Fuel Efficiency control through battery interruption using an artificial intelligent agent in a motorcycle

Abhijeet Bajpayee
M.tech. 2nd Sem (CSE) Rungta College of Engineering Bhilai (C.G)

Ashish Mishal
M.tech. 2nd Sem (CSE) Rungta College of Engineering Bhilai (C.G)

Sipi Dubey
Dean, Rungta College of Engineering Bhilai (C.G)

ABSTRACT
Though mechanical Engineers have already provided many techniques to improve performance of engine, here we are going to propose a new technique which will be electronically influenced .A processor which we called agent would be capable of controlling engine load and its fuel consumption by judging condition of motorcycle, condition of roads, speed and distributing the load between Battery and engine itself. This will also tell you how much kilometers your bike can go with current quantity of fuel. Every motorcycle has an average which we called as overall average of our bike. Here agent will be capable of showing you range of average, at elevated hilly roads, at free or empty roads so that it will help us to identify the most optimal path.

Keywords
Critical speed, agent, metaXML.

1. INTRODUCTION
With the fusion of Electronics, Mechanical and Computer Science Stream, the world is witnessing revolutionary Changes in the forms of various devices and machines. With the affordable rates of electronic components, in near future, motorcycle will be "electronify" or you can say computers will be deployed on wheels [1] which will open new field of research challenges in front of us such as vehicular adhoc network [1], traffic analysis and optimization etc. Lots of modifications have been done in terms of mechanical parts and techniques to minimize the fuel consumption as fuel is a conservative source of energy and it also emits CO_2 which is the major factor for global warming. However increase in number of vehicles in road has no impact on decreasing pollution instead of that its emission rate has increased tremendously. Because of this dilemma Lots of Companies are searching for substitutions for petrol and one of best solution provided by Electrotherm India Ltd [5] to use of Battery engines instead of petrol engines, they are light weighted and can go with max speed of 30 km/h which is very suitable for riding within a city. There successful launch has given us and idea to assemble both petrol engine and Battery engine in a single bike .When we ride in city we can run it on battery and for long tour petrol engine will be ON. That will be very advantageous as there is no emission of CO_2 in metro cities and it will save petrol and increase durability of petrol engines.

2. CONCEPT AND SCOPE

2.1 Concept
Our Concept has just exploited the idea of electric bike [5] and extended it, which is described now. It's a general observation that bikes inside cities travels in between 10 to 15 km a day. And due to traffic their speed range is 25- 30km/h. And we know that petrol engines always give better performance at high constant speed. Hence for long drive we prefer petrol engines. So we are going to assemble both the engines and activate battery engine first then when speed is going beyond the 25 or 30 km/h it off the battery and activate the petrol engine. After activating petrol engines it will start recording metadata which is described in Approach of implementation section.

2.2 Scope
In this paper we are giving more emphasis on our programming construction methodology and output analysis format. Interfacing of sensors and assembling of engines is treated here as a part of introduction only.

3. METHODOLOGY
Our system will consist of Input –Agent (Controller).

3.1 Input
We will get (F) Suspension Sensors to provide roughness of road, Infrared sensors (E) will provide slope of road, (S) Speedometer, rate of change of fuel quantity (C).

3.2 Agent
Our processor on (TINI Board ,will going elaborate it below) [2][7] which would be developed in embedded java will take the input binary signals in an interval of seconds and then parse those input in the format of XML data file which we are giving name metaXML. This metaXML file will help us to calculate the dynamic Averages of Bikes as per road conditions and using that average it will calculate the prediction of distance from Current quantity of fuel. All this operations will be done only when speed of the bike is above 25 km/h which we called Critical speed, because below that speed, bike will run on battery generated electricity engine. Hence no need to analyze petrol consumption. Our processor will always run a thread [3] which will utilize a polling technique [4] to detect whether speed of bike exceeds critical speed or not so that it will start petrol engine if speed crosses critical speed.

3.2.1 TINI BOARD: -
It's an embedded java microcontroller [2] in which we need to deploy our bytecode i.e. .class files and sees the execution results. TINI is the size of a 72-pin memory SIMM module, has a hybrid 8/32 CPU (DS80C390 - backwards compatible with 8051), 512K/1M non-volatile (battery-backed) RAM, 512K flash ROM, 2x serial, 2x CAN, 1-wire, parallel, real-time clock, and runs Slush (with a command shell similar to Unix), a Java virtual machine (JVM) and Java API.Software includes FTP server, TELNET server, TTY server and HTTP server. The CPU in both boards is a Dallas Semiconductor DS80C400 Network Microcontroller. Each module plugs into an evaluation board that contains components and connectors for a power supply, RS-232 serial port, and Ethernet networking. The TStik has an on-board Ethernet transceiver, while the DSTINIm400's Ethernet transceiver is on the evaluation board (the DSTINIs400).Creating the Device Controller software, compiling it, and deploying it to the TINI requires a variety of software tools.The TINI Software Developers Kit (SDK), available from Dallas Semiconductor, includes the TINIOS operating system and the Java Virtual Machine (JVM) that executes Java programs in the TINI. Dallas Semi also provides the JavaKit utility for configuring the TINI over its RS-232

serial port.To compile Java programs for a TINI, you can use just about any Java compiler and Java development system, including the compiler in the free Java Development Kit (JDK) from Sun Microsystems. I use Borland's JBuilder environment, which includes a compiler and graphical interface for developing. A free Personal Edition is available. The Ant build tool and the TINI-specific add-on TiniAnt are useful for compiling code and deploying it to the TINI.

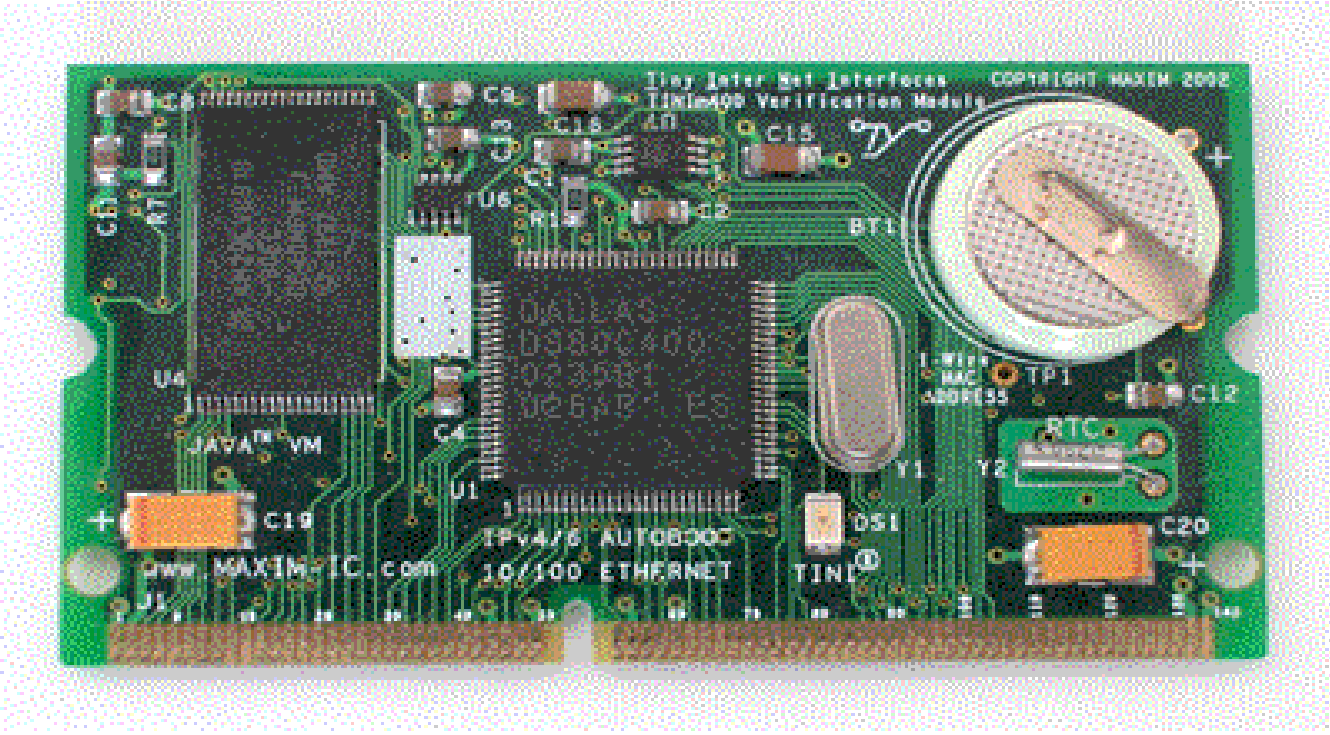

Fig 1: TINI Board Showing on Left Side chip having JVM and at middle Microcontroller to execute code

3.3 Algorithm for Calculations: -

For the above working conditions we have proposed a step by step process which would operate on data received from sensors. Get all the parameters in declared variables and keep that inside a loop until speed will not go below critical speed. Now check for speed, capacity, angle, frequency variations with respect to time and store it in a metaXML document. If speed is greater less then critical speed.

//Methods:-

Eval (speed ,capacity,frequency,timeclock)
Bflag← true;
i← 0;
While speed !< Critical Speed
do
storeS[i]←speed;
storeC[i]←capacity;
storeF[i]←frequency;
storeT[i]←timeclock;
if speed >25
Bflag←false;
Petrolengine ← true;
//This will call again the same process for new time set which may have difference with 1 sec.
Call: Eval (speed ,capacity,angle,frequency,timeclock)

4. OVERVIEW

To get the XML Output using java we need to understand some Buzzwords of XML.

5. OVERVIEW OF DOM

Document Object Model: [8] DOM is a platform- and language-neutral interface, that provides a standard model of how the objects in an XML object are put together, and a standard interface for accessing and manipulating these objects and their inter-relationships. The DOM is an interface that exposes an XML document as a tree structure comprised of nodes. The DOM allows you to programmatically navigate the tree and add, change and delete any of its elements. The DOM programming interface standards are defined by the World Wide Web Consortium (W3C). The W3C site provides a comprehensive reference of the XML DOM. DOM provides many handy classes to create XML file easily. Firstly, you have to create a Document with DocumentBuilder class, define all the XML content – node, attribute with Element class. In last, use Transformer class to output the entire XML content to stream output, typically a File.

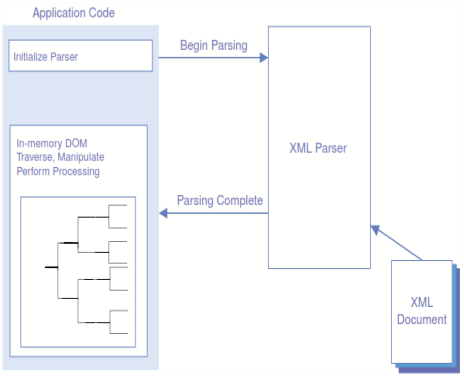

Fig 2: Using the DOM API.[9]

6. XML SCHEMA

An XML [9] Schema definition (XSD) is an XML-based grammar declaration for XML documents. XSD is itself an XML language. Using XSD, data constraints, hierarchical relationships, and element namespaces can be specified more completely than with DTDs.XML Schema allows very precise definition of both simple and complex data types, and allows types to inherit properties from other types. There are numerous common data types already built into the base XML Schema language as a starting point for building specific languages.

6.1 Databases with XML:-

[10] The connection between databases and XML is a logical one: Both store data in a structured manner. A very common use for XML is to take data stored in XML documents and move it into a database so that the data can be accessed and manipulated by an application. All the major database systems — from Oracle to Microsoft SQL (Structured Query Language) Server and beyond — have XML utilities that help you work with XML in the context of the database. Another common use of databases with XML is to let the database serve as an index to the content described with XML. For example, an online article archive uses XML as the format for describing articles. When a new article comes into the system, the tool that pulls the article into the archiving system populates a database with some basic data about the article, such as the author, date, title, and topic. The rest of the article is stored as a flat text file on the system's hard drive for later access. You can then use the database to search for articles by the information stored in the database: author, date, title, and topic.

[11] It's possible to store and retrieve XML documents in a relational database even if that database does not boast XML features. The secret lies in careful use of Java (the database does have to be JDBC-compliant) and a third-party middleware framework named XML-DBMS, which acts as the glue between the relational database and the XML document flow.In general, there are two common approaches to mapping XML to databases: a table-based mapping and an object-relational (object-based) mapping. Both are commonly used as the basis for software that transfers data between XML documents and databases, especially relational databases. An important characteristic in this respect is that both methods are bidirectional, allowing XML documents to be both stored and retrieved. With object-based mapping, XML-DBMS maps the XML document to the database according to an object-relational scheme in which element types are generally viewed as classes, and attributes and PCDATA are treated as properties of those classes. This models the XML document as a tree of objects that are specific to the data in the document and then maps these objects to the databaseXML-DBMS does not need any specific database, it must simply understand standard SQL and have a JDBC driver (or ODBC driver with a bridge). This framework can be used in applications that want to search across or merge information from diverse sources, because relational database schemas can be easily built by XML DTDs and XML Schemas, and there are tools for converting DTDs and Schemas to map-files. It is also possible to build data-driven applications such as a CMS or CRM system. It is a good example of an XML-to-DBMS middleware product that can be useful in integration of XML-featured and non-XML-featured systems.

6.2 STEPS TO GET XML OUTPUT

Our XML schema will be defined below which will have following parameters:-

```
<xsd:schema
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

     <xsd:element name="DataSet"
        type="xsd:decimal" minOccurs="1">
      <xsd:attribute name="quantity"
        type="xsd:decimal"/>
      <xsd:attribute name="distance"
        type="xsd:decimal"/>
      <xsd:attribute name="frequency"
        type="xsd:decimal"/>
     </xsd:element>
</xsd:schema>
```

This XSD will be use whenever any validation for metaXML has to be done. Next we have to associate this Schema with our metaXML file which can be done by mentioning url pattern for finding location of XSD. Our programs will take all inputs from sensors and parse it to a metaXML file. This file will be transferred to our PC using USB socket and then we will use it to analyze data.

7. EXPERIMENTAL INVESTIGATION

7.1 Prototype Testing

We have developed a prototype based on algorithm in Java platform .We made two programs for testing purpose one will take inputs and assign those values to variables which will be save in metaXML file in the form of attributes. Next program is based on file handling in which we transferred metaXML file from chip to our PC with the help of USB socket. Now those metaXML files are imported in Excel Sheets where we have created a graph which is shown below. For this XML we have prepared our own sample data to test proper working of program and its response time.

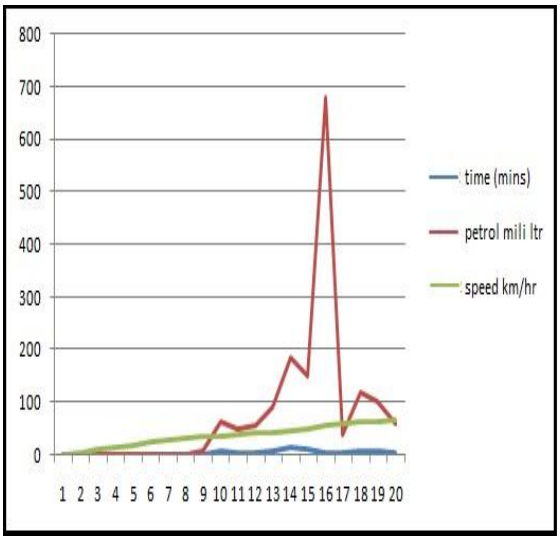

Fig 3: When agent is off

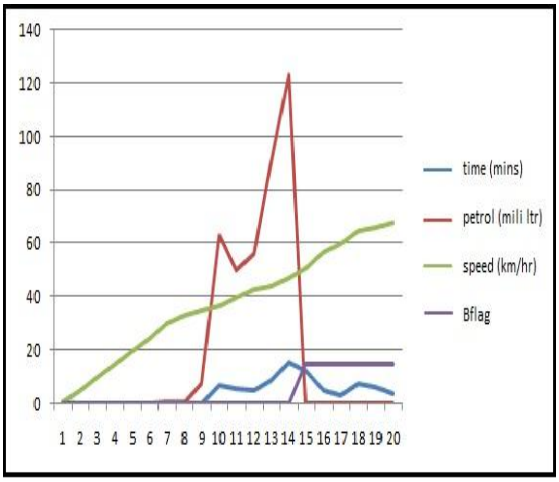

Fig 4: When agent is On

8. FUTURE EXTENSION

Our Selection of chip and XML format is not random. We have considered here about scalability of this project. This chip is capable of extending the configuration without affecting present design in it. Our next approach will be Ad hoc path creation in villages where GPS signals not available and GPS interfacing with XML as GPX format for city implementation. Next will be [1] Vehicular networks which are very likely to be deployed in the coming years and thus become the most relevant form of mobile ad hoc networks.

9. CONCLUSION

In this paper we have discussed and focused on designing Methodology of an agent that must be capable of providing information about bike performance and improve engine efficiency by battery interruption when we drive inside city.

10. REFERENCES:-

[1] Securing vehicular ad hoc networks By Maxim Raya and Jean-Pierre Hubaux in Journal of Computer Security 15 (2007) 39–68 IOS Press.

[2] www.goosee.com/tini

[3] Thinking in Java by Bruce Eckel and Complete Reference of Java by Herbert Schildt

[4] math.hws.edu/javanotes/c1/s2.html

[5] www.yobykes.in

[6] Liquid cooled pmbldc hub motor by Electrotherm

[7] www.lvr.com/devicecontroller.htm

[8] www.roseindia.net/xml/dom

[9] Java 2EE and XML Development by Kurt A. Gabrick And David B. Weiss

[10] XML for Dummies By Lucinda Dykes and Ed Tittel

[11] Bridge your database to XML with Java and XML-DBMS By Peter V. Mikhalenko July 22, 2003