COVER SHEET

Framework for Event Discrete Simulation (FEDS)

Siak Chuan Tan and On Wong †
Queensland University of Technology, Brisbane Australia
+61 3864 9579, Email: {a2.tan, wongo}@qut.edu.au

Abstract. Wireless Sensor Networks (WSN) have recently gained much momentum in research and development, its potential to provide valuable services in key areas like military surveillances and mobile applications have attracted much attention from government and commercial research organisations. Researchers however, might not have the resources to purchase large numbers of sensors to obtain concrete research findings. Thus simulators are being created in an attempt to mimic conditions in the real world, providing research conclusions with an acceptable level of accuracy.
Discrete event simulation is one way of building up models to observe the time based (or dynamic) behavior of a system. Existing discrete event sensor simulators, e.g. ATEMU, JProwler, SNetSim and SensorSim, have a tendency to be specific to hardware, not platform portable or even lack information on the flow of events. Different simulators will be reviewed in this article, some simulators are build to accept only MICA2 sensors models while others are either incomplete or has insufficient visual impact.
This article will introduce FEDS (Framework for Event Discrete Simulation) which is designed to simulate objects functioning in the real physical world, events occurred are treated like messages which are being sent to environment mediums (air or water) for processing and decision making. FEDS is a generic framework meant for discrete event simulations, it can be customised by developers to simulate many scenarios from shipping lines planning to sensor networks and the objects in FEDS themselves can be further customized. FEDS is also designed to allow easy tracing of events, users can easily trace the flow of events and better understand the details of operation. Being a java implementation, FEDs is designed to be scalable and portable.
This article will also look at how we can adapt a sensor network test bed to the proposed framework, messages are being sent from a sensor to another sensor in a sensor bed. A trace is also being created to observe the flow of events and how sensors react when messages are received/sent, this will allow better understanding of what has happened and can be used to prove a routes performance over others.

Keywords: Sensors, Discrete event simulation, Frameworks, Networks.

1. INTRODUCTION

Wireless sensor networks (WSNs) have gained much attention and there have been a steady growth in researches and development. The potential of sensor networks could range from simple environment aware systems to military implementation, which attracted researchers to this area of study. Sun have only recently released a new version of sensors which runs on Squawk Virtual machine, Squawk is a pure java environment which is recently developed for small mobile devices. This is only an example of the growing trend of researchers committing to improving or setting standards for wireless sensors.

The feasibility of implementing a pervasive network of small and low powered devices have became much better with the recent advancement in technology, sensors with onboard processors are now able to last longer with better power saving techniques or more efficient processor capabilities. The potential implementation of these sensors are almost non exhaustive only to be restricted by one's imagination. There are already plans for location aware systems, inventory systems, military surveillance, biological warfare and even traffic network management.

Large software or techniques are usually tested in simulators to obtain predicted results. Discrete event simulation is one way of building up models to observe the time based (or dynamic) behavior of a system. There are formal methods for building simulation models and ensuring that they are credible. The results can then be used to provide insight into a system and a basis to make decisions on [BAL96].

This article will discover the motivations behind the introduction of a simulation framework and subsequently, its implementation. The requirements of sensor networks will also be discussed to help identify the needs of a sensor simulator to support the motivations. Current simulators will then be reviewed for advantages and potential problems. The FEDS architecture is then introduced and explained, FEDS is a framework for simulation which can be adapted to a wireless sensor scenario. This paper will then be concluded and future work identified.

2. MOTIVATIONS

Researchers often have limited resources when it comes to sensor procurement as each sensor costs a significant amount of money. Building an effective sensor networks usually involves tens of sensors in order to be tested reliably, assistance on resources or support from industries are only viable with some solid evident result from prototyping. Thus researchers always look into simulators to simulate ideas or logics, results from simulation prototypes are usually the key to the next step of research or development.

Time is also another limited resource, while simulators have the ability to compress timeline. It is possible for simulators to speed up results and reduce potential errors in measurements by sensor components.

Control is much better with simulation as simulators have the ability to pause, inspect and debug. It is also possible for implementer to review, improve and rerun again immediately. Sensor variables can also be controlled easily with settings, these allows greater control and saves huge amount of time.

There are several simulations to date which are commonly used for testing in sensor networks, most of them lacks in the ability to control and customize effectively. This article attempts to introduce a common framework where researchers are able to have full control over what they implement, they are allowed to write their own energy source classes, routing strategies or others if they follow a certain set of rules.

Due to the diversity of researchers in the world, it is difficult to predict the preference of language which is used for simulation. Java for example is a very good platform for a simulation to be written in, it has multi platform compatible and it could be written even with a notepad. Unfortunately, there are not a lot of simulators to make use of its advantage. This means that additional steps or knowledge need to be learned before being able to just set up a simulation system, ignoring the fact that some programming will need to be done also for customisation.

Many simulators are being introduced, each to mimic specific sensor hardware or to adapt to specific scenario. Frameworks for these simulators are always different to each other, and thus it is not possible to perform collaboration between different researches. It is desirable for there to be a common framework for these simulations, this article aims to introduce a simulator which has a framework generic enough to be adapted by different researchers and yet specific enough for future collaboration between different implementation.

3. CURRENT SIMULATIONS

3.1 ATEMU

ATEMU [PBM04] (ATmel EMUlator) is a very accurate MICA2 wireless sensor mote emulator, which tried to achieve extensibility by implementing the model of the "air" simulating the medium in the physical world. ATEMU is able to support binary TinyOS code from MICA2 mote, which makes ATEMU a very good simulator to use if you are developing with MICA2 hardware platform. Unfortunately, this also means that ATEMU is restricted to work efficiently in MICA2 hardware platform only.

3.2 GloMoSim

GloMoSim [ZBG98] is a wireless sensor network build by UCLA in 1998. GlomoSim is written in parsec, a variant of C to support parallel programming extension. However, GlomoSim only allow fixed protocols, it is limited to P networks due to the low level design assumptions. Glomosim do not provide support for the environmental factor which makes the result of simulations questionable.

3.3 J-Sim

J-Sim [AWC05] is quite similar to what this paper wishes to achieve, J-Sim is implemented in java making it platform independent. J-Sim uses Jacl which is similar to NS2 which uses C++. J-Sim simulates event-to-sink protocols, this means that target nodes could only send data packets over the sensor channel where other sensor nodes can only receive over the sensor channel.

3.3 Prowler

Prowler [SIS06] is a probabilistic sensor simulator written in Matlab, it has a version build in java (JProwler). JProwler is build for MICA Mote hardware platform, which is running on Tiny OS. It also has a very efficient throughput, but it provides only one MAC protocol of tinyOS. JProwler has a very good visual display for its results too, showing the path of propagation.

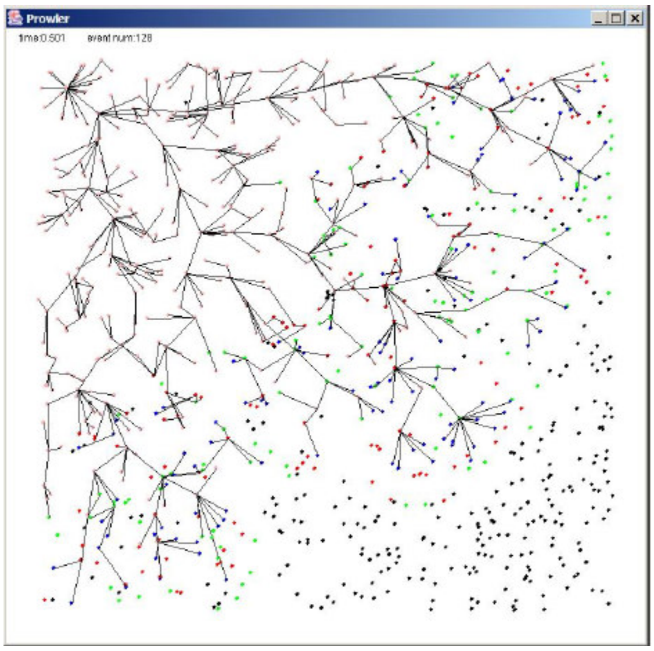


Figure 1: Prowler simulation system

3.4 NS-2

NS2 [INS06] is a very mature discrete event network simulation software written in 1989, NS2 is the defacto standard for network simulations, but it is limited to IP networks due to low level assumptions. NS2 supports 802.11 and single-hop TDMA wireless protocols. Most sensor modes uses a standard IEEE 802.15.4 protocol.

4. REQUIREMENTS OF SENSOR NETWORKS

Before this paper introduces FEDs, it will show you the different requirements of a typical sensor network. It is important that the requirements are properly identified as it will help to understand potential problems while planning for a simulator's framework. Coincidentally, the requirements can easily be mapped onto the network OSI layers.

4.1 Hardware Resources (Layer 1 – Physical)

Hardware Resources is a basic requirement for a sensor network, it is also related to OSI layer 1's physical layer because hardware's physical and electrical characteristics are involved. Implementing a sensor networks will require a large number of sensors to create a test bed, the numbers however usually adds up to a significant amount of cost. Thus there is always reluctance to obtain sensors immediately especially when the theory or prototype is not tested or proven at all.

Due to the potential cost, this article attempts to remove this layer by replacing it with a simulator, this simulator should have provision for handling the objects (sensors) in which details will be discussed in the later chapter.

4.2 Reliability (Layer 2 – Data link)

Sensor networks have their own purpose and applicability, these applications are dependent on the ability to communicate and send messages from the source to the destination sink [PS03]. The constructs of a reliable message delivery system is just what data link layer in the OSI aims to achieve, this layer is responsible for an error free transmission by means of synchronization, error and flow control. Some of the protocols currently used include CSMA/CD, 802.5 Token bus and FDDI [RAD00].

As reliability is caused by the environment medium like air or water, the simulation framework will need to have a medium for the objects to communication with each other. This medium will be governing the laws of the physical world, i.e. attenuation, deflection or others.

4.3 Communication Framework (Layer 3 – Network layer)

Although messages can be assured a reliable passage from source to destination, there must be a destination to start with. Routing and energy conservation is the most important factor for this layer [BJ05]. There are many protocols implemented including RIP, Rumor routing, Directed diffusion and more.

The simulator should allow objects to decide where is it sending communicating packets to, and allow the object to handle the method of sending too. This will allow the objects to be responsible for their own communication method.

4.4 Security (Layer 5 – Session)

Security is very important for sensors as wireless transmission are broadcasted to the environment, any intruder can easily snoop around and obtain information by sniffing alone. Thus messages must be well encrypted and destination reached, there should be mechanism to prevent or deter hacks of different kinds.

Security is usually handled by logic but supported by a protocol in any networks, and thus simulators are expected to allow different protocols to be tested for different security strategies.

4.5 Goal and Objective (Layer 7 –App)

Objects will all require having a purpose in simulation, or else there won't be a need to include them in. This simulation framework should allow objects to have an objective, which can range from a sensor (object) listening to seismic activity (objective) to a robot (object) trying to reach the finishing line (objective). There should even be provision for cloning or transportation of application agents via wireless transmission.

5. FRAMEWORK FOR EVENT DISCRETION SIMULATION (FEDS)

FEDS describes a set of framework which is similar to the physical environment itself, where there are objects which interacts with each other and the environment which acts as the medium. There should also be a trace capability to be handled by the simulator, where all 'traffic' in and out of an object will be captured by the trace.

In view of the different simulators created, FEDs is designed to be very scalable. FEDS's can be applied from sensor network scenario to shipping network scenario or from High level application layer in application classes to low level data layer in environment classes. The goal is for user to be able to increase the objects' ability (i.e. adding sensor's component) or increase the simulator's scope (i.e. creating 'water' environment).

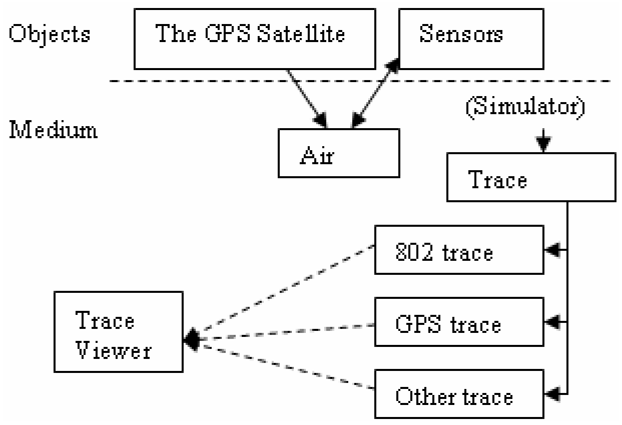


Figure 2: Backbone framework of FEDs.

Sensors are the objects in this context, and these objects follow a small framework by itself. This is important for FEDS as it will allow simulation of different scenarios too in the future instead of just wireless sensors. Objects in the simulator have the following properties.

5.1 Priority

The object's priority determines the speed or efficiency of the object in the simulator, which equate to the CPU of the sensors itself. One is free to set his/her implementation of the speed as long as the priority interface is implemented.

5.2 Power source

The object's power source determines the amount of energy the object has to perform operations, which is similar to a typical AA battery attached to the sensor. Just like the Priority, you can easily write your own power source as long as you implement the power source interface. Power is not restricted to just AA batteries, but it has a more generic representation from nuclear power plants to watch batteries.

5.3 Application

Every object will have an objective or goal, and there must be an application running to fulfill it. It could be a recognition program trained to recognise a particular wave pattern, and to send this information to a listener. Application can be easily written and implemented in the sensors as long as the application class is implemented, application migration of sensors are also possible in the future.

5.4 Components

Extra components can also be added to the sensors, sensor boards could be added to the sensors to provide additional capability. But take note that extra components equates to extra drain to the power source. You can easily create your own sensor component as long as you implement the component class.

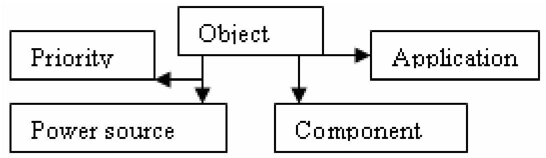


Figure 3: Framework of an Object in FEDs.

So a typical sensor will look like fig. 3 in comparison from the framework in figure 2.

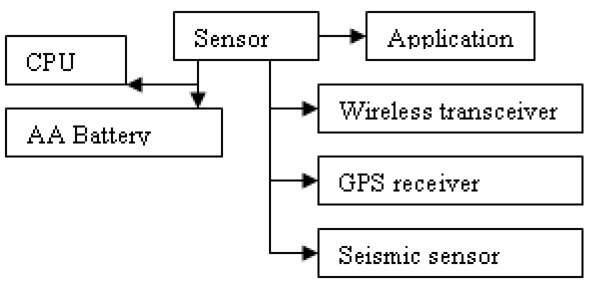


Figure 4: An implementation of a Sensor object.

5.5 Communication with environment

Air is the medium in this context, the environment air is expected to handle all transmission/event and determine which objects are eligible to receive the transmission/event. For example, the air will decide the recipient sensors from the sender base on the signal strength, range and surrounding objects (see fig. 4). It is important to note that message received is not restricted to just wireless transmission by sensors, there can be GPS transmission by satellites or other events which uses air as a medium to transport itself.

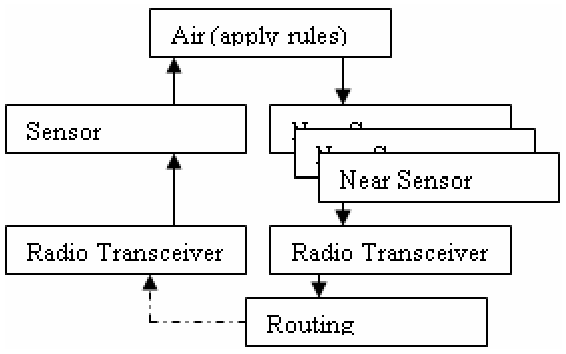


Figure 5: Flow of FEDs in a transmission of sensor via environment air.

5.6 Traceability

FEDS also take trace into consideration, and you are allowed to create your own trace as long as you follow the Trace interface. It is important to follow the trace interface for the trace viewer to display the result accurately. Note that this is an event discrete simulator, and thus trace display is solely on message transmission timeline.

6. IMPLEMENTATION

An implementation of the framework is created, this simulator simulates X sensors in a simulator space. Each sensor is being attached a wireless transmitter and a GPS receiver, a simple GPS satellite is also being added to the simulator. The following illustrations will demonstrate how the implementation of FED works.

Sensors are firstly created with a wizard framework, sensors to be created could have a choice of diffrent batteries and components. Due to the nature of the wireless sensors, this implementation assumes that all wireless sensors have a wireless transceiver attached and a routing strategy chosen (broadcast route).

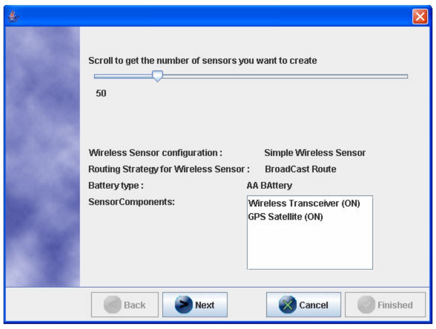

Scroll to get the number of sensors you want to create
50
Wireless Sensor configuration : Simple Wireless Sensor
Routing Strategy for Wireless Sensor : BroadCast Route
Battery type : AA BAttery
SensorComponents: Wireless Transceiver (ON) GPS Satellite (ON)

Figure 6: Creating of sensors in implementation.

A trace is then created, from a source to destination. Note that different routing strategies will require different settings. And these could be customized by the user themselves.

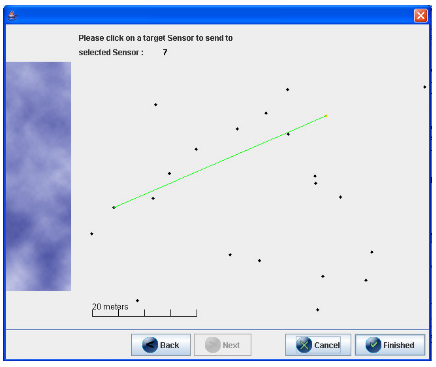

Please click on a target Sensor to send to
selected Sensor : 7

Figure 7: Creating of trace in implementation.

With the simulator running, you can opt to view the trace are several modes: 'Accepted Packets Only'/ 'All packets' / 'print text'. As all wireless transmissions are of broadcasting nature, there will be rejects either due to back propagation or invalid route. And thus we need to specify which transmissions are accepted (green link) and which ones are rejected (red link).

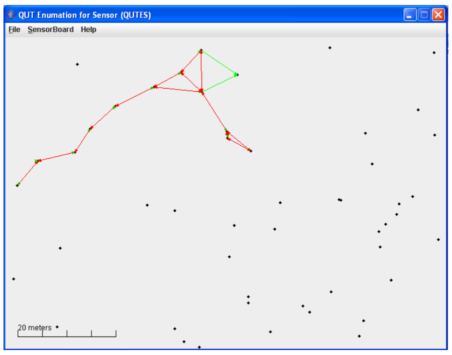
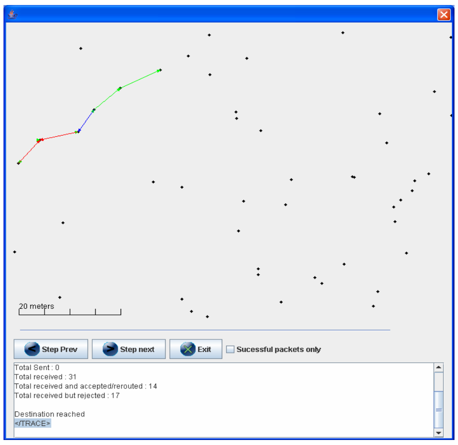


Figure 8: Simulator running trace.

Traces can also be inspected in details, every packet sent can be viewed in steps. There are two modes in viewing too, user can view accepted packets only or all packets bring broadcasted. This is great for understanding the routes and how they behave. The summary of packets consisting of number of send, number of accepts, number of reject and other information are displayed. These information are very useful to chart out results and comparisons in terms of effectiveness.



```
Total Sent : 0
Total received : 31
Total received and accepted/rerouted : 14
Total received but rejected : 17

Destination reached
</TRACE>
```

Figure 9: Viewing activity step-by-step and summary of trace.

7. FUTURE WORK AND CONCLUSION

FEDS should look into security very seriously as the role of security is currently being heavily rested on the components (wireless transceiver) of the objects (sensors). It is possible for FEDS to incorporate security features into the current framework, this will reduce the load of components as security will then be handled by higher layers. It will also make writing of customized components easier.

A proper implementation of FEDS should be created, current implementation is a skeletal version only. It will only be better if this project can be placed in an open source area where other developer can contribute and write their customized components.

Some research and prototyping should be done on FEDS to prove its real scalability by implementing other scenarios which are also event discrete. Although current implementation and discussion are all on wireless sensors but FEDS is not limited to just sensors, its potential can expand to other boundaries. Looking at a shipping simulator, figure 9 and 4 with figure 10 and 5 respectively will show that both scenario are very different but they both fulfills the simulator framework's requirement (figure 2 and 3). The ships in this case have crews which affect its efficiency or speed, while the coal generator will ensure energy source's power. The objective could be an application to move the ship from one shipping port to another, components like propeller and radio transceiver and radio transceiver still transmits to the environment air, but the propeller might send messages to environment water instead. This will allow the 'water' to calculate the distance moved by ship, and even determine if collision occurred.

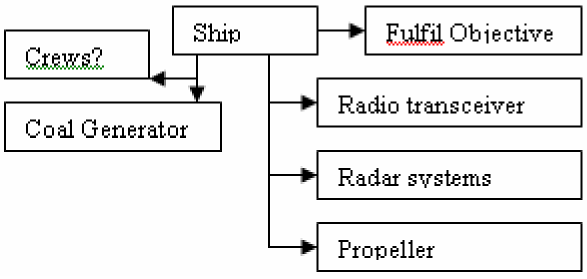


Figure 10: Alternative implementation of FEDs.

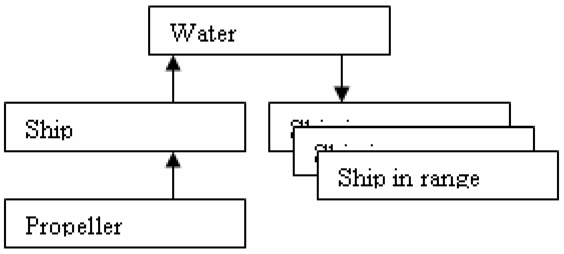


Figure 11: Flow of FEDs in an activity by ship via environment water.

To conclude, FEDS is a scalable simulator where developers can freely improve on as it follows a generic framework suitable for all event discrete scenarios. FEDS is currently being implemented as a wireless sensor network with the intention of studying how messages travel and routing efficiencies.

REFERENCES

[AWC05] A. Sobeih, W. Chen, C. Hou, L. Kung, N. Li, H. Lim, H. Tyan, H. Zhang, "J-Sim: A Simulation Environment for Wireless Sensor Networks", In Proceedings of the 38th Annual Simulation Symposium (ANSS-38 2005), 4-6 April 2005, San Diego, CA, USA.

[BAL96] P.Ball, "Introduction to Discrete Event Simulation", 2nd DYCOMANS workshop on 'Management and Control : Tools in Action', pp. 367-376, 15th - 17th May 1996, Algarve, Portugal.

[BJ05] N. Bulusu and S. Jha, "Wireless Sensor Networks", Artech House Publishers, July 31, 2005

[INS06] Information Sciences Institute. The network simulator - ns-2. Retrieved from the world wide web,http://www.isi.edu/nsnam/ns/, 29 Mary 2006.

[PBM04] J. Polley, D. Blazakis, J. McGee, D. Rusk , J.S. Baras, M. Karir, "ATEMU : A Frine-grained Sensor Simulator", In proceedings of the First IEEE Communications Scoiety on Sensor and Ad Hoc Communications and Networks (SECON), 4-7 October 2004, Santa Clara, CA, USA.

[PS03] S. Park and R. Sivakumar, "Sink-to-sensors reliability in sensor networks", SIGMOBILE Mob. Comput. Commun. Rev., ACM Press, 2003, 7, 27-28

[RAD00] RADCom Academy, "World of Protocols Poster", Technical papers, RADcom Academy, Tel-Aviv , Israel, 2000.

[SIS06] Institude for Software Intergrated System. Jprowler. Retrieved from the world wide web, http://www.isis.vanderbilt.edu/Projects/nest/jprowler/, 28 May 2006.

[ZBG98] X. Zeng, R. Bargrodia, M. Gerla, "GloMoSim: a Library for parallel simulation of large scale wireless networks", In proceedings of the 12th Workshop on Parallel and distributed simulations (PADS), May 26-29, 1998, Banff, Alberta, Canada.

AUTHOR BIOGRAPHIES

Siak Chuan Tan (Andrew) is a PhD student under the scholarship from SAP in School of Software Engineering, Queensland University of Technology, Australia. He is researching on routing techniques in sensor networks and event discrete simulations. Andrew has also received a scholarship for his honours degree from QUT and his academic achievements includes diploma with merit, degree with distinction, scholarship for honours degree, selection into the accelerated honours scheme, student excellence award for honours degree and scholarship for PhD. Andrew has also perform duties of a lecturer and tutor in QUT, his teaching and research interests includes web applications, wireless technologies, sensor networks, neural networking, mobile devices and location aware systems. His email address is a2.tan@qut.edu.au or http://sky.fit.qut.edu.au/~tan9/

On Wong is a lecturer in, School of Software Enginee ring, Queensland University of Technology, Australia. He had been working in the IT industry for more than 10 years before joining Queensland University of Tec hnology as a postdoctoral fellow in 1999 and as a lect urer in 2000. While in the industry, he worked on a number of projects involving system architecture and d esign of real time transaction systems, load balancing system, fault tolerant systems, and network administrati on system. His recent work experience was with a We b-based e-commence software Development Company, where he designed and developed Internet based order transaction processing system and payment technology system. Since joining QUT, Dr. Wong had been rese arching in the area of component software technologies , Web Services, agent system, mobile technologies and location-aware system. His email address is <o.wong @qut.edu.au>