A read, append, prune (RAP) formalism for spatio-temporal information

Geoff Hay[1], Peter Whigham[2] & G. Brent Hall[1]

[1]School of Surveying
University of Otago. Dunedin, New Zealand
Phone: +64 3 479-7585 Fax: +64 3 479-7586
Email: geoff.hay@otago.ac.nz
Email: brent.hall@otago.ac.nz

[2]Department of Information Science
University of Otago. Dunedin, New Zealand
Phone: +64 3 479-7391 Fax: +64 3 479-8311
Email: peter.whigham@otago.ac.nz

Presented at SIRC NZ 2013 – GIS and Remote Sensing Research Conference
University of Otago, Dunedin, New Zealand
August 29th – 30th 2013

***Keywords and phrases*:** spatio-temporal, database, cloud

1.0 INTRODUCTION

In the land administration domain, there is a need for software applications that are capable of faithfully recording the history and state of national land resources, which are easily adaptable to meet varying needs that occur within and across jurisdictions, and which are easily evolvable to meet growing sophistication in the requirements and uses of such spatio-temporal information (see, for example, Van der Molen, 2002). The faithful recording of history includes the chronology and causes of changes to spatial and thematic value data, the full retention of data in the face of continuous updating, and the faithful representation of the many and widely varying and evolving concepts and practices underlying land, its use, and its associations with people, organisations and government.

Critical to long-lived, authentic, and evidential record keeping software applications is their ability to also accommodate change over time in the concepts and practices they implement while at the same time maintaining support for existing information and its representation that accumulates over time. These needs are not well-supported by conventional geographic information systems (GIS) and this has been attributed the inadequacy of the relational model in representing the temporal dimensions of time-varying information and evolving schema, and structural limitations in the representation of domain concepts and record keeping practice in general (Raper, 2000, Gounaris and Theodoulidis, 2003, Ekweozor and Theodoulidis 2004).

In response to these needs, this paper proposes a read, append, prune (RAP) information theory that addresses domains where the concepts and practices of information are non-stationary with respect to both space and time. The model mirrors the essential concepts and practice associated with formal record keeping and as such is not limited to the land administration domain. Conceptually, the variation (or heterogeneity) in the concepts and practices that occurs across contexts (and space), is similar to the variation that occurs across time since the impact of these heterogeneities on implementation is essentially the same (Galante et al, 2005, Roddick et al, 2001). An implementation that allows varying conceptualisations (e.g. schema, processing modules) to coexist can be more easily evolved since there is less need to migrate existing representation as new concepts and practices are implemented. Such an implementation should thus be capable of supporting the spatially varying conceptualisations. This alludes to an implementation that supports an emergent and evolving representation through a non-delete amendment-based temporal model of information and implementation.

2.0 RELATED WORK

Append-only and non-delete strategies for data have long been considered essential concepts in temporal record keeping and faithful history recording (see, for example, Hunter, 1988; Roddick *et al*., 2000). Associated with this is the notion of limitless data storage capacity and the conclusion that removing the concept of delete makes database operations generally simpler (Copeland, 1982). Entity, attribute, value (EAV) models of data have been employed successfully in medical applications to address the problems of record heterogeneity that results in large numbers of nulls in records (Nadkarni *et al*, 1999). The ideas of flexible schema and emergent schema have also been considered in response to the problems of schema evolution in application areas that require rapid response to changes in the problem space, and areas where the schema is difficult to know in advance of implementation or for which there are competing hypotheses (Roddick *et al*, 2001, 2007).

Some of these ideas date from the early days of computing. However, they were generally ignored due to the then predominant focus on efficient use of limited memory and processing capacity that resulted in systems that do not allow redundant data, which require data to be deleted (or overwritten), and which rely on essentially static and predefined models that are difficult to evolve and reuse. Interestingly, the technological basis of the 'cloud' supports the reinvigoration of these ideas and a result has been a move away from conventional relational technologies and methods in favour of more evolvable and less ridged information models (see, for example, Cattell, 2011; Strauch, 2011; Taylor, 2009). The `cloud' essentially assumes spatio-temporal non-stationarity in both its applications and data models. For example, similar in concept to EAV models, Facebook and Google utilise multidimensional maps of key/value pairs which do not have a predefined list of columns and, as such, are able to add dynamically new attributes to records and can thus support varied and emerging conceptualisations (Chang et al., 2006, Sarkissian, 2009, Lakshman and Malik, 2010). Through its promise of unlimited storage and processing capability, the concept of 'the cloud' provides a platform in which concepts of time and record keeping associated with an append-only information model can be reconsidered (Furht, 2010).

3.0 READ, APPEND, PRUNE (RAP)

An append-only theory assumes that all data are retained. Implicitly, it assumes that all schemas associated with data are also retained, and that the schemas of data can evolve over time without intervention. Schema evolution that only involves the addition of elements does not impose the need to migrate data which is required when schema elements are to be removed. Logically, if schema can evolve seamlessly, then so must the applications and 'stacks' that rely on them. An append-only theory therefore should include the notion that not only are data appended, but their schema, and the implemented applications that provide processing and query, and human interaction, are also append-only and can accumulate. Essentially, once an item has been appended and has impacted the stored information, its impact can never be removed. This does not, however, preclude the notion that data or implementation can be removed from use, i.e. retired or made inaccessible. This, and all other change including schema change can be achieved with the concept of amendment.

Amendment is logically required in an append-only theory since data values and implementation artefacts cannot be deleted from the reality they have impacted. Amendment relates to the appending of new information or understanding, and to an intention with regard to an effect on existing information and understanding. An amendment conceptually modifies an existing value, attribute or entity or creates new attributes and entities. Various specialised amendment intentions can be defined and these include retirement, expiration, invalidation, and accumulation. Amendment is logically a bi-temporal concept since amendments are time-ordered both in the order in which they are recorded and the order associated with the time at which their intention becomes valid and affects a domain. Thus, amendments can be logically arranged into chains of facts representing, for example, time-varying attributes and their values, or the logical evolution of implementation artefacts that comprise a software deployment.

In addition, an amendment can, as part of its implementation, impose new kinds of understanding as well as create new values. This notion underpins the concept of variable and evolvable schema where the appending of a new value may in fact add a new attribute to an entity instance. Thus, individual entities may have unique and time-varying schema. For spatial data, this means that each spatial object instance may have an independent and unique set of attributes. The evolving understanding can be imposed through the appending of a new version of an implementation artefact such as a user interface application or service that contributes a new process or understanding and which potentially imposes a different understanding for new records.

The concepts of read and append are interrelated in the sense that the particular understandings imposed by an implementation through its set of valid append operations may also be used to understand the results of read operations. This alludes to a dual nature to service implementation and is an extension of the application-based transaction and query paradigm associated with cloud-like application models described above, and the idea of cloud services and software-as-a-service (SaaS) (Furht, 2010).

The conventional concept of pruning relates to the removal of nodes (whether they are attributes, value data, semantic descriptions, or implementation artefacts) that are redundant or irrelevant to a context. Pruning does not relate to the management of the large volumes of data that might be expected to accumulate under an append-only theory. Here, pruning simply acknowledges the legal requirements of privacy and authentic archival or destruction of records associated with retention and privacy policy implementation. Pruning is a function that processes intentions defined by amendments. Generally, pruning makes inaccessible certain facts (or implementation) either within a specific context (which might be a temporal context) or for all contexts and hence does not violate the append-only theory.

4.0 CONCLUSION

The RAP theory alludes to an architectural model that separates time-referenced amendments, semantic descriptions, and application or implementation modules such as services and service applications, allowing these to vary across space and over time. The theory is eminently suitable for underpinning a cloud-like implementation of formal record keeping that supports spatially, contextually varying and evolving data, understanding, and implementation.

The theory has impacts for the way in which spatial information and spatial change is managed and recorded and potentially for the implementation of GIS which are conventionally reliant on relational databases for the implementation of thematic attributes. A RAP-based data store allows the domain specific or contextual information to be recorded separately from the spatial representation thus allowing the form of individual records to vary across space and across spatial records, and for the particular spatial representation to vary across thematic records. Hence, it also allows the particular spatial representation to vary both in sophistication and accuracy across records and over time for individual records, thus potentially modelling more closely actual and evolving reality. It also allows temporal relations, for example, parent/child associations, between spatial objects to be explicitly recorded and for temporal relationships to be considered along with spatial relations, for example, spatial and temporal overlap where objects can be allowed to overlap in space only if they do not overlap in time. Ultimately it reduces the coupling between spatial representation, the implementation of which is reasonably stable, and the domain specific models that are highly variable across space and context. The RAP model for evolvable spatio-temporal record keeping forms the basis of an approach to the development of a widely applicable and long-lived GIS for land administration described by Hay (2013).

REFERENCES

Cattell, R. (2011) Scalable SQL and NoSQL data stores. *SIGMOD RECORD, ACM, 39*, pp 12-27

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. E. (2006) Bigtable: A Distributed Storage System for Structured Data. *In Seventh Symposium on Operating System Design and Implementation (OSDI'06)*, Seatle, USA.

Copeland, G. (1982) What if Mass Storage Were Free? *Computer*, 15(7), pp 27-35.

Ekweozor, U. and Theodoulidis, B. (2004) Review of retention management software systems. *Records Management Journal*, 14(2), pp 65-77.

Furht, B. & Escalante, A. *(ed.)* (2010) Cloud Computing Fundamentals. *Handbook of Cloud Computing, Springer US*, pp 3-19

Galante, R. d. M., dos Santos, C. S., Edelweiss, N., and Moreira, A. F. (2005) Temporal and versioning model for schema evolution in object-oriented databases. *Data & Knowledge Engineering*, 53(2), pp 99-128.

Gounaris, A. and Theodoulidis, B. (2003) Data Base Management Systems (DBMSs): Meeting the requirements of the EU data protection legislation. *International Journal of Information Management*, 23(3), pp 185-199.

Hay, G., (2013) Architecture for instrument-centred land administration applications, *PhD Thesis*, due for submission December 2013.

Hunter, G, J. (1988) Non-current data and geographical information systems. A case for data retention. *International Journal of Geographical Information Systems*, 2(3), pp 281-286.

Lakshman, A. and Malik, P. (2010) Cassandra: a decentralized structured storage system. *SIGOPS Oper. Syst. Rev.*, 44, pp 35-40.
Nadkarni, P. M.; Marenco, L.; Chen, R.; Skoufos, E.; Shepherd, G. & Miller, P. (1999) Organization of Heterogeneous Scientific Data Using the EAV/CR Representation. *Journal of the American Medical Informatics Association*, 6, pp 478-493.

Raper, J. (2000). Multidimensional Geographic Information Science. London: Taylor & Francis.

Roddick, J. F.; Ceglar, A.; de Vries, D. & La-Ongsri, S. (2007) Postponing schema definition: Low instance-to-entity ratio (LItER) modelling. *Active Conceptual Modeling of Learning*, LNCS 4512, pp 206-216.

Roddick, J. F.; Grandi, F.; Mandreoli, F. & Scalas, M. R. (2001) Beyond Schema Versioning: A Flexible Model for Spatio-Temporal Schema Selection. *GeoInformatica*, 5, pp 33-50.

Roddick, J. F. (2009) Schema Vacuuming in Temporal Databases. *IEEE Transactions on Knowledge and Data Engineering*, 21(5), pp 744-747.

Roddick, J. F., Al-Jadir, L., Bertossi, L., Dumas, M., Estrella, F., Gregersen, H., Hornsby, K., Lufter, J., Mandreoli, F., Mannisto, T., Mayol, E., and Wedemeijer, L. (2000) Evolution and change in data management - Issues and directions. *SIGMOD RECORD*, 29(1), pp 21-25.

Sarkissian, A. (2009) WTF is a SuperColumn? An Intro to the Cassandra Data Model. Accessed 3 August 2011 <http://arin.me/blog/wtf-is-a-supercolumn-cassandra-data-model>

Strauch, C. (2011) NoSQL Databases. Accessed 26 July 2011< http://www.christof-strauch.de/nosqldbs.pdf>

Taylor, B. (2009) How FriendFeed uses MySQL to store schema-less data. Accessed 3 August 2011 <http://bret.appspot.com/entry/how-friendfeed-uses-mysql>

van der Molen, P. (2002). The dynamic aspect of land administration: an often-forgotten component in system design. *Computers, Environment and Urban Systems*, 26, pp 361–381