Development of Web-Based Application for Shapefile Coordinate System Conversion for Malaysia

Elysonia Alim (1), Shahabuddin Amerudin (1)

[1] Department of Geoinformation, Faculty of Geoinformation and Real Estate, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia
Email: elysoniaalim@gmail.com; shahabuddin@utm.my

KEY WORDS: coordinate system, conversion, shapefile

ABSTRACT: It is difficult to process GIS vector data when they are not aligned with one another. The need for different coordinate systems rose from the fact that some coordinate systems are better fitted to describe the geographical phenomenon occurring in a specific area. However, even commercial software had been proven to have questionable accuracy in coordinate system conversions. The purpose of this study is to develop a web application capable of converting the coordinate system of a GIS data format such as a shapefile for Peninsular Malaysia. The web application named Coordinate Conversion Application (CCA v1.1) was developed using Django 2.0 with Python 3.6 and is capable of five coordinate transformations namely WGS84 to GDM2000 (forward and backward), WGS84 to MRSO (old) (forward only), MRSO (old) to Cassini (old) (forward and backward). Results obtained were compared with existing software such as GDTS v4.01 and ArcGIS 10.3, and analysis shows that CCA v1.1 has achieved satisfactory accuracy.

Topic: Web GIS Applications

1. INTRODUCTION

1.1 Background of Study

The spatial alignment of different layers is crucial in Geographic Information System (GIS) in order to be used together (Chang, 2014). Before the arrival of the Global Navigation Satellite System (GNSS), coordinate reference systems around the world have been using classical coordinate reference systems that uses a local datum and span a particular region only. Advancements in Global Positioning System (GPS) has led countries around the world to adopt a global geocentric coordinate reference system (JUPEM, 2009a), meaning a common datum is used for all regions. One such system is the World Geodetic System 1984 (WGS84). GIS users usually work with shapefiles, which is considered the standard and de facto data format. For shapefiles to align, these coordinates must be reprojected into a common spatial reference system. It is preferable to use the local geocentric datum (i.e. Geocentric Datum of Malaysia 2000 (GDM2000)) for region-specific analysis or mapping, since it fits the geoid of the region better (Navipedia, 2012).

1.2 Problem Statement

The non-alignment of spatial data because of difference in spatial reference system in GIS analysis cause significant problems. Other than that, not all coordinate systems are suitable for use in Malaysia. There is commercial GIS software such as Feature Manipulation Engine (FME) by Safe Software, Geographic Calculator and more commonly used, ArcGIS by ESRI, that are capable of converting coordinate systems into another but the accuracy of the converted coordinates is questionable. Moreover, they use generalized formulas in their conversion algorithm and are different from JUPEM standards, which Malaysia uses. Additionally, coordinate conversion applications such as the Geodetic Datum Transformation System (GDTS), Malaysian Coordinate Transformation (Amerudin, 2000) and most publicly accessible web applications such as the GDM2000 and State Cassini-Soldner Coordinates Conversion Calculator (Domino925, 2017) are only able to convert individual coordinates to another system, and are unable to accept shapefile input, which is the current standard for GIS vector data format. MaCGDI's MyGeoTranslator is a web application that allows GIS data format input in its coordinate transformation module, but it requires the user to possess a government email and register themselves with it in order to access the application itself. Therefore, there is a lack of a publicly accessible application that can convert the coordinate system of GIS data formats that complies with the JUPEM standards.

MapInfo and ArcGIS are two software that is commonly used to convert coordinates. Based on the study by Hooi (2011) who studied the difference between the level of accuracy in georeferencing using MapInfo and ArcGIS, found that both

showed error of around 0.1 m compared to the input point coordinate differences obtained from Digital Cadastre Database (DCDB). This error also affected the processes that was done afterwards such as area calculation. Therefore, it can be said that even commercial software is not completely error-free. Hooi (2011) also compared the gap magnitude between three layers of lot parcel. To do so, lot parcels from DCDB, MapInfo and ArcGIS were overlapped and the gap magnitude between them was measured. The result was that ArcGIS yielded a smaller gap compared to MapInfo. The smaller the gap magnitude between the raster dataset and the reference dataset yields better alignment result (Hooi, 2011). In relation to the focus of this research, spatial reference system affects the gap magnitude of datasets hence the alignment as well.

The study conducted by Hendrikse (2003) had an objective of finding a reliable set of datum transformation parameters by comparing old and new topographic maps. While the results for Netherlands were satisfactory, when the same method was applied to find the datum shift parameters for regions in Mozambique, it was found that the datum transformation parameters between the local system and GPS measurement defined in the WGS84 system weren't accurate enough. Therefore, the inaccuracies in coordinate conversion may not be due to difference in formula, it is because of the difference in predefined parameters. In the same paper, Hendrikse (2003) stated that parameters of local datums provided by the national survey are becoming more precise, and that they are also subject to updates once in a while. Hence, Hendrikse (2003) developed a custom VBA program to counter the absence of user-defined datum transformation using ArcObject. This supports the purpose of this research, which is to develop an application capable of converting coordinate systems using parameters defined by JUPEM, which is the government body responsible for monitoring changes in datum origins for Malaysia, rather than relying on current software's parameters that are most likely haven't been updated according to current datum conditions that has changed due to disasters such as big earthquakes.

2. LITERATURE REVIEW

2.1 Background to Malaysian Coordinate Systems

Work done in the 1880s was instrumental in initiating trigonometrical works in Malaya which led to trigonometrical survey across Malaya that laid the foundation for the existing control framework. However, these traditional coordinate systems were inconsistent and unconvincing in terms of reliability that it had to be revised in order to comply with the modern standards at the time (JUPEM, 2009a). This is why Malayan Revised Triangulation 1968 (MRT68) for Peninsular Malaysia was established. The work to establish Sabah's primary triangulation work began with a project known as the Borneo West Coast Triangulation (1930 – 1942) whereas for Sarawak and Brunei, around 1935 (JUPEM, 2009a). The task to readjust the primary triangulation of Borneo was undertook by the Directorate of Overseas Survey (DOS), who later commenced the East Coast Triangulation project that was the basis for the Borneo Triangulation 1968 (BT68). The origins of MRT68 and BT68 are Kertau, Pahang and Timbalai, Labuan respectively.

The Rectified Skew Orthomorphic (RSO) map projection is an oblique Mercator projection that is orthomorphic (conformal) and cylindrical, a suitable projection to use for countries near the Equator as it the scale is exactly true along the great circle of a sphere. Therefore, RSO is the best solution for minimising distortion while maintaining conformality for Malaysia (JUPEM, 2009b). In Malaysia, there are two types of RSO: MRSO (Malayan Rectified Skew Orthomorphic) and BRSO (Borneo Rectified Skew Orthomorphic), with origins Kertau, Pahang and Timbalai, Labuan respectively. These projections must be derived from MRT48/ MRT68 for MRSO and BT68 for BRSO before GDM2000 came into realisation. Due to the shape of Peninsular Malaysia and the number of states in it, Cassini-Soldner projection is used for large scale/ cadastral mapping purposes. There are nine state origins used for this projection.

Kadir, *et. al*. (2003) emphasizes the importance of providing a standardized or homogenous geodetic infrastructure as the basis for integration of spatial data for sustainable development decision making. The regional nature of MRT68 and BT68 are not aligned with global geocentric coordinate frames such as WGS84, so JUPEM made effort to fully utilize space-based technology by establishing a GPS network of 238 stations in Peninsular Malaysia which was called the Peninsular Malaysia Geodetic Scientific Network 1994 (PMGSN94); and then in East Malaysia, where 171 GPS stations were established to form the East Malaysia Geodetic Scientific Network 1997 (EMGSN97) (JUPEM, 2009a). The origins for East and West Malaysia are still Timbalai, Labuan and Kertau, Pahang, respectively. The origins were used as fixed points to do minimally constrained adjustments on their respective geodetic networks.

GDM2000 was developed by firstly by establishing the zero-order network of permanent GPS stations known as the Malaysia Active GPS System (MASS), then Malaysia Primary Geodetic Network 2000 (MPGN2000) through the strengthening of PMGSN94 and EMGSN97 via GPS observations at selected stations (JUPEM, 2009a). MASS network

was replaced by another network of active GNSS stations known as Malaysia Real-Time Kinematic GNSS Network or MyRTKnet, which was instrumental in detecting subtle changes in position after the major Sumatran earthquakes in 2004, 2005 and 2007 that led to the revision of GDM2000. For the establishment of GDM2000 as the national datum, JUPEM had 78 RTK reference stations all over the country. These stations continuously send data to the central facility in Kuala Lumpur where they are stored and corrected (JUPEM, 2009a). The revised version of GDM2000 became known as GDM2000 (2009), obtained after applying the multiple regression model on GDM2000.

2.2 Django Framework for Web Development

There are many programming languages that can be used to develop an application system and one of the most popular programming languages is Python. It is also considered a beginner's language and is easy to learn. Python can be used to build desktop and web applications, but the latter needs a web framework. The Django Framework is one of those frameworks. Django is a high-level Python Web framework that stimulates fast development and clean, pragmatic design. It has its own modules that may be imported to use in the website but user-created modules are allowed as well. It is free and open source, meaning the public may access its source code without paying. Hansen (2017) summarised the pro and cons of Django in Table 1.

Table 1: Advantages and Disadvantages of Django (Source: Hansen, 2017)

Django Framework for Web Development	
Advantages	**Disadvantages**
Fast – Designed to help designers make applications as fast as possible. Ideal for projects with tight deadlines.	Uses routing pattern to specify its URL – Access to web page is controlled by whether or not there is a URL to its view function specified in *urlpatterns.* Some URL require the use of regex.
Fully loaded – Includes many extras to help with user authentication, site maps, content administration, RSS feeds etc.	Too monolithic – Rigid and require extra work to undo some automatically set variables. User must follow Django rules.
Secure – Ensures developers don't commit mistakes related to security (e.g. SQL injection, cross-site request forgery, clickjacking, cross-site scripting).	Everything is based on Django Object-Relational Mappers (ORM) – Works slightly differently than Python, and takes practice to master
Scalable – Meets the heaviest traffic demand, good for busy websites.	Components get deployed together – Each component used will be deployed, the more modules used the bigger the project file.
Versatile – Efficiently manages content, scientific computing platforms, and even big organisations.	Require knowledge of full system to work

Django employs the model-view-controller client/server code reuse which involve the model layer, templates layer and views layer. The model layer deals with data definition and structure, focusing on fields, metadata, attributes and methods as well as model inheritance for further customisation. Basically, it is the database-builder layer that also provides methods for model querying, model instances and migrations. It also offers an assortment of other methods to assist in constructing and manipulating data. The templates layer is the syntax Django uses to generate information that is shown to the user. Templates in Django are reusable as it generates HTML for repetitive usage. It also supports Jinja2, a templating language, which is useful in changing the content of the web application without having to copy and paste the HTML codes for the interface. The view layer is the main control centre of the Django web application. The processing of the input and the how the output is shown or accessed by the user using simple web responses such as custom URL configurations, view functions, shortcut helper functions and decorators to support HTTP features. The forms layer handles parts needing user interaction such as uploading data and selections, using basic HTML, forms API or built-in fields and widgets. Forms can also be built using models from the model layer. Django also has its own Geographic Framework called GeoDjango (GIS functionality for Django, accessed through the *contrib.gis* package). It contains model fields for OGC geometries and raster data as well as extensions for queries and spatial data manipulation. However, it is difficult to build a generalised data model for shapefiles to use with GeoDjango as many shapefiles do not have standardised attribute fields. Because of that, importing data fields would be very difficult and hard to manageable. Hence, for the sake of completing this research within the time limit, GeoDjango was not used. Figure 1 summarises the Django Framework.

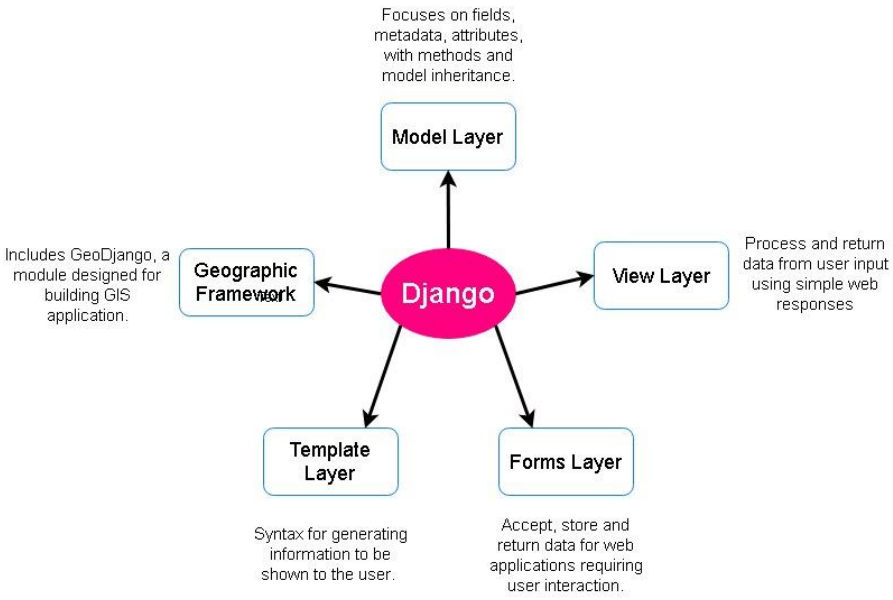

Figure 1: Django Framework Summary

3. METHODOLOGY

3.1 Research Approach

The methodology is comprised of five phases: preliminary study, planning, implementation, validation and conclusion, as illustrated in Figure 2.

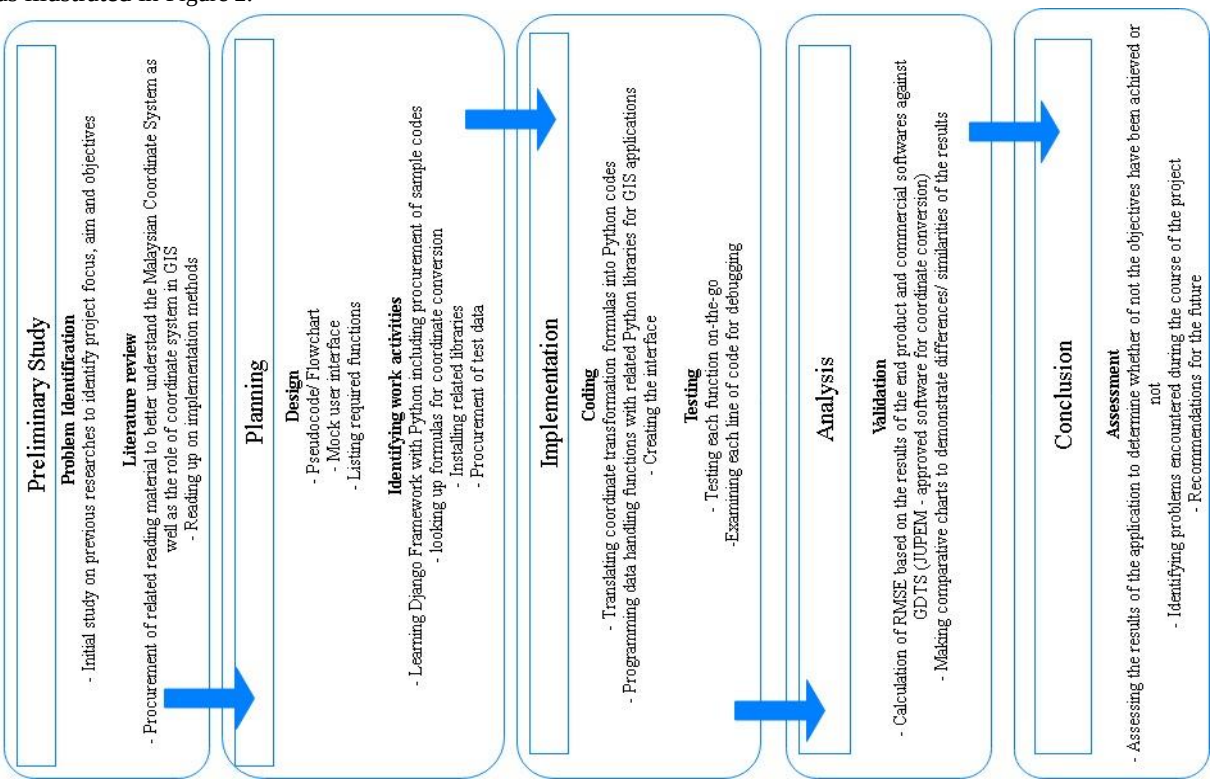

Figure 2: Research Approach

3.2 Development of Coordinate Conversion Application

One of the methods that can be used to plan a project is the System Development Life Cycle (SDLC). SDLC is a conceptual model that is used in system development. It gives the project the opportunity to combine two or more project methods for the best outcome. For this project, the methodology used is illustrated in Figure 3: System Development Life Cycle.

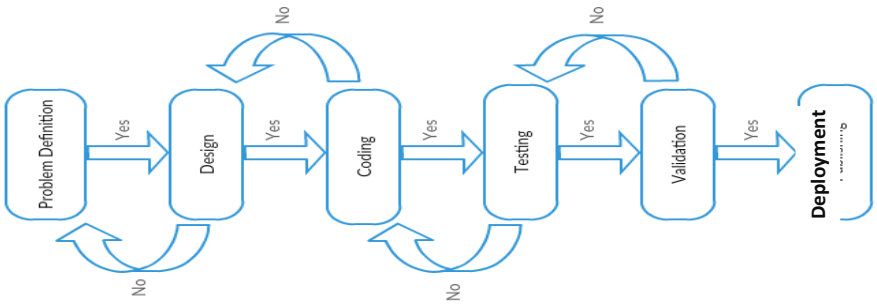

Figure 3: System Development Life Cycle

3.2.1 Problem Definition

Robertson (2000) highlighted three components that must be prioritised in defining a problem: Input, process and output. For this project, the input would be an ESRI Shapefile already in the WGS84 coordinate system, the processes would be the coordinate transformation algorithms and data handling functions, and the output would be an ESRI Shapefile that is either WGS84, GDM2000, RSO or Cassini coordinate systems that is saved in a different file from the input.

3.2.2 Design

The general process of the application would start from the user choosing whether they want to do a Datum Transformation or Map Projection. In Datum Transformation, they would have to choose whether to convert from WGS84 to GDM2000 and vice versa, whereas in Map Projection, they would have to choose either WGS84 to MRSO (old) and MRSO (old) to Cassini (old) (and vice-versa). Overall, the user would have five conversions to choose from. After that, the user would input all the files of a shapefile so it may be saved in a virtual folder for processing. Once saved in the virtual folder, the coordinate transformation takes over by firstly exporting it into GeoJSON format, extracting its coordinates and run it through transformation formulas and then rewriting it in GeoJSON. When all points have been converted, the shapefile writer function takes over and writes a shapefile based on the GeoJSON data. The new shapefile will be saved in a virtual folder before being turned into a .zip file. A download link for the .zip file will then be generated for the user to download the new shapefile. Overall, there will be three folders created for one conversion session: The input shapefile folder, the converted shapefile folder and the compressed ZIP file of the converted shapefile folder which is the one the user downloads. After one day, all folders in the virtual folder will be manually deleted by the administrator doing daily system clean up. Figure 4 shows the flowchart of the solution algorithm for the web application.

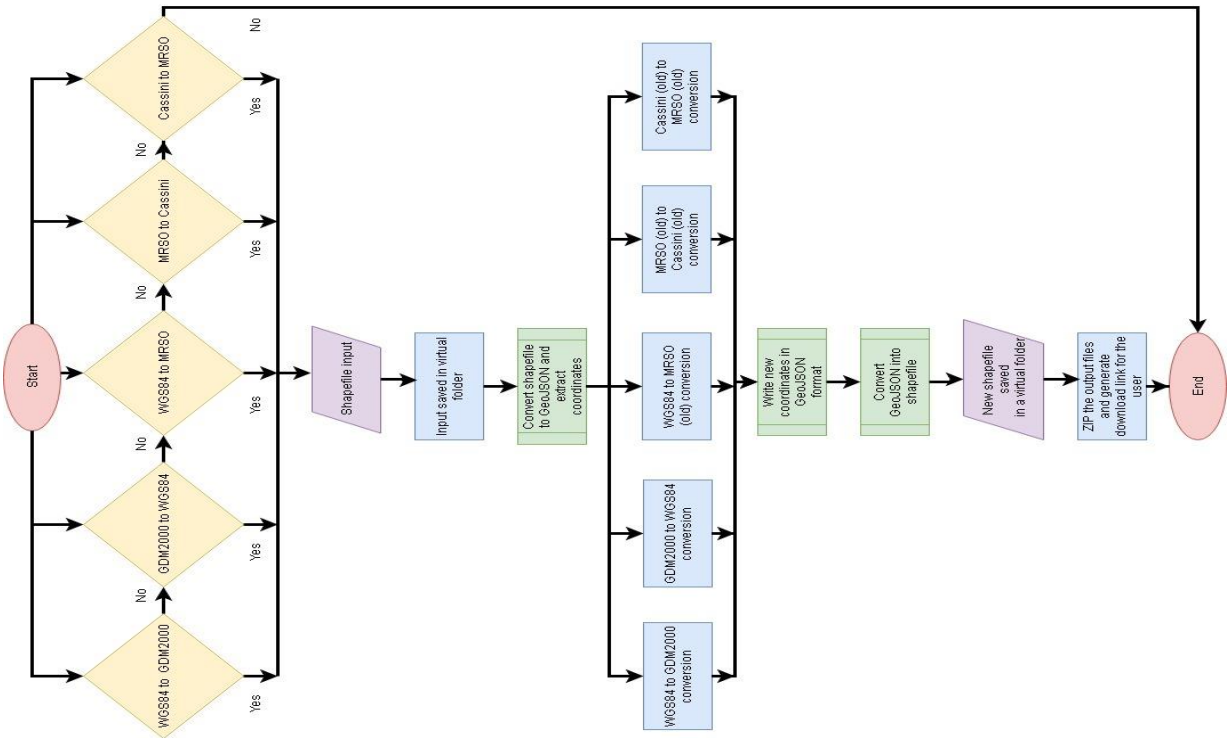

Figure 4: Application Flowchart

3.2.3 Coding

This part of the project follows the Django framework procedures strictly. At project creation, Django creates default modules that serves as framework to build a web application from at project initiation. After that, custom modules may be added for the specific purpose the application has. Figure 5 summarises the organization of files involved, their respective purposes and interaction.

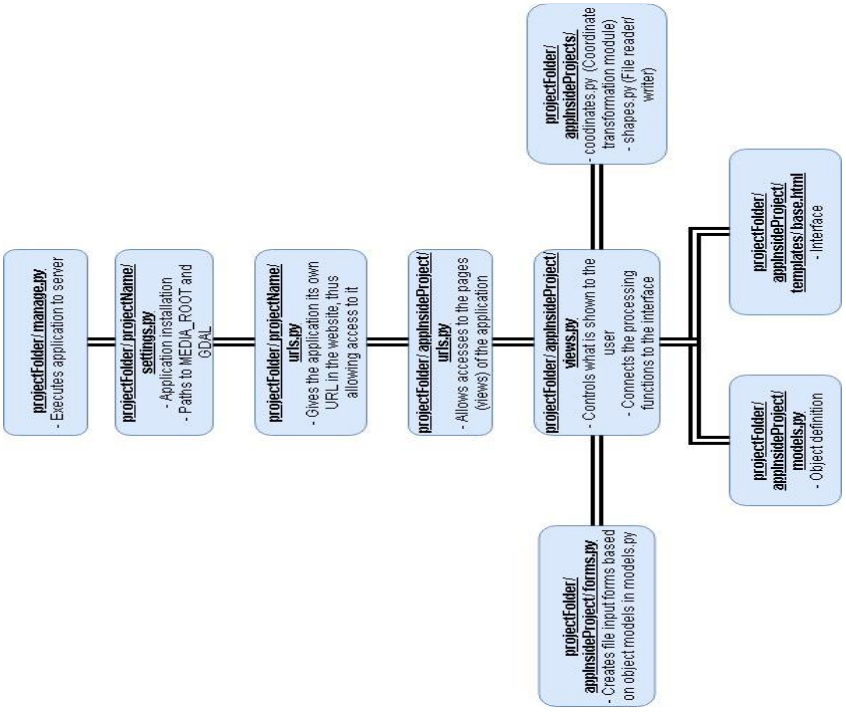

Figure 5: General File Structure and Interaction

3.2.4 Testing

The testing method chosen is the White-Box testing. White-Box testing involves detailed and thorough examination of the internal logic and structure of code. It is also called glass-testing or open-box testing and requires a knowledgeable tester as this method focuses on debugging. Test data may be designed according to the function being tested. This method is tedious and time-consuming but suitable for algorithm testing. Criteria considered in testing are coordinate transformation functions; data handling functions which involves accepting shapefile input, passing data to the coordinate transformation functions for conversion, writing new coordinates while preserving original attribute data and saving a new shapefile as output; and control functions which are the sequence from the data input, processing and output, selection of conversion, and process repetition or looping.

3.2.5 Validation

The Root Mean Square Error (RMSE) method is chosen for validation. This method is a measure of the error around the regression line, the same way the standard deviation is the measure of how varied the data is around the mean. The formula for calculating RMSE is:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x-x_i)^2+(y-y_i)^2}{n}} \qquad (1)$$

where,

x, y = Predicted Coordinates
x_i, y_i = Observed coordinates
n = Number of points

RMSE of the result from the coordinate conversion application will be compared with the RMSE of the result from a commercial software. Hooi (2011) mentions that for a DSMM approved level of accuracy, the RMSE value must be less than 0.2.

3.2.6 Deployment

The Coordinate Conversion Application will be deployed using a private server to the Faculty of Geoinformation and Real Estate website as it contains confidential parameters that must not be shared with public repositories like Github.

4. RESULTS AND ANALYSIS

4.1 Analysis on the accuracy of the coordinate conversion

A shapefile of the contour of Cameron Highlands with 32 features or polylines is used for the analysis of each of the coordinate conversions. The starting point or the first node of each feature is extracted for RMSE calculation. For the WGS84 to GDM2000 transformation, the shapefile in WGS84 was converted into GDM2000 using CCA v1.1 and a customised transformation with ArcGIS 10.3 since it does not have the transformation from WGS84 to GDM2000. The custom transformation was created with the Create Custom Geographic Transformation tool in ArcGIS 10.3 using coordinate transformation parameters obtained from JUPEM (2006). The text file containing the coordinates of the features of the shapefile obtained by exporting its attribute table using ArcGIS 10.3 was used as the input data for GDTS v4.01. Results obtained from CCA v1.1 and ArcGIS were compared with results from GDTS v4.01 and each other. Table 2 show the result.

Table 2: Results of the Coordinate Conversion between WGS84 and GDM2000

	WGS84 to GDM2000					
	GDTS -CCA		GDTS – ArcGIS (Custom)		CCA - ArcGIS (Custom)	
	Lat (“)	Lon (“)	Lat (“)	Lon(“)	Lat (“)	Lon(“)
Max. Error	2.42144×10^{-13}	2.46475×10^{-9}	2.3752×10^{-13}	2.46451×10^{-9}	3.13445×10^{-17}	9.92245×10^{-18}

Min. Error	$1.87888x10^{-17}$	$1.28135x10^{-12}$	$8.38056x10^{-17}$	$1.27577x10^{-12}$	2.0125×10^{-17}	4.66993×10^{-18}
RMSE	0.00003		0.00003		0.00000000556521	

Referring to Table 2, CCA v1.1 had achieved an RMSE of 0.00003" which was similar to ArcGIS 10.3 against GDTS v.4.01. The maximum and minimum latitude error for CCA v1.1 are $2.42144x10^{-13}$" and $1.87888x10^{-17}$" respectively whereas for ArcGIS 10.3 it was $2.3752x10^{-13}$" and $8.38056x10^{-17}$". For longitude, the maximum and minimum errors for CCA v1.1 were $2.46475x10^{-9}$" and $1.28135x10^{-12}$" respectively whereas for ArcGIS 10.3 it was $2.46451x10^{-9}$" and $1.27577x10^{-12}$" respectively. 0.00003" is within the acceptable margin of positional error. Both software achieved an RMSE of less than 0.2 which was the suggested error allowance by JUPEM (Hooi, 2011). Hence, this conversion can be considered a success. The RMSE between CCA v1.1 and ArcGIS 10.3 is 5.56521×10^{-9}", meaning there is less than 1 mm difference between CCA v1.11 and ArcGIS 10.3. Therefore, it can be said that CCA v1.1 and ArcGIS 10.3 has achieved the same results. This may be due to the custom parameters set for GDM2000 in ArcGIS 10.3 that are obtained from the exact same source which is the unpublished report from JUPEM (2006). Furthermore, this confirms that ArcGIS 10.3uses the same formula as GDTS v4.01 for datum transformation.

4.2 Analysis on the cartographic representation

The source shapefile from JUPEM in WGS84 and the same shapefile in GDM2000 converted with CCA v1.1 was overlaid to see if the shapes match. Based on visual comparison in ArcGIS 10.3, those shapefiles align nicely with minor differences, proof that the shapefiles are in different coordinate systems.

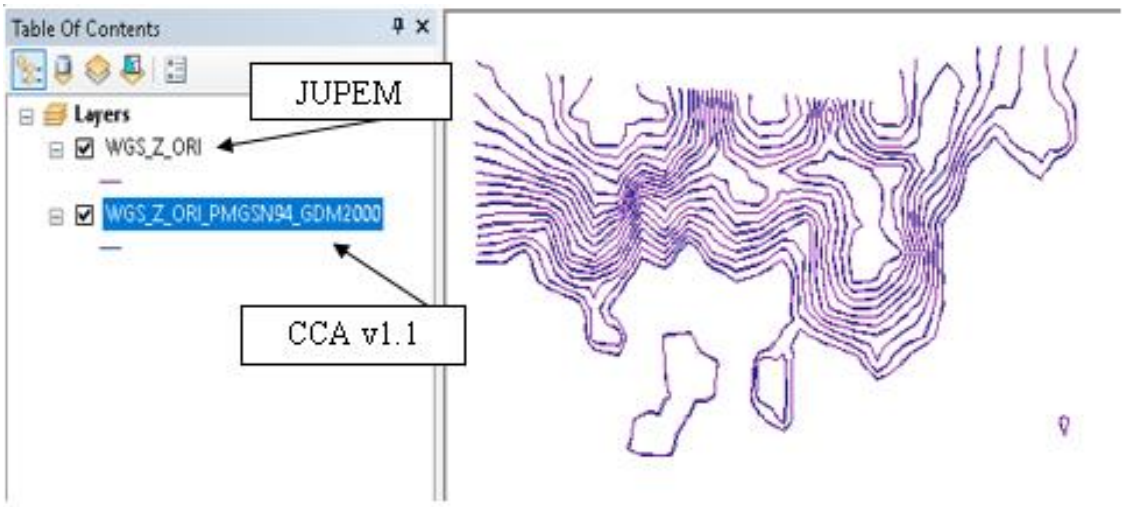

Figure 6: The input (WGS_Z_ORI) overlayed with the output from CCA v1.1 (WGS_Z_ORI_PMGSN94_GDM2000)

The attribute tables of the input and output were also checked to see if the attribute table was preserved after converting coordinate systems using CCA v1.1. There are slight differences in how numeric values in the output attribute table are preserved since CCA v1.1 converts the input to GeoJSON for coordinate processing, which removes truncating. As a result, the numeric values are all displayed with all of its decimal places. Figure 7 shows a part of the input and output attribute tables for comparison.

WGS_Z_ORI

	FID	Shape *	CONTOUR	ID_	ID	lat	lon	hae
▶	0	Polyline ZM	1485	28	0	4.609329	101.50546	1485
	1	Polyline ZM	1490	29	0	4.609318	101.505669	1490
	2	Polyline ZM	1515	30	0	4.608734	101.50629	1515

WGS_Z_ORI_PMGSN94_GDM2000

	FID	Shape	CONTOUR	ID_	ID	lat	lon	hae
▶	0	Polyline ZM	1485	28	0	4.60932942717	101.505459893	1485.0
	1	Polyline ZM	1490	29	0	4.60931804161	101.505668873	1490.0
	2	Polyline ZM	1515	30	0	4.60873392221	101.506290342	1515.0

Figure 7:Input attribute table (WGS_Z_ORI, left) and Output attribute table (WGS_Z_ORI_PMGSN94_GDM2000, right)

4.3 Limitations

Several limitations were identified in CCA v1.1. The exact extent of the file size acceptable by CCA v1.1 is untested, but the larger the shapefile, the longer it takes for CCA v1.1 to convert its coordinate system. The time taken to convert large shapefiles, for example, the original unclipped Cameron Highlands contour from JUPEM is about 48 MB in size and contains 2652 lines or features take up to one minute to process. The clipped shapefile was about 65 KB and took less than one second to process. Another flaw in CCA v1.1 is that it is not capable auto-detecting or write .prj files, therefore forces the user to confirm the coordinate system the shapefile is in before progressing with any coordinate conversion. The most visible weakness CCA v1.1 has is that it does not show progress messages or confirmation message boxes. Although this can be considered an extra feature, but it greatly improves user experience and helps the user keep track of the conversion progress.

5. CONCLUSION

Overall, the aim and objectives of this project was achieved. Although the project achieved satisfactory results, the following may be considered for future improvements:

1. More tests using different software other than ArcGIS 10.3 and GDTS v4.01, as to further validate the accuracy of CCA v1.1 such as FME, Global Mapper, etc.
2. Web-Server testing – due to time constraints, the limitations of the web server in terms of number of concurrent processes allowable at the time and the file size limit were untested. Furthermore, there could be a chance that the files of the stored shapefiles in the virtual folder for 2 or more concurrent users share the same name. Therefore, it would be better if future versions of the program could assign a unique ID to each instances instead of identifying them by folder name.
3. Improve user experience – progress messages (as the transformation is ongoing), confirmation messages etc.
4. More conversions – due to time constraints, focus had to be made on Peninsular Malaysia as the limited time did not allow validations to be made for conversions involving East Malaysia coordinate systems. Therefore, for future researches, it would be nice if coordinate systems from East Malaysia were included.

6. REFERENCES

Amerudin, S., 2000. Malaysian Coordinate Transformation.

Chang, K-T., 2014. Introduction to Geographic Information Systems. New York: McGraw-Hill.

Domino925, 2017. domino925 WebApps, Retrieved November 28, 2017, from https://dominoc925-pages.appspot.com/webapp/

Hansen, S., 2017. Advantages and Disadvantages of Django. Retrieved June 7, 2018, from https://hackernoon.com/advantages-and-disadvantages-of-django-499b1e20a2c5

Hendrikse, J. H. M., 2003. Use of the Spatial Reference Object Model to enhance Projection and Datum Transformation. Paper presented at the Esri User Conference, San Diego (USA), Retrieved November 30, 2017, from https://www. itc.nl/library/Papers_2003/non_peer_conf/hendrikse.pdf

Hooi, T. S., 2011. Comparison of Georeferencing Quality by Using Different Geographic Information Systems (GIS) Software. (Bachelor of Engineering (Geomatics) Thesis), Universiti Teknologi Malaysia.

JUPEM., 2006. Transformation and Projection Parameters: Unpublished Report.

JUPEM., 2009a. Technical Guide to the Coordinate Reference Systems. Jabatan Ukur dan Pemetaan Malaysia, Retrieved October 16, 2017, from https://www.jupem.gov.my/v1/wp-content/uploads/2016/07/Pek-1-2009.pdf.

JUPEM. ,2009b. Technical Guide to the Coordinate Conversion, Datum Transformation and Map Projection. Retrieved November 28, 2017, from https://www.jupem.gov.my/v1/wp-content/uploads/2016/07/PKPUP3-2009.pdf

Kadir, M., Ses, S., Omar, K., Desa, G., Omar, A. H., Taib, K., Teng, C. H., Mohamed, A., Chang, L.H., Saleh, R. and Nordin, S., 2003. Geocentric Datum GDM2000 for Malaysia: Implementation and Implications. Retrieved October 17, 2017, from http://eprints.utm.my/3527/1/gdm2000_DSMMKL.pdf

Navipedia, 2012. Regional Datums and Map Projections, Retrieved September 28, 2017, from http://www.navipedia.net/inde x.php/Regional_Datums_and_Map_Projections

Robertson, L. A., 2000. Simple Program Design: A Step-By-Step Approach (Third ed.). Cambridge: Course Technology.