Energy Management System

A modular solution for power monitoring and management for homes and small businesses.

Data Storage and Database Requirements

The storage and handling of data received by the main unit.

April 7, 2015

(Author) Andrew Cope – ajc4630@rit.edu
Jacob Lauzon – jfl4577@rit.edu
Donald MacIntyre – djm4912@rit.edu
Ryan M^{c}Laughlin – rpm6651@rit.edu

Table of Contents

Overview

The Energy Management System is a system that is designed to allow for homeowners and businesses to easily monitor and manage their power usage and consumption. The system consists of two main components, the main hub and many separate outlet modules. The main unit is installed at the breaker panel of the home or business and monitors overall power consumption. The outlet modules are replacement outlets for the home or business that monitor the power consumption of only that outlet. The main unit collects the usage data from all of the outlet modules and compiles the information. The Energy Management System also has a web application where all of the usage data can be viewed in graphical form. From the web application, the user can also control whether each outlet module is on or off and can limit the power of each outlet. In addition, a schedule can be created to automatically turn outlet modules on and off at certain times of the day. The outlet modules are replacement outlets for the home or business that monitor and control the power consumption of only that outlet, as directed by the main unit.

Database

The power consumption readings from each of the outlet modules of the Energy Management System will need to be stored, so that they may be accessed at any time through the web interface. In order to accomplish this functionality, a database and database management system are necessary. This document covers the risks related to storing the data, as well as the mitigation of said risks.

Outlet Module Data

The Energy Management System will be operating in homes and small businesses, and will allow for the naming and addressing of individual outlet modules. Table 1 below describes the essential information of each outlet module that will be stored in the database.

Table 1 – Required Outlet Module Information

ID	The internal ID used by the database for the outlet module.
Outlet Module Name	The text string that describes the outlet module.
Outlet Module Address	The logical/physical address of the outlet module, used to communicate with and/or send commands.

Outlet Module Reading Data

The outlet modules will send four data values to the main unit: a scale, a voltage, a current, and a power factor. This information will be stored along with a reading ID and a time. Table 2 below shows the essential information of each outlet module reading that will be stored in the database.

Table 2 – Required Outlet Module Reading Information

Reading ID	The internal ID used by the database for the reading.
Outlet Module ID	The outlet module ID that the reading comes from.
Scale	The scale of the data that is stored.
Voltage	The voltage differential.
Current	The measured current.
Power Factor	The power factor.
Time	The current time.

Data Compression

The outlet module readings will default to a period of 15 seconds between each reading. If the user accesses the real time interface for a certain module, then that module will send readings every second. As time goes on, the older readings will be less important. For example, the 15 second or 1 second time intervals for readings 30 days ago are unnecessary and will take up valuable space. To save on space, the data will be averaged across a larger period of time.

Risk Specification

Below is the subset of system marketing requirements that are related to this particular component.

Marketing Requirements

1. The system shall accurately monitor power consumption.
2. The system shall allow for control of individual outlets.
3. The system shall provide intuitive visual representations of usage data.

Engineering Specifications

Table 3 – Engineering Requirements

Marketing Requirements	Engineering Requirements	Justification
1	A. The voltage, current, power factor, and a scale should be stored for each outlet reading.	Storing values such as voltage and current will result in a more accurate power calculation.
1,2,3	B. The outlet names, addresses, and readings will be stored in a database.	A database allows for ease of access to all of the outlet readings. The database will be accessed by the web interface to show the data.
1	C. A scale will be stored using	One byte will allow for a large number of

	1 byte.	possible scale values.
1	D. Current will be stored using 2 bytes of data.	Two bytes will provide sufficient accuracy when measuring current.
1	E. Voltage differential will be stored using 1 byte.	One byte of data will provide the necessary amount of accuracy for the voltage differential (above 100 volts).
1	F. Power factor will be stored using 2 bytes.	Similarly to current, two bytes will provide the necessary accuracy for the overall power factor.
2	G. A table in the database will store the names and addresses of each outlet module.	The user will be able to name each outlet, to allow for quickly identifying the outlet. The MAC Address is stored to send commands to a specific outlet.
3	H. After 30 days, average the readings into minutes or hours, to condense the information for that day.	The size of the main unit's storage is limited, so the "compression" of older data allows for the database to take up less space. The data will still be visual, but provide less resolution.
3	I. The current time will be stored with each outlet module reading.	The time will allow for the user to access specific periods of time to show the outlet module readings of a time range.

Risk Investigation

This section will cover the existing solutions for the necessary functionality of the component. Pugh analysis will be used to rank the solutions, and provide a rational explanation for the choices made. The chosen solutions will be explained further below.

Database Management System

Since the data is to be stored in a database, a database management system is needed. A DBMS allows for the creation, updating, querying, and administrating of databases. Several popular DBMSs include MySQL, SQLite, Microsoft SQL, and PostgreSQL. All four of these DBMSs use the SQL standard programming language to operate the database.

Microsoft SQL Server

Microsoft SQL Server is a relational database management system (RDBMS), developed by Microsoft. It is available only for Windows Server operating systems. SQL Server uses a commercial license, and is closed source. SQL Server is ACID compliant. ACID is an acronym for Atomicity, Consistency, Isolation, and Durability.

Microsoft SQL Server is available through a limited number of supported programming languages. SQL Server uses multiple locking techniques, depending on the transaction that is taking place. The lock can be performed on a single row, or an entire table.

SQLite

SQLite is a simpler SQL based RDBMS, developed by Richard Hipp. Like MySQL, SQLite is available on all major operating systems. SQLite is considered server-less, as it is more of a library than an actual application. SQLite is open source through the public domain license. SQLite also uses transactions through ACID compliance.

Because of SQLite's simplistic library nature, it is available in a wider range of languages than MySQL, but is more designed to be for general database applications. SQLite supports a variation of table-level locking, in which the entire file is write-locked until the transaction is finished. Multiple processes can perform Reading from databases at any time.

MySQL

MySQL is the most widely used open-source RDBMS, developed by Oracle. It is available for all major operating systems, like Windows, Linux/Unix, and OSX, and more. MySQL uses the GPL open source commercial license, as it is still developed by Oracle. MySQL is ACID compliant, when using the InnoDB storage engine. MyISAM storage engine does not require transactions, and is therefore not ACID compliant.

It is available through a wide variety of popular programming languages, as it is more emphasized and used with web-based applications. The InnoDB storage engine supports row-level locking, allowing writes and reads to take place while preventing changes on the locked row. MyISAM supports table-level locking, which prevents any writing changes to be made until the current query is over.

PostgreSQL

PostgreSQL is an object-relational database management system (ORDBMS), developed by PostgreSQL. Like MySQL, it is available on all major operating systems. PostgreSQL uses the BSD open source license. PostgreSQL is ACID compliant, and ensures that all transactions taking place are

PostgreSQL is available through fewer programming languages than MySQL, but a different subset than SQL Server. PostgreSQL supports both table-level locking and row-level locking, and appropriately uses the right lock based on the transactions that are taking place.

Database Storage Engine

There are several database storage engine types, all of them performing similarly to accomplish the same task: creating, reading, updating, and deleting data from a database.

Memory
As the name suggests, storage is saved in memory. Once enough changes have been made, the data currently in memory is flushed to the hard disk. Since the data is stored in memory, it can be accessed much quicker than any file-based storage engines.

InnoDB
InnoDB is a highly reliable, high-performance storage engine. It is ACID compliant, allowing it to use transactions to commit data. In the event of a power failure, there are rollback and crash-recovery capabilities to ensure data integrity. The support for row-level locking allows for one client to lock a specific row, while other clients may access other rows (or wait for the tow to be unlocked). This is ideal for write-heavy applications, as the data coming in can be written quickly without the need to wait for another row.

MyISAM
MyISAM is a storage engine that splits the database table into three separate files. As such, the engine supports table-level locking, which will lock the entire table from being written to until the current query is finished. This is ideal for read-heavy applications, as the data is only read and not written to.

Solution Selection

To determine the most appropriate solution for the necessary functionality, Pugh analysis tables were created. Each database management system (DBMS) is compared using several different aspects. Multiple solutions are possible, and would be decided upon at a later time, so long as enough information is presented to make a clear choice.

To analyze the solutions based on certain criteria, a rating of +1, 0, or -1 will be assigned to each of the solutions. The ratings will then be added up for each solution, and an ascending rank will be assigned to that solution based on the descending overall rating. A +1 rating signifies that the particular solution is acceptable or desirable for that criterion. A rating of 0 denotes that either the criterion is not applicable to the solution, or it would not affect the outcome significantly. A rating of -1 indicates that the solution is undesirable for that criterion, or the solution cannot meet that criterion.

Database Management System

Table 4 – Database Management System Solution Analysis

	Solutions			
Criteria	Microsoft SQL Server	MySQL	PostgreSQL	SQLite
Windows Server OS	-1	0	0	0
Linux OS	0	+1	+1	+1
Commercial License	-1	0	0	0
Open Source License	0	+1	+1	+1
Supports C/C++	0	+1	+1	+1
Supports Java	+1	-1	+1	+1
Supports Perl	0	+1	+1	+1
Supports PHP	+1	+1	+1	+1
Supports Python	+1	+1	+1	+1
Supports Ruby	+1	+1	0	+1
Concurrency	+1	+1	+1	+1
Sufficient Data Types	+1	+1	+1	-1
ACID Compliance	+1	+1	+1	+1
Total Rating	5	10	10	9
Rank	4	1	1	3
Consider Solution?	No	Yes	Yes	Yes

After applying a Pugh analysis on the four DBMS solutions, Microsoft SQL server was found to be undesirable for the component's need. This is based off the two primary criteria of Server OS support and Licensing. A Windows Server operating system would not be used in the main unit, and as such, the Microsoft SQL Server cannot be used on non-Windows-based operating systems. Also, a commercial license is needed in order to use the SQL Server, which would cost money and increase the price of the main unit.

SQLite is also a potential option, as it is a simpler DBMS. However, it lacks some of the data types supported by MySQL and PostgreSQL, making it a less desired solution, but still considerable. Its single file storage allows for a simplified backing up process by the user.

MySQL and PostgreSQL are the two highest-ranking solutions, tied at rank 1. They both possess similar features and capabilities that would allow for the desired functionality of the database and data storage. The web interface will easily be able to access the database, as both of these DBMSs have proper drivers for many programming languages. MySQL will be the chosen database management system, as it is simple and will be effective for the Energy Management System.

Database Storage Engine

Table 5 – Database Storage Engine Solution Analysis

	Solutions		
Criteria	Memory	InnoDB	MyISAM
Reliability	-1	+1	+1
Row-level Locking	+1	+1	-1
Table-level Locking	0	0	+1
Read-Heavy Performance	+1	0	+1
Write-Heavy Performance	+1	+1	-1
Transaction Support	+1	+1	0
Storage Medium	-1	+1	0
Total Rating	2	5	1
Rank	2	1	3
Consider Solution?	No	Yes	No

After applying Pugh analysis, InnoDB is the most desirable storage engine to be used for the database. It has the most performance and reliability related features, such as transactions and row-level locking. Due to transactions, in the event of a power failure, the database will be able to recover any data commits that have not taken place. MyISAM is slightly less feature-full, but is still an option. Since transaction support is preferred, to ensure data integrity and reliability, InnoDB will be the chosen storage engine.

Risk Mitigation Design

This section details the specific design and functionality of the data storage and database management system that will be used in the Energy Management System.

Database Tables

Outlet Module Table

As mentioned in the overview, the outlet module table will hold the information pertaining to each outlet module. To better allow control of the outlet modules, the name of the outlet and its logical/physical address will be stored. The web interface will allow the user to change the name of the outlet module, while the logical/physical address will be internally stored when communicating with the outlet. Table 6 below shows the required table columns, along with their data type and size requirements.

Table 6 – Outlet Module Table Data Types and Sizes

Column Name	Data Type	Maximum Data Size
Outlet ID	TINYINT / SMALLINT	1 Byte / 2 Bytes
Name	VARCHAR(100)	101 Bytes
Address	BINARY(8)	8 Bytes
Frequency	TINYINT	1 Byte
Real_Time	BIT(1)	1 Byte
	Max size per row:	112 - 113 Bytes

Given that the average household contains 50-60 outlets, only one byte is sufficient to hold the outlet module ID. In the case of a business, where there may be several hundred outlets, a small integer (2 bytes) is sufficient to hold the outlet module ID.

For the purpose of naming an outlet module, 100 characters will be the imposed, which can be changed if necessary. For example, “Family Room – TV” would be a typical outlet module name, which uses less than 100 characters. MySQL requires L + 1 bytes, where L is the actual length of the string.

The address of the outlet module will either be a logical or a physical address. In either case, the maximum address size will be 64-bits, or 8 bytes. The address will be stored in binary, to allow it to be converted if necessary.

The frequency will be a tiny integer (0-255), which dictates how often the outlet module will send data to the main unit. By default, this value will be 15, meaning that every 15 seconds, the outlet will send data. In the case of real time viewing, the frequency will be modified, and the real-time bit will be set to ‘1’ to signify that outlet module(s) is sending real time data.

Outlet Reading Table

The outlet reading table will hold all the information that is sent by the outlet modules to the main unit. The four main pieces of information (scale, voltage, current, and power factor) are sent by the outlet module to the main unit. The remaining values, reading ID, time, and the real time bit, will be determined at the time of insertion to the table. Table 7 below shows the data types and sizes that will be used for each reading.

Table 7 – Outlet Reading Table Data Types and Sizes

Column Name	Data Type	Maximum Data Size
Reading ID	INT	4 Bytes
Outlet Module ID	TINYINT/SMALLINT	1 Byte / 2 Bytes
Scale	TINYINT	1 Byte
Voltage	TINYINT	1 Byte
Current	SMALLINT	2 Bytes
Power Factor	SMALLINT	2 Bytes
Time	TIMESTAMP	4 Bytes
Real_time	BIT(1)	1 Byte
	Max size per row:	16 -17 Bytes

As stated before, there would only need to be 1-2 Bytes to hold the outlet module ID, based on the number of outlets needed. This ID is used internally, which is connected to the outlet ID in the Outlet Module Table.

The scale, voltage, current, and power factor values are the measured values that come from the outlet module. Although the actual measurement size transmitted may only be 3-14 bits, the integer data type used by MySQL is only in multiples of 8 bits, or 1 byte. Because of this, the values are rounded up to the next highest byte multiple.

The time column is the current timestamp at the time of insertion. This is used to display a range of data in the web interface, as well as in data compression mentioned later on. It is stored using the format of "YEAR-MONTH-DAY HOUR: MINUTE:SECOND", which is converted from the user's time zone to UTC for storage, and back to the current time zone during retrieval.

Similarly to the Outlet Module Table, the real-time flag dictates whether or not the reading is a real time value. Every 15th reading will have the real time flag set to '0', as it will be considered a normal interval reading. This will be explained more in the data compression section below.

Database Size

Although MySQL puts some overhead data in the database, that will be considered miniscule against the amount of data needed for the outlet modules and readings. Table 8 below shows the overall maximum size of the Outlet Module Table for various numbers of outlet modules in the household or business.

Table 8 – Outlet Module Table Size for Various Quantities of Outlet Modules

Number of Outlet Modules	Maximum Size for Outlet Module Table row:	Total Estimated Size
1	112 Bytes	112 Bytes
10	112 Bytes	1.09 KBytes
50	112 Bytes	5.47 KBytes
100	112 Bytes	10.94 KBytes
300	113 Bytes	33.11 KBytes
1000	113 Bytes	110.35 KBytes
10000	113 Bytes	1.08 MBytes
65535	113 Bytes	7.06 MBytes

The last row in Table 8 is the absolute maximum size for the Outlet Module Table, based on the small integer limitations of 65,535 ID values. Overall, the amount of disk space necessary for storing the outlet modules is trivial, when compared to the size of the Outlet Reading Table.

The size of the Outlet Reading Table is entirely dependent on how many outlets there are, as well as how frequently the modules are transmitting data. Tables 9 and 10 show the amount of data for one outlet module and multiple outlet modules, respectively, for various lengths of time and reading intervals.

Table 9 – Outlet Reading Table Size for One Outlet Module

Update Interval	Max Bytes per reading	Table Size Per Day	Table Size Per Year	Table Size Per 5 Years
Every 1 Second	17 Bytes	1.40 MBytes	511.27MBytes	2.50 GBytes
Every 15 Seconds	17 Bytes	95.63 KBytes	34.09 MBytes	170.43 MBytes

Table 10 – Outlet Reading Table Size for Multiple Outlet Modules

Number of Outlet Modules	Update Interval	Max Bytes Per Reading	Table Size Per Day	Table Size Per Year	Table Size Per 5 Years
10	1s	16 Bytes	13.18 MBytes	4.70 GBytes	16.99 GBytes
50	1s	16 Bytes	65.92 MBytes	23.50 GBytes	84.98 GBytes
100	1s	16 Bytes	131.84 MBytes	46.99 GBytes	169.96 GBytes
500	1s	17 Bytes	700.38 MBytes	249.64 GBytes	902.91 GBytes
10	15s	16 Bytes	0.88 MBytes	0.313 GBytes	1.133 GBytes
50	15s	16 Bytes	4.39 MBytes	1.566 GBytes	5.665 GBytes
100	15s	16 Bytes	8.79 MBytes	3.133 GBytes	11.331 GBytes
500	15s	17 Bytes	46.69 MBytes	16.64 GBytes	60.194 GBytes

As seen above in Tables 9 and 10, it is possible to read the outlet modules once per second. However, it is unfeasible to do so, as the data storage size grows considerably large with more and more modules. To mitigate the risk of storing too much data, readings will occur every 15 seconds, unless the user selects to see the real-time data for a specific outlet module(s), in which case the readings will occur every second with a flag. When real time monitoring is taken into account, the reading table size will be larger, depending on how long the user is viewing the real time data.

Data Compression

It is unlikely that users will need to see the 15 second interval readings for outlet modules from 30 days ago, or sooner. Therefore, the data stored is able to be compressed, by taking the average of the readings from each minute, or hour, and reduce the readings to one single reading. Table 11 below shows the Outlet Reading Table size before and after the data compression.

Table 11 – Outlet Reading Table Size Before and After Data Compressions

Number of Outlet Modules	Original Data Size		Average 4 Readings to 1 (1 Minute Interval)		Average 240 Readings to 1 (1 Hour Interval)	
	Table Size Per Day	Table Size Per Year	Table Size Per Day	Table Size Per Year	Table Size Per Day	Table Size Per Year
1	95.63 KB	34.09 MB	0.03 MB	8.02 MB	384 B	136.9 KB
10	0.88 MB	0.313 GB	0.23 MB	80.2 MB	3.75 KB	1.337 MB
50	4.39 MB	1.566 GB	1.09 MB	401 MB	18.75 KB	6.683 MB
100	8.79 MB	3.133 GB	2.19 MB	802 MB	37.50 KB	13.37 MB
500	46.69 MB	16.64 GB	11.7 MB	4.16 GB	199.22 KB	71.01 MB

As seen in the table above, averaging a certain number or readings into 1 reading will save a great amount of space. The reduction will take place once per day, and any readings past a certain date will be averaged into larger interval readings. In the case of real-time monitoring, the flagged rows will be deleted once the user has ended their session. This is due to the real-time, 1-second intervals being temporary for that user's session.

This mitigation allows for the main unit to come with less disk space installed (i.e. 4 or 8GB versus 16GB), which is essential to maintain data storage without user intervention. Some other potential mitigations are to allow the user to back up their readings to a local file or hard disk, or the potential use of a cloud-backup service. However, the Energy Management System will default to data compression over the other options.

As MySQL is open-sourced through GNU Public License, there are no patent infringements.

Parts List

The only necessary part for the data storage is the MySQL database, which is provided free through the GNU Public License. It is available for Linux operating systems. The operating system chosen for the main unit may not have MySQL installed by default, so an installation may be necessary.

Testing Strategy

There will be several tests performed, to ensure the entire database is working correctly.

Outlet Module Reading Simulation

1. Setup multiple clients to act as the outlet modules (Naming/Addressing).
2. At 15 seconds +/- 100ms, each client will send a reading to the main unit.
3. The main unit should insert a new row into the Outlet Reading Table, with the proper values and timestamp.
4. Verify the new readings match the sent values by each client.

Real-Time Outlet Module Reading Simulation

1. Set the real-time flag on one or two outlet modules to ‘1’.
2. The main unit will send the signal to the selected clients to send 1 second interval readings.
3. The clients will send a reading every 1 second +/- 100 ms.
4. The main unit should insert a new row into the Outlet Reading Table, with the proper values, timestamp and real-time flag.
5. Verify the new readings match the sent values by each client.

Outlet Module Reading for a Time Range

1. Through the web interface, select a time range
2. The database will query the readings and produce all readings from the specified start time to the specified end time.
3. Verify the readings fall within the correct time range

Request Real-Time Readings

1. The web interface will send a request to the database for a real-time update of a specific outlet module.
2. The database will update the frequency and real-time flag for the specified outlet module.
3. The web interface will send a request every second for the latest reading of the specified outlet module.
4. Verify that the outlet module has the correct frequency and real-time flag.
5. Verify that the outlet module readings have the real-time flag set to ‘1’.
6. Verify the values seen by the web interface are the correct values being stored.

Compress Readings Older Than 30 Days

1. Insert Readings into the database that are 30 days old.
2. Verify that the inserted readings are averaged into one reading, and that the others are deleted.

Delete Real-Time Readings after Session Close

1. Request a real-time monitoring of a specific outlet.
2. Wait for 30+ readings to populate the database.
3. End the web interface session cleanly.
4. Verify that the real-time flagged readings are deleted.

5. Verify that the outlet module frequency and real-time flag have been reset.

Uncertainties

Database Management System

There are no uncertainties with the chosen DBMS, MySQL.

Database Tables

Since the Outlet Reading ID is a 4 Byte (32-bit) integer, there are 4,294,967,295 readings that can take place before a reset/refresh would have to occur. In a larger business with, for example, 700 outlet modules, the minimum number of readings per year would be 1,471,680,000. This would result in just under 3 years before the reading IDs would begin to conflict. Doubling the size of the reading ID could mitigate this.

Data Compression

There are no uncertainties with the data compression design.

Appendix

Distinctive Features of SQLite

SQLite Data Types

Microsoft SQL Server Installation Requirements

PostgreSQL Data Types

Memory Storage Engine

MyISAM Storage Engine

InnoDB Storage Engine

MySQL Data Type Storage Requirements

ACID Model