nodes are now out of the operational domains, which can be interpreted as if they were either become voluntarily idle or forced to be off. Most

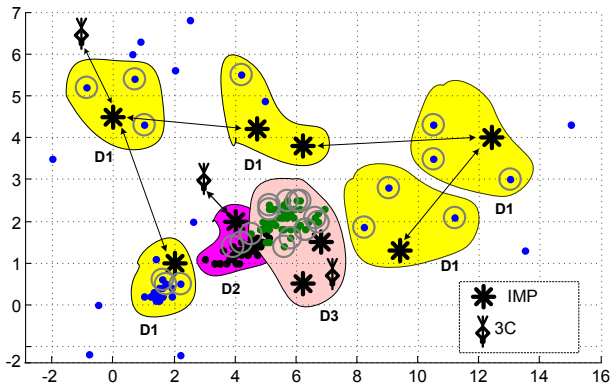


Figure 7. Clustering around *k*–nearest neighbors

importantly, during the initial setup of a UBIMOS domain, the nodes are clustered into a WSN based on the nearest neighbor algorithm. That is, the nodes that are close to each other are organized into a dedicated routing level. The UA nodes are first clustered into nearest neighbor clusters among themselves, where each cluster is assigned a routing level with other clusters. These clusters are than hierarchically clustered towards the nearest IMP. Further, the nearest IMPs in the hierarchy are also clustered towards their nearest 3Cs. Figure 8 shows the clustering of three UBIMOS domains, D1, D2, and D3. By this clustering, the number of active routing (flooding) nodes reduces to the number of joint

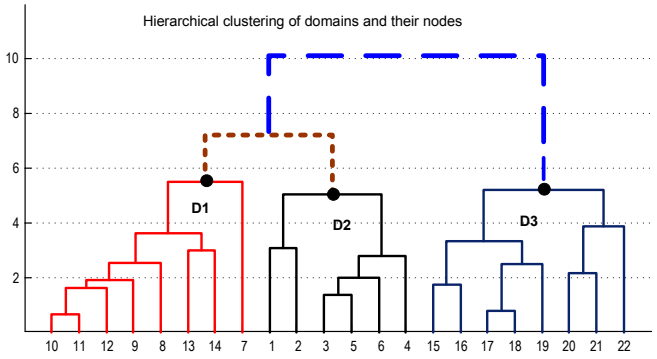


Figure 8. A dendrogram showing the hierarchical clustering around nearest IMPs and 3Cs using the Minkowski metric

points of the entire domain. For example, as shown in Figure 8, after clustering the UAs, the number of flooding nodes of the domain D1 reduces to 6, which is 8 for the plain multi-path flooding algorithm. The clustering of the nodes in this example are done binary-wise, however, since we use average distance to compute the routing level of a cluster we can organize each cluster as a splay tree. This can further reduce the number of flooding points significantly compared to the binary tree.

VII. Secure Communication

Some operations require secure authentication during the connection establishment, and confidentiality and integrity of data during the data exchange. Especially, communications needed for anti–terror operations, public transportations, and civil defense must be securely carried out. Rescue operations related to public safety operations such as urgent health-care services, earthquake, snowslide, mugslide, fire, flooding, and other natural disaster rescue operations need not be confidentially done, however, integrity must always be ensured. For example, during a rescue operation dealing with flooding or fire catastrophes, operation team member accountability and information on vital signs, including body temperature, pulse rate, respiration rate, and blood pressure must be reliably and quickly transmitted to the control center. However, during anti–terror or highly confidential operations data should be exchanged both confidentially and reliably.

It is important to note that, security mechanisms, such as integrity, peer authentication, and confidentiality, are applied to all types of nodes. However additionally, the command control centers must encrypt before saving the sensor data and communication information that are logged during the domain operations. That is, prior to a connection establishment, each node must use secure authentication regardless of the type of the domain operation. However, during the data exchange phases, nodes may perform unencrypted data transport, depending on the operation type. Unless specified, the command control center saves the data in the encrypted form. Nevertheless, during the initial setup of a UBIMOS domain, nodes do authenticate each other using a light weight implementation of the RSEP protocol, [32].

A. Transport Protocol

Transport protocol of UBIMOS nodes is thus accordingly designed to ensure integrity, confidentiality, and secure authentication. Packet format of the transport protocol is shown in Figure 9. One octet is reserved for flags, where

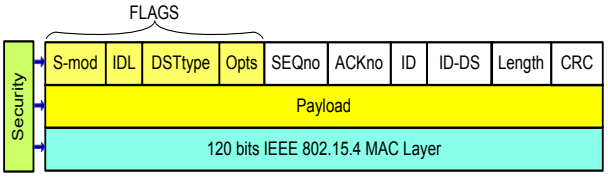


Figure 9. The structure of the Transport Protocol of UBIMOS.

three of the flags are predefined for security, idle nodes, and for destination type, and the remaining five flags are optional and available to other applications. The first flag, *S-mod*, is used for switching to secure mode, *IDL* is used to inform destinations if the source node is in one of idle, sleep, or in off mode. *DSTtype* identifies the type of the destination, whether the destination is a sink, i.e., IMP or 3C. We have two 16-bit fields storing the packet sequence number (*SEQno*) and acknowledgment (*ACKno*), which are comparable to that of the TCP fields. The *Length* field contains a 16-bit length of the entire packet defined as number of octets. The final field, *CRC*, is indeed a hash value of either the entire packet or the header only. This field is used to ensure the integrity of the received data. In the secure mode, *CRC* contains the hash value of only the header, however, for the insecure mode this contains the hash value of both the header and the payload (data). In the secure mode the CRC of payload is not necessary since the payload will always be hashed by the security protocol.

Regarding the *ID* field, for the implementation of the required security functionality, each node is associated with an ID defined as a lightweight X–509 certificate, which has the function of describing current ID of the source node, operation team member, and the operation code (or the operation ID). During the authentication, every node (UA, IMP, and 3C) should present its certificate, where each certificate, among others, is comprised of the issuer of the certificate, a legitimate team member ID, and a unique ID of the node. That is, each node is assigned a unique ID, users's public key information, name and digital signature of the issuer, time stamp, serial number, version, last update date, period of validity, and digital signature algorithm (the algorithm used to sign the certificate). The reader is referred to the description of CCITT X.509 v3 [33] for further details, which defines a standard certificate format for public key certificates and required procedures for the certification validation.

A digital signature, which is generated from the sender node ID, is stored in the *ID-DS* field. This field is used by the receiver to securely authenticate the source node. To do so, the receiver, during the domain setup, computes and stores a list of hash values of the IDs of its domain members. Later, during the authentication, the receiver computes a new hash value of the ID field coming from the remote node, and compares the newly computed hash value with the one already registered in the list. If these hash values are equal then the authentication of the remote node will be verified and acknowledged back to the sender.

In order to reduce the energy consumption and avoid redundancy in security mechanisms, the security functionality of the IEEE 802.15.4/ZigBee Protocol is not used. That is, in the secure mode, in order to ensure secure authentication of the UBIMOS nodes and confidentiality of the data exchanged between the nodes, all communications are tunneled with a light weight implementation of the RSEP protocol [11], [34] using the elliptic curve algorithm [35].

VIII. Prototype Implementation

Ubiquitous agents and IMPs are designed to run on ARM–based platforms, whereas mainstream computers are chosen to be used as the 3C servers. In essence, the 3C servers are stationary and relatively high-capacity systems, which must perform resource demanding operations such as encrypted database functions, satellite communication, monitoring, real–time alerts, and real–time rescue management tasks.

As the ARM processor we have used TS-7350® [36], which is a compact full-featured single board computer based on the Cirrus EP9302 200MHz ARM9 CPU, which allows development of multi-function embedded applications through its multiple peripheral interfaces. The ARM processor is a 32-bit reduced instruction set computer (RISC). It was known as the Advanced RISC Machine, and before that it was known as the Acorn RISC Machine. The ARM architecture is a widely used 32-bit RISC processor, which was originally conceived as a processor for desktop personal computers by Acorn Computers. The relative simplicity of ARM processors made them suitable for low power applications. This has made them dominant CPUs in the mobile and embedded electronics market as relatively low cost and small microprocessors and microcontrollers.

The sensors were designed separately and connected to related input connector pins. For the GPS module, we have used Atheros AR1511®, which consists of a tiny CMOS AR1511 GPS IC, a highly-integrated GPS receiver comprised of a single conversion RF front-end and a GPS baseband processor all combined on a single die. Additionally, we have used five other sensors: OTP-538U for body and environment temperature measurement, SDT1-028K for heart rate, MQ135 for gas and air quality, SHT75 for humidity, HP03D for pressure measurement.

A. Reviewing the Implementation

We have focused on the power consumption and the efficiency of the choice of the programming language used for the implementation. Generally, main limitations of nodes in a sensor network relate to power consumptions and necessary energy–saving algorithms. Battery life for such mobile units can be slightly prolonged by use of efficient routing algorithms and denial of sleep protection mechanisms. For the security enhanced operations, UBIMOS nodes will naturally consume more power due to secure authentication, encryption, and decryption algorithms used for the confidentiality of the data exchanged.

Use of the development language has a crucial role in both the transmission speed and the power consumption. Therefore, we have implemented the system both in Java and C++ languages. Following the implementation, we conducted several test and evaluations of the system on two different CPU architectures, ARM and Intel®. Since the software components other than the security and transport modules run in constant time, we evaluated only the implementation of the security and transport modules. Encryption, decryption, and secure authentication algorithms require extensive CPU and data transmission resources, all depending on the size of data blocks being handled. The results of running times versus input block sizes for Java and C++ implementations of the security module are shown in Figure 10. Although the encryption algorithm can use larger key sizes, we have used 116-bit elliptic curve algorithm and 128-bit RC4 stream encryption algorithm in the prototype version of UBIMOS. These key lengths are known to be relatively moderate. As known, the key length in an encryption algorithm increases the strength of the algorithm while decreasing the time-efficiency. Hence, the key length can be increased for the 3C systems, but for the mobile nodes 116-bit elliptic curve and 128-bit RC4 are optimum.

Finally, we have experimented with a secure real–time operation using 5 UAs, 2 IMPs, and 2 3Cs in order to observe the efficiency of the communication speed. The nodes were spread around an area of 50 km/radius. Both the simulation and the real–time operation results with this size of network were shown to be successful. However, simulation results of a real–time operation have shown that increasing the number of UA and IMP nodes beyond 200 nodes caused a running time of 10 ms/packet per UA node, which caused an unacceptable delay in total for the security–enabled operation. This shows that, in order to increase the communication efficiency, a UBIMOS WSN with larger sizes should be segmented and bridged using more IMPs. Although the prototype implementation of UBIMOS gave satisfactory results, in a future work, we need to perform more efficiency analysis regarding the power consumption, communication speed, and strength of the security functions.

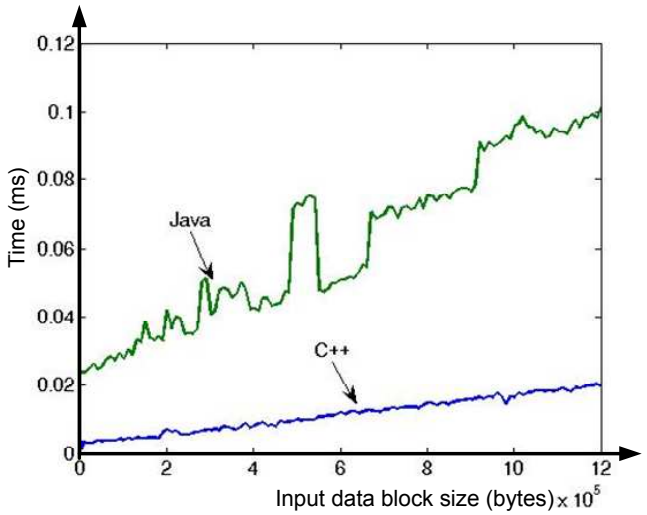


Figure 10. Running times of the secure authentication; C++ implementation versus Java implementation

IX. Conclusions

We have designed and implemented a ubiquitous management system to exchange vital body signs, environmental conditions, and locations of subjects during critical rescue operations. Data on the environmental conditions and vital body signs known as heart rate, oxyhemoglobin saturation, and thermal stress factors were successfully collected by ubiquitous nodes within a WSN, and transmitted to the command control centers for further processing. The results obtained from the simulations and the prototype implementation will allow us to refine new assumptions made when designing future hardware, software, protocols and mechanisms for more critical operations. Although the prototype implementation of UBIMOS gave satisfactory results, in a future work, we need to perform detailed efficiency analysis regarding the power consumption, communication speed, and strength of the security functions.

X. Acknowledgment

We thank our students Gökhan Yilmaz, Emre Kocabiyik, Fethican Coskuner, Alper Akçöltekin, and M. Serhat Yüksel for helping us in coding and testing a prototype system, and special thanks to Computer Sciences Laboratory staff of Izmir University of Economics for their support in facilitating the laboratory experiments during the implementation of the prototype system. Finally, we owe special thanks to the management of the Faculty of Engineering & Computer Sciences for funding the entire project.

References

[1] S. Kondakci, G. Yilmaz, E. Kocabiyik, F. Coskuner, A. Akcoltekin, and M. S. Yuksel, "Ubiquitous monitoring system for critical rescue operations," in *Proceedings of the 2010 6th International Conference on Wireless and Mobile Communications*, ser. ICWMC '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 515–520. [Online]. Available: http://dx.doi.org/10.1109/ICWMC.2010.103

[2] J. L. W. K. J. Ruskin, "Pulse oximetry: basic principles and applications in aerospace medicine," *Aviation, space, and environmental medicine*, vol. 78, no. 10, pp. 973–978, October 2007.

[3] R. B. Hetnarski and M. R. Eslami, *Thermal Stresses – Advanced Theory and Applications (Solid Mechanics and Its Applications)*, 1st ed. Springer, December 2008. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/1402092466

[4] "Council of Europe Convention on the Prevention of Terrorism," accessed June 2010. [Online]. Available: http://conventions.coe.int/Treaty/EN/Treaties/Html/196.htm

[5] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Comput. Netw.*, vol. 52, no. 12, pp. 2292–2330, 2008.

[6] K. Romer and F. Mattern, "The design space of wireless sensor networks," *Wireless Communications, IEEE*, vol. 11, no. 6, pp. 54 – 61, dec. 2004.

[7] A.-K. Chandra-Sekaran, C. Kunze, K. D. Mller-Glaser, and W. Stork, "Self-organizing zigbee network and bayesian filter based patient localization approaches for disaster management," *International Journal on Advances in Intelligent Systems*, vol. 2, no. 4, pp. 446 – 456, 2009. [Online]. Available: http://www.iariajournals.org/intelligent_systems/

[8] K. Sha, W. Shi, and O. Watkins, "Using wireless sensor networks for fire rescue applications: Requirements and challenges," in *Electro/information Technology, 2006 IEEE International Conference on*, May 2006, pp. 239–244.

[9] X. Du and H.-H. Chen, "Security in wireless sensor networks," *Wireless Communications, IEEE*, vol. 15, no. 4, pp. 60 –66, aug. 2008.

[10] H. Kumar, D. Sarma, and A. Kar, "Security threats in wireless sensor networks," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 23, no. 6, pp. 39 –45, june 2008.

[11] S. Kondakci, "A remote IT security evaluation scheme: A proactive approach to risk management," in *IWIA '06: Proceedings of the Fourth IEEE International Workshop on Information Assurance*, vol. 1. Washington, DC, USA: IEEE Computer Society, 2006, pp. 93–102.

[12] M. Ismail and M. Sanavullah, "Security topology in wireless sensor networks with routing optimisation," in *Wireless Communication and Sensor Networks, 2008. WCSN 2008. Fourth International Conference on*, dec. 2008, pp. 7 –15.

[13] W. K. Koo, H. Lee, Y. H. Kim, and D. H. Lee, "Implementation and analysis of new lightweight cryptographic algorithm suitable for wireless sensor networks," in *Information Security and Assurance, 2008. ISA 2008. International Conference on*, april 2008, pp. 73 –76.

[14] A. Boukerche, Y. Du, J. Feng, and R. Pazzi, "A reliable synchronous transport protocol for wireless image sensor networks," in *Computers and Communications, 2008. ISCC 2008. IEEE Symposium on*, july 2008, pp. 1083 –1089.

[15] Y. W. Law, M. Palaniswami, L. V. Hoesel, J. Doumen, P. Hartel, and P. Havinga, "Energy-efficient link-layer jamming attacks against wireless sensor network mac protocols," *ACM Trans. Sen. Netw.*, vol. 5, no. 1, pp. 1–38, 2009.

[16] M. Brownfield, Y. Gupta, and N. Davis, "Wireless sensor network denial of sleep attack," in *Information Assurance Workshop, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC*, june 2005, pp. 356 – 364.

[17] M. Mura, F. Fabbri, and M. Sami, "Modelling the power cost of security in wireless sensor networks : The case of 802.15.4," in *Telecommunications, 2008. ICT 2008. International Conference on*, june 2008, pp. 1 –8.

[18] N. Wakamiya, S. Arakawa, and M. Murata, "Self-organization based network architecture and control technologies for new generation networks," *International Journal on Advances in Intelligent Systems*, vol. 3, no. 1 & 2, pp. 75–86, 2010. [Online]. Available: http://www.iariajournals.org/intelligent_systems/

[19] K. Yamasaki and T. Ohtsuki, "Design of energy-efficient wireless sensor networks with censoring, on-off, and censoring and on-off sensors based on mutual information," in *Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st*, vol. 2, may-1 june 2005, pp. 1312 – 1316 Vol. 2.

[20] H. Shaban, M. El-Nasr, and R. Buehrer, "A framework for the power consumption and ber performance of ultra-low power wireles swearable healthcare and human locomotion tracking systems via uwb radios," in *Signal Processing and Information Technology (ISSPIT), 2009 IEEE International Symposium on*, dec. 2009, pp. 322 –327.

[21] D. Walsh, S. Capaccio, D. Lowe, P. Daly, P. Shardlow, and G. Johnston, "Real time differential gps and glonass vehicle positioning in urban areas," *Space Comms.*, vol. 14, no. 4, pp. 203–217, 1997.

[22] M. Sahmoudi and M. G. Amin, "Robust tracking of weak gps signals in multipath and jamming environments," *Signal Process.*, vol. 89, no. 7, pp. 1320–1333, 2009.

[23] R. Ennaji and M. Boulmalf, "Routing in wireless sensor networks," in *Multimedia Computing and Systems, 2009. ICMCS '09. International Conference on*, april 2009, pp. 495 –500.

[24] J. Heo, J. Hong, and Y. Cho, "Earq: Energy aware routing for real-time and reliable communication in wireless industrial sensor networks," *Industrial Informatics, IEEE Transactions on*, vol. 5, no. 1, pp. 3 –11, feb. 2009.

[25] S.-C. Jung and H.-K. Choi, "An energy-aware routing protocol considering link-layer security in wireless sensor networks," in *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on*, vol. 01, feb. 2009, pp. 358 –361.

[26] S. Khokhar and A. A. Nilsson, "Estimation of mobile users trajectory in mobile wireless network: Framework, formulation, design, simulation and analyses," *International Journal on Advances in Intelligent Systems*, vol. 2, no. 4, pp. 387–410, 2009. [Online]. Available: http://www.iariajournals.org/intelligent_systems/

[27] D. Jain and V. Vokkarane, "Energy-efficient target monitoring in wireless sensor networks," in *Technologies for Homeland Security, 2008 IEEE Conference on*, may 2008, pp. 275 –280.

[28] L. B. Saad and B. Tourancheau, "Towards an optimal positioning of multiple mobile sinks in WSNs for buildings," *International Journal on Advances in Intelligent Systems*, vol. 2, no. 4, pp. 411 – 421, 2009. [Online]. Available: http://www.iariajournals.org/intelligent_systems/

[29] C. Ding, R. Fuji-Hara, Y. Fujiwara, M. Jimbo, and M. Mishima, "Sets of frequency hopping sequences: Bounds and optimal constructions," *Information Theory, IEEE Transactions on*, vol. 55, no. 7, pp. 3297 –3304, july 2009.

[30] M. Strasser, C. Pöpper, and S. Čapkun, "Efficient uncoordinated fhss anti-jamming communication," in *MobiHoc '09: Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*. New York, NY, USA: ACM, 2009, pp. 207–218.

[31] B. Niehöfer, A. Lewandowski, R. Burda, C. Wietfeld, F. Bauer, and O. Lüert, "Community map generation based on trace-collection for gnss outdoor and rf-based indoor localization applications," *International Journal on Advances in Intelligent Systems*, vol. 3, no. 1 & 2, pp. 1 – 11, 2010. [Online]. Available: http://www.iariajournals.org/intelligent_systems/

[32] S. Kondakci and G. Yilmaz, "Implementation and performance evaluation of the RSEP protocol on ARM and Intel platforms," in *SIN '10: Proceedings of the 3rd international conference on Security of information and networks*. New York, NY, USA: ACM, 2010, pp. 194–202.

[33] CCITT, "The directory authentification framework," Draft Recommendation X.509, 1987, version 7.

[34] S. Kondakci, "A high level implementation of the RSEP protocol," in *ISC'07:Int. Conf. on Information Security & Cryptology*, vol. 1. ISC Turkey, December 2007, pp. 63–69.

[35] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987. [Online]. Available: http://www.jstor.org/stable/2007884

[36] "ARM9 Processor Development Board," accessed June 2010. [Online]. Available: www.embeddedarm.com

www.iariajournals.org

International Journal On Advances in Intelligent Systems
ICAS, ACHI, ICCGI, UBICOMM, ADVCOMP, CENTRIC, GEOProcessing, SEMAPRO, BIOSYSCOM, BIOINFO, BIOTECHNO, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS, CLOUD COMPUTING, COMPUTATION TOOLS
issn: 1942-2679

International Journal On Advances in Internet Technology
ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING
issn: 1942-2652

International Journal On Advances in Life Sciences
eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO
issn: 1942-2660

International Journal On Advances in Networks and Services
ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION
issn: 1942-2644

International Journal On Advances in Security
ICQNM, SECURWARE, MESH, DEPEND, INTERNET, CYBERLAWS
issn: 1942-2636

International Journal On Advances in Software
ICSEA, ICCGI, ADVCOMP, GEOProcessing, DBKDA, INTENSIVE, VALID, SIMUL, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS, CLOUD COMPUTING, COMPUTATION TOOLS
issn: 1942-2628

International Journal On Advances in Systems and Measurements
ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL
issn: 1942-261x

International Journal On Advances in Telecommunications
AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA
issn: 1942-2601