



Research MiniConf

January 15th, 2007

Speakers:

- Andrae Muys
- Alison Young
- Ole Neilsen
- Darren Skidmore
- Clinton Roy
- Pia Waugh

Panel:

- Chris Samuel
- Gernot Heiser

Organiser:

- Elspeth Thorne

Building an Enterprise-Scale Database for RDF Data

Andrae Muys¹

Abstract

The Resource Description Framework (RDF) is a suite of technology standards produced by the W3C that provides a possible solution to the problems posed by semi-structured data. Supporting these standards, the Mulgara Project (<http://www.mulgara.org>) is currently the leading Open-Source implementation. We first provide a formal definition of semi-structured data in terms of vocabulary and semantics. We then examine a rationale for RDF as a model for semi-structured data from the perspective of Relational Normalisation. We then discuss the Mulgara Project, its existing design and functionality. The paper then introduces the design of a new store layer based on a combination of functional programming techniques, specifically Okasaki-style persistent datastructures, and traditional techniques of efficient external-memory datastructures. This combination is shown to provide substantially simplified implementations of advanced enterprise-functionality including: Lock-free multiversion concurrency; Live backup and restore; Federation; and Replication.

Introduction

Irregularity is the bane of traditional database schema design. Regularity requires coordination and consequently centralised standardisation of design. Regularity also often requires an abstracted world-view, where exceptions to the rule are considered inconsequential and are ignored for the benefits a well defined schema can bring. The success of the Relational Model is a testament to the degree to which these two traits dominate many of the data management problems we solve. The Resource Description Framework (RDF)[1] is a series of standards managed by the W3C that are intended to provide a framework for handling the failure of either of these constraints. RDF was first proposed in 1997 as an evolution of PICS [2] to provide a metadata standard for web resources. Since its release as a W3C recommendation in 1999[3] RDF has been widely adopted for resource description tasks it was originally designed for, including content rating[3], bibliographic[4], social-networking[5], and syndication[6]. However it has also been widely used in a broad spectrum of metadata and semi-structured data domains. RDF is now being used to provide web-services[7], life-sciences research[8], lexicons[9], document management[10], and data-integration[11].

With the RDF's widespread deployment on the internet, and its continuing adoption by traditional metadata users, there is an increasing demand for the ability to store and query large quantities of RDF data. The largest RDF datastores currently support datasets of the order of 1 billion statements - roughly equivalent to a 1,000 tables x 10 columns x 100,000 rows in a relational database[12]. This is sufficient for existing uses of RDF, but there are many problems in the life-sciences, archival, and internet-search domains that require substantially larger datasets. The problem is that like other domains such as GIS and full-text-search, RDF data has a very different 'shape' to traditional relational datasets. This imposes significant limitations on the ability of RDBMS to support it efficiently. Therefore as with dedicated spacial-data servers and full-text indexing before it, it is necessary for RDF to consider alternative datastore implementations designed to exploit its specific properties. This approach has allowed the Mulgara RDF Datastore[13] to scale to

¹ Andrae Muys <andrae@netymon.com> is a Principal Consultant at Netymon Pty Ltd and is completing a Research Masters with The University of Queensland in the application of persistent datastructures to external-memory.

250,000,000 statements thus far[14], and should eventually allow us to scale up to 10-100 billion - an enterprise scale RDF datastore.

The rest of this paper is divided into 4 parts. The first is an introduction to, and rationale for, RDF; the second an introduction to the existing Mulgara implementation; the third details the design of a new store-layer that is expected to provide a 1-2 order-of-magnitude improvement on the current implementation; and the paper concludes with a discussion of the additional enterprise features[15] that become feasible with the new design including live backup and restore, federation, and live replication.

Defining “Semi-structured Data”

Since Codd's seminal 1970 paper introducing the relational database[16], the concept of a database has become increasingly conflated with SQL and RDBMS. The rigor provided by relational theory assists developers by allowing them to reason about such issues as referential consistency, constraints, transaction serialisation guarantees, and the correctness of query optimisations.

However while the solution provided by the relational model is both impressive and durable, it does not solve every problem software practitioners face. With the advent of the web and internet data sources, these problems are becoming increasingly prevalent. The fundamental cause of the problem is a conflict between the a priori-regularity demanded by the relational model, and the messy, changing nature of the world we wish to model.

One common problem that eludes the relational model is that of unstructured data, and there has been extensive research in recent years on full-text search and query support for natural language text, image, and multimedia data that has almost no structure to guide search. There is also an increasing amount of data that is insufficiently structured to support traditional database techniques, but does contain a sufficiently regular structure that we wish to exploit in the formulation and execution of queries[17].

Two early projects attempting to address the issue of semi-structured data include the UnQL query language[18] from U.Pennsylvania, and the Lore lightweight object repository[19] developed at Stanford. Both of these are built on earlier work on heterogeneous database integration, the Object Exchange Model (OEM)[20]. One important insight of OEM is the critical importance of source defined vocabularies to semi-structured data - a concept that is the crucial distinction between unstructured and semi-structured data.

One thing that has remained a difficulty is the definition of the term semi-structured. The most thorough description of semi-structured known to the author is the one used by the Lore project[21]. Its weakness is its focus on only differentiating between structured and semi-structured data.

Our Definition: Semi-structured data is...
Data that:

- a) has a well defined vocabulary and
- b) has a well defined semantics and
- c) either

makes the open-world assumption, or assumes names are not unique, or makes no assumptions about relationships cardinalities.

surface syntax of the data for many of the properties it identifies, which makes it difficult to differentiate semi-structured data from both unstructured and fully-structured data.

It is useful to define semi-structured data with respect to both ends of the fully-structured/unstructured spectrum. Taking bibliographical information as an example, we would like to draw a distinction between the loose grammar of natural language, and the strict grammar of a BibTeX entry. We also wish to distinguish between the rigid semantic constraints of an XML serialisation of a relational table, and the loose semantics of that same BibTeX entry - even if it is also serialised as XML.

This paper uses the concepts of 'source defined vocabulary' first identified by Papakonstantinou[20] and 'source defined semantics' to distinguish structured from unstructured data. Using this definition structured data is data its source defines a vocabulary used to specify it, and also defines a semantics for that vocabulary that can be used to give the data meaning. Note that these definitions (vocabulary and semantics) are not required to be consistent with any entity or standard external to the source. In semi-structured data the vocabulary is often implicitly defined within the data itself, however without a semantics we believe this data should be considered unstructured. Requiring a semantics allows us to distinguish between pure markup languages (XML/HTML) and semi-structured data serialised using them (RSS[6]/FOAF[5]).

Having distinguished between structured and unstructured, it becomes necessary to distinguish between 'fully-structured' and semi-structured data. Both share the property that they have a defined vocabulary and semantics. That is to say, given any statement it should be possible to identify if the statement is valid, analogous to identifying "The sky laughed phone without running" is a valid english sentence; and if that statement is true or false in a given context, except in some very esoteric contexts the previous sentence would clearly be false. However it is in the semantics that we find the key to distinguishing the two.

Traditionally structured data makes three simplifying assumptions, these assumptions are foundational to both relational and traditional proof-theoretic data models[22].

- The Closed World Assumption: which allows the full use of first-order-logic including universal quantification (forall) and failure as negation.
- The Unique Name Assumption: which allows the specification of primary and candidate keys and uniqueness constraints in the relational model.
- The Strict Cardinality of Associations: which allows the specification predicate arities in logic, and of NOT NULL attributes, and referential consistency constraints in relational datamodels.

It is in violating these assumptions that semi-structured data gains its flexibility.

- The Open World Assumption: metadata is regularly incomplete. Just because a citation does not list a publication date, does not mean the document was never published. You can only rarely make universal inferences from metadata. Without a closed-world, negation gets you in trouble ie. "Return all the books NOT written by me".
- The Unique Name Assumption: metadata from multiple sources routinely use different names for the same entity. Worse, many of the entities we wish to describe have no 'name'. Consider the impossibility of identifying a person uniquely and how this affects the management of data from heterogeneous sources.
- The absence of Strict Cardinality: Strict Cardinality was only ever a useful approximation of the real world. The email address relation with a reference to its 'owner' breaks down when you introduce mailing lists, remailers, and shared-mailboxes. In the real-world there are no NOT NULL attributes.

We therefore come to the following formal definition of semi-structured data. Semi-structured data is data described with a well defined vocabulary with well defined semantics, that has one or more of the following characteristics: open-world semantics, absence of unique-names, absence of strict cardinality.

A Rationale for RDF

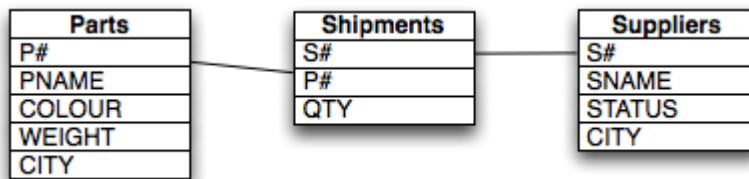
The Semantic Web is a suite of W3C standards intended to provide the ability to define, describe, and manipulate any semi-structured data. As a result it provides the means for users to define vocabularies (both implicit and explicit); semantics; and makes none of three assumptions of traditional structured

databases. It is instructive to consider RDF in terms of the response of a relational model to increasing complexity.

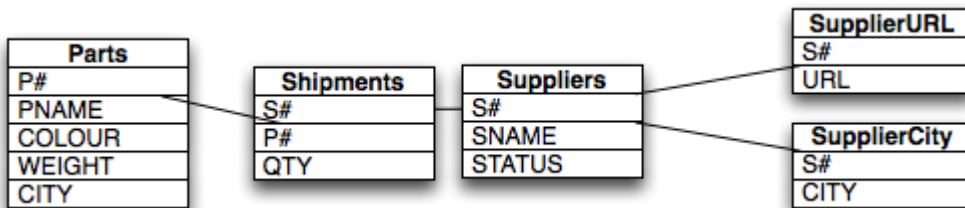
As with all structured data-models, the relational model works with an abstraction of reality. The exceptions to the rules are either pruned or appended as unstructured 'miscellaneous' attributes. In the vast majority of data management domains, this abstraction is sufficient to meet the requirements of the applications built upon it. It is when the exceptions are non-trivial, or the requirements demand attention to them that the abstraction breaks down. It is when the exceptions are of interest and cannot be ignored Semi-structured data-management techniques become important.

To illustrate this problem we progressively renormalise the classic "Suppliers and Parts Database"[22] as we reintroduce the complicating exceptions ignored by the original schema.

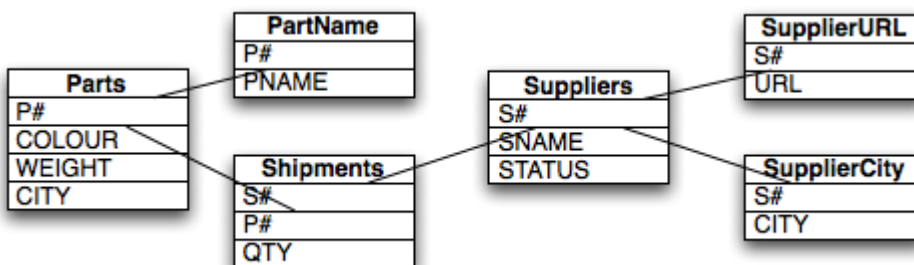
The Parts and Suppliers database: Only Three relations:



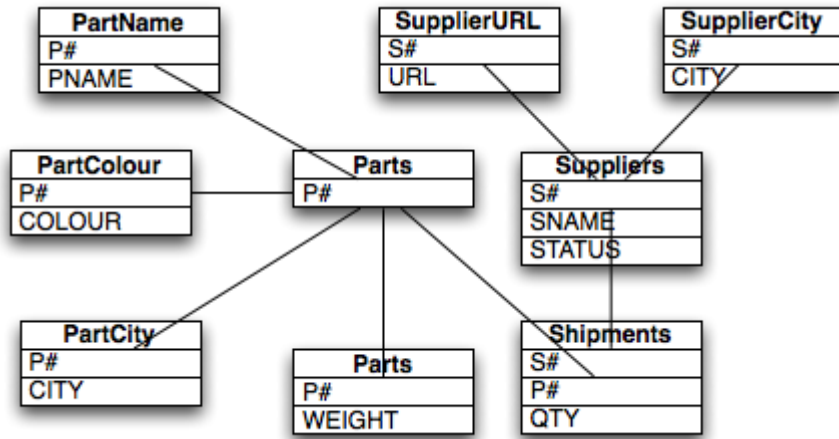
The world changes, and therefore so must the schema: **Suppliers go on-line**



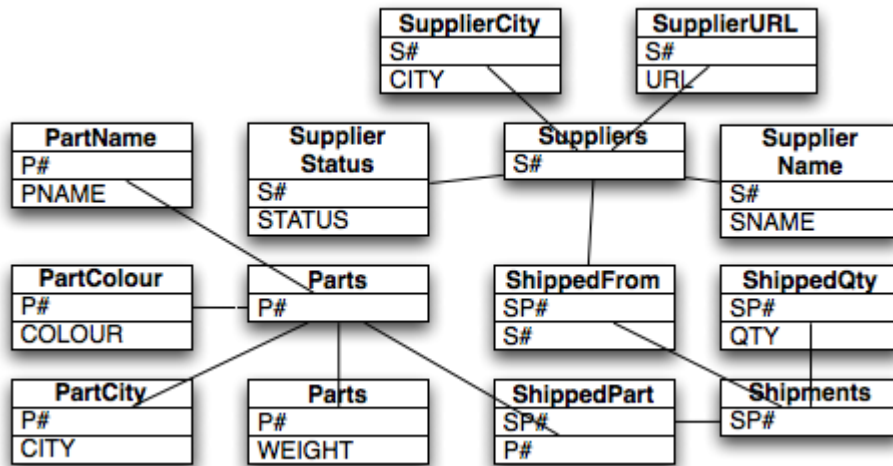
Marketing declines to use part names invented by Engineering: **PName Cardinality Broken**



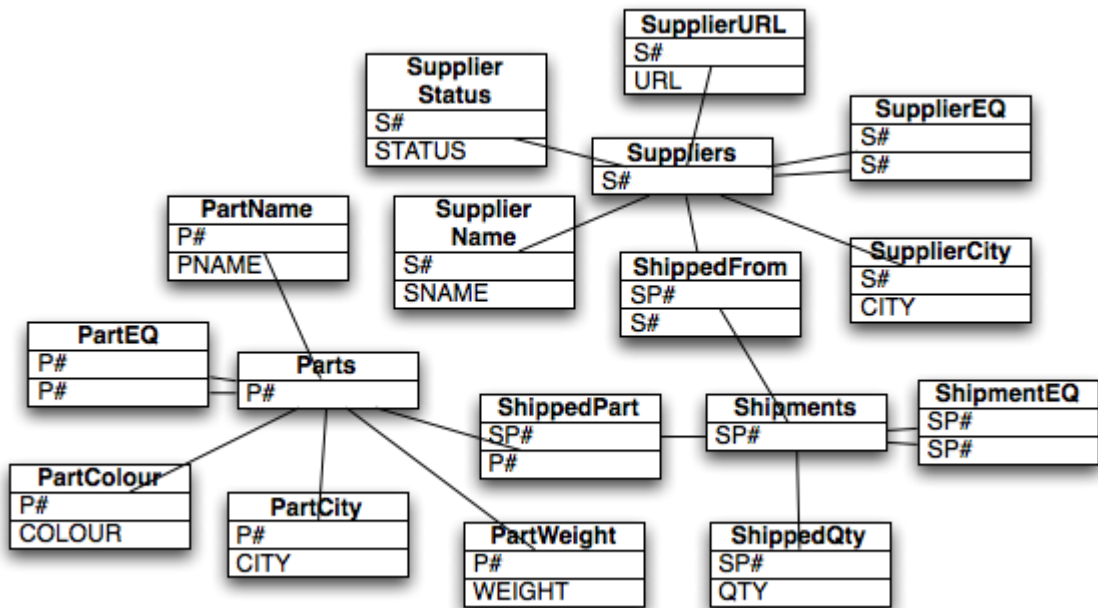
The company merges with another that doesn't track colours and weights become optional. While Parts become stored in multiple cities: **No Valid Cardinality constraints on Parts.**



We can continue this process until we have eliminate all strict cardinalities



Now we drop the unique name assumption.



So now you can express the fact that the shipment of 500 bolts that arrived without an origin address is the same shipment as the 500 'units' from 'Smith's Supplies' that you were told about last week.

We won't revoke the closed-world assumption as this does not impact the schema directly. If you can't assume a closed-world then denormalisation is the least of your problems. Moving to an open-world assumption invalidates aspects of the query language you are using to access your data!

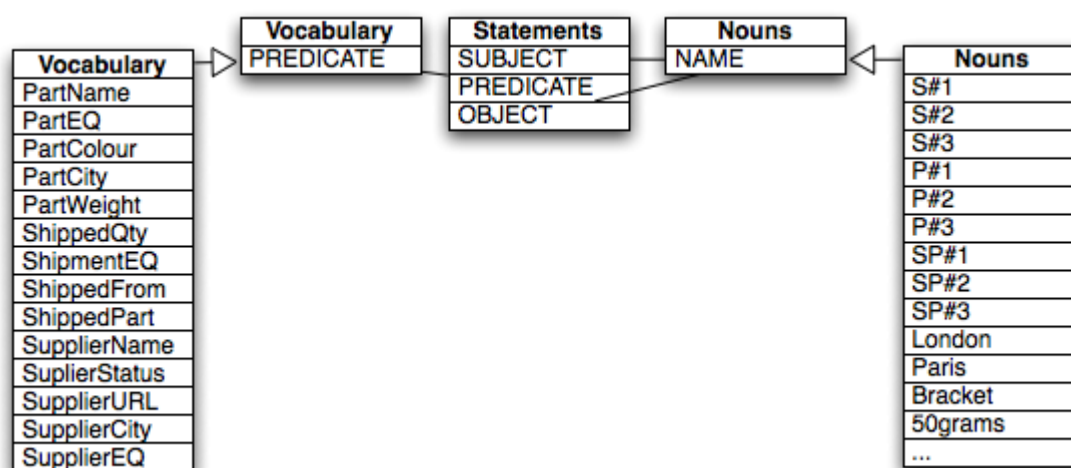
Real-world intervention leads us to a fully-normalised schema consisting of tables of 1 column + foreign keys. At this point we have a representation of the pure prepositional logic that underlies relational theory. Around the time your 'ideal' schema starts to look like this, RDF becomes interesting.

However, besides the performance, and conceptual issues there are the data-lifecycle aspects worth considering. At any non-trivial scale the migration of data from the original schema to the newer schemas is extremely problematic — even more troublesome is the migration of applications. This arises from the conflation of three orthogonal concepts within the relational model.

1. The data — the contents of the database - “there is a part with part-number 26, name 'Bracket' and weight 50grams”
2. The vocabulary used to describe the data - in the relational model, table/attribute names
3. The semantics of the data - the relational predicates and their arities/foreign-key relationships

The data is fixed in time — at the point at which we load it, and the full extent of that data is defined by the contents of the data we load. We need to load data using a standardised vocabulary to describe it, but extending the vocabulary as requirements evolve does not affect previously loaded data, neither does it necessarily invalidate applications written against a prior version. Finally while some of the relationships associated with structure are implied by the data itself, the arities and cardinalities are often independent, external to the data — and being external are subject to change as requirements shift with time.

The lifecycle problems are a result of the relational model starting with an apriori schema, that defines a vocabulary, with which you load data. This leaves the vocabulary rigid and inflexible. For structured data this inflexibility can cause problems, but is manageable and widely viewed as an acceptable tradeoff. Semi-structured data cannot afford the luxury of a rigid vocabulary, and an apriori schema as its vocabulary and semantics are source defined, independent of the database. As a result if we are going to define a semi-structured suppliers and parts database, we are going to have to represent this inversion explicitly, which introduces a structure very similar to the RDF data-model.



By using a single table we eliminate the structural constraints on our data. There exists no data that cannot be loaded into this table given appropriate values for Predicate and Noun. In the example above the valid predicates are defined by the Vocabulary table as shown. In defining this set, we are defining a vocabulary independent of the structure of the data itself. In doing this we define a 'lexicon', which we can use to describe our data. To return to our natural language example, the STATEMENTS table provides us with a

sufficient syntax grammar to combine words into sentences (The ipsum zwoks lorem). The VOCABULARY table defines a lexicon that introduces the concept of an 'english sentence' (The ipsum has colour lorem). To recover an analogous concept to 'a meaningful english sentence' we need to introduce an ONTOLOGY, which moves beyond the scope of this paper. It is worth reiterating that it is this careful deconflation and layering of concepts, syntax/lexicon/ontology, that provides the flexibility we require to represent semi-structured data.

The RDF Data Model

The most distinctive feature of the fully-normalised suppliers and parts schema is the reduction of the n-ary relations to binary predicates. Indeed we can map any 1NF relational schema to such a schema of binary predicates by introducing skolem constants to reify each tuple, and introducing a binary predicate for each attribute of the relation.

A schema that consists solely of P(S,O) binary predicates, can be readily represented as a directed labeled graph, with each subject/object 'noun' being represented by a vertex, and the predicates as labeled edges from subject to object. These 'triples' are the fundamental building blocks of the "Resource Description Framework" or RDF. It is the flexibility in representing arbitrary structure without apriori schemas that provides RDF with its power. Each edge in the graph is a single fact, a single statement; equivalent to the relationship between a single cell in a relational table and its row's primary key. To this graph RDF applies a field of logic called "Description Logics", that provide a rigorous mathematical foundation permitting valid and consistent reasoning — analogous to Relational Theory in Relational DBMS'.

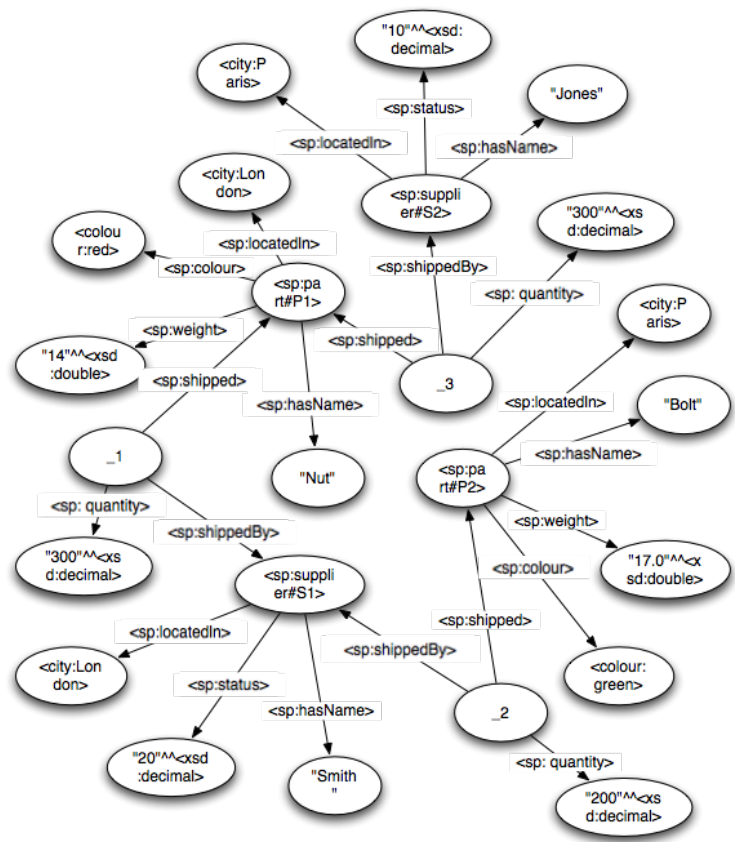
RDF supports three types of 'values'[23]

1. Names — URI's
2. Values — Typed Strings
3. Existential References — Blank-nodes

Only names are permitted to be used as Predicates, and Values are not permitted to be subjects; both these constraints are under review, there is a good case for abolishing them.

RDF Data is modeled as graphs of triples, which provides the absolutely simplest structure that could possibly work. The vocabulary of the data is defined implicitly by the data using the Concepts and Abstract Syntax[24] specification published by the W3C. A base semantics is defined rigorously in the RDF Semantics specification[25].

RDF Schema (RDFS)[26] is a standard that sits on top of RDF that allows developers to define standardised vocabularies to permit the applications to provide a basic shared interpretation of RDF data.



Binary Predicate Schema represented as a Directed-Labeled-Graph

the applications to provide a basic shared interpretation of RDF data.

Web Ontology Language (OWL) is a series of standards[27][28] that permit the definition of semantic-structures that can be applied to data defined using an RDFS vocabulary. These can be used to define deductive axioms for an semi-structured deductive DBMS, or to provide consistency constraints allowing applications to manipulate consistent data, and/or identify exceptions of interest.

This stack of standards, starting from a simple data description through vocabulary definition, to structure inverts the traditional relational approach to this problem, and in doing so provides the late-bound structure property that makes semi-structured data and changing requirements tractable.

Enterprise RDF

The RDF model performs admirably as a metadata description format. Current applications of RDF include:

- Rich Site Summaries (RSS)[6]
- Application Configuration (Mozilla)[29]
- Online Directories (Open Directory)[30]
- Social Networking (FOAF)[5]
- Citation and Bibliographical Information (Dublin Core)[4]

However in each of these cases RDF is being used to directly describe or identify a specific resource or set of resources. When we come to aggregate this information we find an immediate requirement to:

- Distinguish between independent RDF graphs
- Separate schema/vocabulary graphs from the 'data' graphs they describe
- Specify and track the provenance metadata associated with the RDF data

The only mechanism RDF provides explicitly is reification[25], a process where every statement is itself described in RDF and becomes a resource that can subsequently be described. However in practice this heavyweight solution is unnecessary as all these requirements are with respect to RDF graphs, not individual statements. Consequently the universally adopted approach is to introduce the concept of 'model' (also referred to as a context, or named-graph). This approach has since been made explicit in the recent work on a standard RDF query language SPARQL[31].

What this means in practice is that RDF datastores introduce a fourth node to each statement identifying the graph to which it belongs. Therefore while at the level of statement insertion and deletion Mulgara retains traditional RDF triple semantics, what it stores, and what users ultimately query, are quads. These models are constrained to be URIs, but beyond that they are treated as equal entities with subjects, predicates, and objects. It is therefore considered equally valid, and consequently there is no performance penalty associated with, querying "All models that contain the statement <SPO>", as opposed to "All the subjects with the property <PO in M>".

Once you have a datastore there are various properties that are required to support the various administrative tasks required of an enterprise scale store.

- Transactions
- Failure Recovery
- Backup/Restore
- Redundancy

Overview of Mulgara

Mulgara is an RDF datastore written in 100% Java utilising the NIO classes introduced in 1.4 to provide low level IO support. It provides full and equal support for named-graphs, and both query-based, and programmatic client APIs. In 2005 Mulgara was demonstrated to scale to over 250,000,000 quads on a 1GHz AMD Opteron, and was able to service 250 simultaneous queries with an average query time of 150ms[14]. Mulgara also provides a Resolver Service Provider Interface (SPI), that allows transparent integration with non-RDF data and datamodels. Current resolvers include MBox, Lucene full-text indexing, JDBC relational mapping, and even MP3 ID3-tags — also supported are 'virtual' resolvers that provides access to intervals and type-operations on XSchema datatypes, and string/uri prefixing[13]. Mulgara currently provides full serialisability of both implicit and user-demarcated transactions, lock-free live backups, and 0-time failure recovery. It achieves these features through a combination of a phase-tree based multiversion statement-store, and complete indexing of quad-prefix permutations.

Six indices are required to provide complete indexing of all possible prefixes to a 4-tuple. There are numerous choices of index sets, the six used by Mulgara are:

(0 1 2 3) (3 0 1 2)
(2 0 1 3) (3 2 0 1)
(1 2 0 3) (3 1 2 0)

The consequence of having all six indices available is that given any partially defined 4-tuple, the unification of the pattern with the store (and consequently with the graph) is defined as the interval defined by two $O(\log n)$ lookups on the appropriate index. Given the need to support ad-hoc querying on semi-structured data, this provides Mulgara with an $O(\log n)$ primitive sufficient to support a full query algebra. Iteration over the query result is then the trivial $O(m)$ case for a query result size m .

Each index is implemented as a multiversion blocked AVL-tree storing 4 64-bit longs as entries[14] — each long being mapped via a lookup table to global data (URI/Literal). This allows for efficient comparison for equality, which is the only operation generally performed within a query as it is sufficient to obtain the primitive interval, and also to calculate the conjunction (natural join) of primitive constraints. This more than compensates for the $\log(m \log m)$ sort normally required by the user to order the final query result.

The use of a multiversion phase tree is also critical to Mulgara's read performance. As all updates operate on their own phase (implemented as a shared copy-on-write AVL tree), reads can be performed without locking or synchronisation with update operations. For a more detailed examination of the existing implementation see Wood et al[14].

The Transactional Store Version 2 (XA2)

While the Mulgara team is proud of what Mulgara has achieved to date, we are not satisfied. Our estimates suggest that with the current store architecture we can reach 1-2 billion statements, at which point we run up against the limitations of the existing store layer. There are also several useful user-visible features whose development are infeasible without a radical rethink of how the custom store layer is designed. With this in mind over the past 2.5 years we have been radically rethinking the transactional store layer design. A proof-of-concept spike has been done that validates the core design[32].

As discussed in Vitters[33], and Lage[34], the first-order approximation of algorithmic complexity is not CPU time, but rather the number of IO operations performed. Ruemmler and Wilkes[35] measure the accuracy of this approximation and find that in absolute terms the RMS error for this model is 35% (without caching), and in excess of 100% when caching is provided. These results validate the intuition that guided the design

of Version 2 of the 'Transactional Statement Store' (XA2), that minimising disk-seeks is as important as minimising IO operations.

The other primary consideration in the design of XA2 is the substantial benefit the previous version derived from the lock-free operations enabled by the use of multiversioning. This led us to investigate various non-traditional datastructures and ultimately to Chris Okasaki's work on persistent datastructures[36] that provided a framework for analysing them. Adapting these methods to external storage provided us with a merge algorithm that reconciles the tension between the immutability of persistent data, and the need for locality required for efficient external IO. It is this locality issue that leads us to diverge from the prior literature on external-sorting that has generally preferred distribution to merge based sorts[33].

The design process of XA2 can be summarised thus[15]:

IO is minimised under two conditions:

- All data is compressed on disk
- Disk blocks are accessed sequentially

Contention is minimised if we eliminate locking — which can only be done if

- All data on disk is immutable

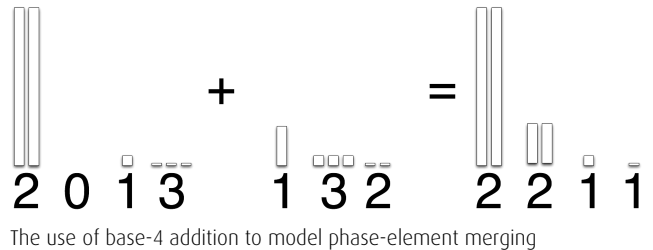
This leads to a conceptual model for an XA2 index consisting of a single sorted immutable array of compressed statements stored in a linear sequence of disk blocks. Conceptually queries consist of a binary search through the array, and inserts and deletes result in the building of a new sorted immutable array on disk.

To move from concept to design we address the various factors reality imposes on us that prevent us from using the conceptual design directly.

- Because compression interferes with binary search we adopt a skiplist[37] inspired data-structure that allows us to overlay a tree-like index structure over the sorted array.
- Because copying the entire array on each update is infeasible, we store each update as its own sorted array. The files of the resulting "phase-set" are then merged at query-time — building the conceptual single array on demand.
- Because querying one file per prior update is infeasible, we use a variation on persistent binomial heaps to limit the maximum number of files in each 'phase-set'.
- Because the worst-time cost of an update to a binomial heap involves a merge of 1/2 of the entire index we use lazy evaluation to defer merges that would cost more than a constant multiple of the size of the update.
- Because the deferred merge can leave an arbitrary number of updates unmerged, lazy evaluation breaks the size-limit guarantee provided by the binomial heap. We force the evaluation of the deferred merges as background tasks independent of ongoing queries.
- Because background merges require IO, contending with queries and updates, we prioritise merging elements of phase-sets "phase-elements" involved in current queries.

The result is an index that consists of a set of phases, one of which is the current phase and the target of all incoming queries; the rest are being retained because a prior query is still using them. Each phase consists

of a 'phase-set' of files, most of which are shared with other phases. Due to the files immutability, this sharing is possible without locking — the result is a significant improvement in our cache efficiency. Each phase-set consists of 'phase-elements'. Each element is a file that contains a sorted array of compressed statements and an index that restores $O(\log_2 N)$ access. All files are completely immutable, and all updates are progressively merged into the most recent committed "current" phase-set.



If we consider the structure of a single phase-element we see that it is fundamentally an external-memory version of a deterministic skip-list. This datastructure was referred to by Pugh in his original paper where he writes[37]:

If every $(2^i)^{\text{th}}$ node has a pointer 2^i nodes ahead, the number of nodes that must be examined can be reduced to $\text{ceil}(\log_2 n)$ while only doubling the number of pointers. This data-structure could be used for fast searching, but insertion and deletion would be impractical.

Fortunately as we are developing a persistent datastructure, insertion and deletion are by necessity handled externally to the skip-list and so this simpler version is ideal to our purposes. Also as this is an external structure, instead of nodes we use blocks, which changes the bases from 2 to B (the block size).

Detailed Design Overview

Note: By necessity this section elides some details and assumes a familiarity with Chris Okasaki's work on persistent datastructures[36]; Vitter and Arge's work on external-memory[33,34]; and details of the existing store [13,14,15,32].

The detailed design of a phase set.

The current design includes the use of a base-4 numerical heap[36]. In this structure the phase-set of size N is defined to be the merge of at worst $\log_4 N$ phase-digits; each phase-digit consisting of 0-4 phase-elements; each phase-element consisting of a single skip-list index. The merger of two phase-sets is therefore directly analogous to the addition of two base-4 numbers. While the naive implementation is linear in the size of the smallest number and therefore requires $O(N_{\min} \log_4 N_{\min})$ worst case IO's, we can exploit the comparative size of memory (M) to provide infinite carry-lookahead provided $M/B > 4 \log_4 N_{\min}$ — which results in only $O(N_{\min})$ IO's. This results in a linear sort IO complexity, which agrees with Vitter's optimal sorting complexity for one disk[33].

The detailed design of a phase-element.

At the head of each phase-element is a header-block. This block contains sufficient metadata concerning the phase-element to allow it to be read. It contains the following information:

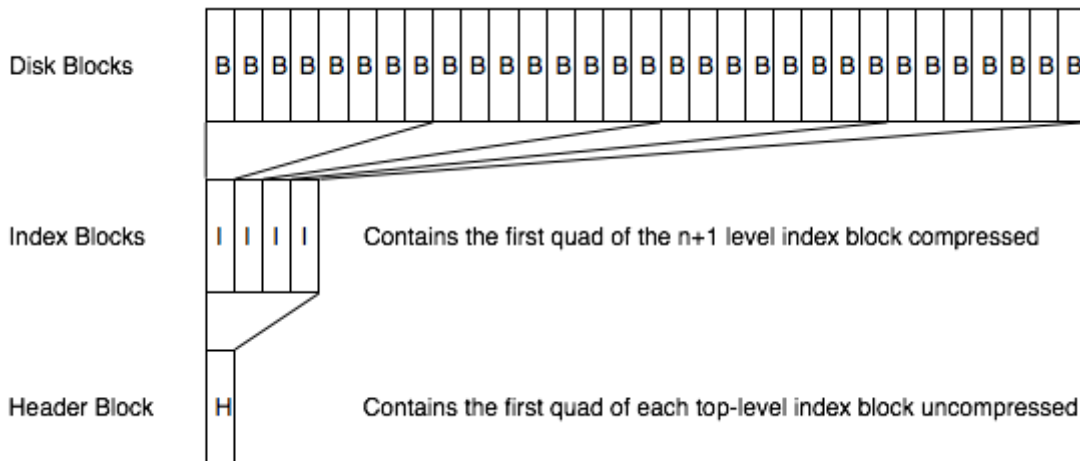
- a magic number
- a version number
- the block-size used
- the number of quads in the element
- the offset of each index-level
- the number of quads represented by the the incomplete block for each index-level
- the uncompressed first-quad in each block of the top-level index

All other data in the phase-element consists of quads and is compressed, only storing for each element of the quad, the delta from the same element in the previous quad. The skip-indices are placed at the end of the file, as due to the compression we do not know the exact amount of space to reserve.



The layout of Header/Data/Indices for a phase-element

Each skip-index contains a sorted array of the first quad of each block in the next index level. This gives us $\log_{B^3} 2B$ index levels, which given a typical value for B (32KB) will be 2 levels of skip index and 512 direct references from the header required for 2^{34} (17,179,869,184) quads. In fact we don't need to exceed 3 levels of skip index until we exceed 1000×2^{44} (3.5228×10^{16}) quads, which is so far beyond any reasonable expectation that search using this structure becomes a linear time operation!



The logical layout of a phase-element

The proposed compression scheme.

There are two compression schemes that have been proposed to reduce the number of IO's further. The first was proposed by David Makepeace[32] and involves bit-stuffing deltas using a 1-bit flag and 7-bits of data per byte. An alternative proposed by the author involves using a 1-byte flag followed by 4 byte-aligned deltas. Due to their use of deltas, the efficiency of both schemes increases with the size of the store. As Mulgara avoids fragmenting the allocation of longs, this results in extremely efficient compression. We anticipate ratios of the order of 25% for the bit-stuffing approach and 28% for the flag approach. Experimentation will be required to determine if the benefit of byte-alignment is sufficient to overcome the estimated 3% reduction in compression efficiency.



An example compressed quad demonstrating the four type-flags

Benefits of a Persistent Multiversion Datastructure Design

One crucially important consequence of the decision to base XA2 on persistent datastructures is the impact this decision has on the ability to support the administrative features mentioned before. By continuing to use multiversioned datastructures we retain the instant restart failure recovery currently supported by the

existing store. Of the other features Transactions and Backup/Restore are also supported by the current store, however their capabilities are substantially improved. Redundancy however is a new feature enabled by the new design. Both Federation and Replication are greatly facilitated by the use of persistency and multiversioning.

Lock free concurrent read-repeatable operation

By basing XA2 on immutable on-disk structures we avoid the need for locking in almost all cases. Specifically the only locking required is while obtaining the current phase, or in the resolving of conflicting writes. We are preparing a paper demonstrating that it is impossible to provide full serialisable transaction support[38] for an RDF datastore without devolving to a single-writer/write-queue model. However the use of a multiversion store does permit lock-free read-repeatable isolation[39]. The reason for this is the absence of any UPDATE operation in RDF. All modifications are purely expressed in terms of INSERT or DELETE, therefore while a deterministic protocol is required to reconcile conflicting insert/delete pairs the write itself can proceed lock-free.

We have developed a deferred conflict resolution algorithm that is consistent with read-repeatable isolation and still permits the provision of a serialisation service on top of the read-repeatable store. Writers also need to obtain an additional lock when committing. Given this algorithm both readers and writers need only hold a lock while copying a single pointer to obtain a reference to the current phase. Writers will also require a separate lock to be held while flushing a new phase root-block to disk. This block (the metaroot) identifies the files in the freshly committed phase-set. It is worth noting that this means that while using XA2 Mulgara will retain the property of always remaining consistent ondisk. This is important as it provides Mulgara with its capacity for 'instant' restarts.

Lock-free live backup/restore

Immutability can make backup/restore a trivial lock-free operation. As a reference to a phase is only a reference to a set of immutable files, a backup can be as simple as a tarball of those files. The corollary of this is that restore is only slightly more complicated — the metaroot must be updated to point at the backup's phase-set. Mulgara's current merge-restore semantic can also be accommodated by treating the backup as a data format, and treating it as an optimised load operation, appending the phase-set to the metaroot. However the most important property of backup and restore under XA2 is that the backup described above can be trivially performed against a live system. In fact a live backup is no more complicated than that of a suspended or shutdown server. Combined with a merge-restore on a live system being indistinguishable from a bulk 'pre-loaded' load alongside any concurrent writes in progress, this makes the decision of live vs.

suspended backup/restore strictly an administrative issue.

Federation

Reads suffer a small penalty from the decompression requirement, but gain substantial benefit from reduced bandwidth utilisation of the compressed block, and the improved IO times resulting from the locality of the skip-index structure. Queries should therefore see an appreciable performance improvement. Immutability provides additional benefits, including the opportunity to support multiple Mulgara instances sharing a single store. This will allow Mulgara to take maximum use of multi-core/cpu/bus systems to support 1000's of concurrent connections. This is possible as a result of the reduced coordination between Mulgara instances required to share immutable files. The only communication or coordination required between the separate Mulgara instances is small constant time operations at the start and end of each

transaction to ensure the transaction obtains a valid phase, and to ensure that write-commits are well ordered. Independence of this sort is the key to linear and super-linear scalability in any application, consequently XA2 is expected to scale linearly up to IO-saturation.

Replication

Both availability and a desire to scale beyond IO-saturation make replication a necessity. The lock-free live backup and restore provided by XA2 neatly sidesteps the complexity normally inherent in replication. Even the following more sophisticated replication strategies become tractable.

Master-slave replication is equivalent to the problem of live-backup/merge. As discussed above the use of persistence makes this problem much simpler. Replication then becomes a matter of pushing incremental phase-sets from the master to the slave, and performing a live-merge.

As **asynchronous peered replication** needs to support multiple write-nodes supporting both distributed serialisable transactions and the ability to merge concurrent updates, it is traditionally extremely complex to implement. However that complexity derives from two sources. The first is the requirement for serialisable transactions, which we address by only providing a serialisation service implemented on top of read-repeatable transactions. The second is the need to merge updates. RDF assists us here by not directly supporting UPDATE, but rather by only supporting INSERT and DELETE operations. This allows us to reduce the problem to tolerating missing phases-sets; supporting the injection of an incremental restore to provide a missing phase; and a coordination at the end of each transaction to agree on a canonical phase ordering.

Synchronous peered replication is in many ways simpler than asynchronous. Instead of focusing on the store-layer, we add distributed transaction support above it. We do this by blocking clients who enter PREPARE until their updates have been propagated to all nodes. This makes the phase manipulation logic discussed above unnecessary, however as it does require blocking writers until the update is synchronised it will almost certainly require a multicast based incremental update propagation, or this approach won't scale beyond 2-3 nodes.

Conclusion

We were able to differentiate between unstructured and structured data by the presence of a defined vocabulary and a semantics that define the 'structure' of structured data. We also recognise properties of fully-structured data that are common to both traditional relational and logic models — the open world assumption; the unique name assumption; and the existence of fixed cardinalities. All three properties are essential to the ability to define a rigorous universally-consistent structure. We can therefore define semi-structured data to be data with a defined vocabulary and semantics that does not make these three assumptions.

We have shown how to take a fully-structured relational schema and relax these properties, while retaining normalisation, and derive a graph based data-model compatible with RDF. This approach combined with a decoupling of the data, vocabulary, and semantics also suggests a possible solution to the schema lifecycle problems that plague relational implementations.

Finally we have presented a design for a large-scale RDF datastore based on multiversed persistent datastructures that meets the design constraints imposed by the need to consider external-memory, specifically the need to reduce the number of IO operations and the number of seek operations. This design highlights the critical importance of persistent datastructures to ensuring high-performance concurrent operations. With the increasing use of concurrency in even the most basic machines this lesson cannot be

emphasised too much.

References

- [1] Resource Description Framework (RDF), <http://www.w3c.org/RDF>
- [2] Platform for Internet Content Selection (PICS), <http://www.w3c.org/PICS>
- [3] W3C Issues Recommendation for Resource Description Framework (RDF), <http://www.w3c.org/Press/1999/RDF-REC>, Feb 1999, Press Release
- [4] Dublin Core Metadata Initiative, <http://dublincore.org/>
- [5] The Friend of a Friend Project, <http://www.foaf-project.org/>
- [6] Beget-Dov, G. et al., RDF Site Summary (RSS) 1.0, <http://web.resource.org/rss/1.0/spec>
- [7] Mindswap, <http://www.mindswap.org/2004/owl-s/>, University of Maryland
- [8] Wilbanks, J., W3C Workshop on Semantic Web for Life Sciences, Summary Report
- [9] WordNet, <http://wordnet.princeton.edu/>, Princeton University Cognitive Science Laboratory
- [10] The Fedora Project, <http://www.fedora.info/>
- [11] Bizer, C. et al., D2RQ V0.5 - Treating Non-RDF Relational Databases as Virtual RDF Graphs, User Manual and Language Specification, <http://sites.wiwiwiss.fu-berlin.de/suhl/bizer/D2RQ/spec/>
- [12] Berners-Lee, T., What the Semantic Web can represent, <http://www.w3.org/DesignIssues/RDFnot.html>, 1998
- [13] The Mulgara Project, <http://www.mulgara.org/>
- [14] Wood, D. Gearon, P. Adams, T., Kowari: A Platform for Semantic Web Storage and Analysis, Xtech-2005, May, 2005
- [15] Muys, A., Six Reasons why Mulgara's XA2 Storage Layer should matter to you, Discussion Paper, Netymon, 2006, <http://www.netymon.com/papers.html>
- [16] Codd, E., A Relational Model of Data for Large Shared Data Banks, CACM 13, No. 6, 377-387, 1970
- [17] Cluet, S., Modeling and Querying Semi-structured Data, SCIE, Lecture Notes in Computer Science, Volume 1299, 192-213, 1997
- [18] Buneman, P. et al., A Query Language and Optimisation Techniques for Unstructured Data, SIGMOD Conference 1996, 505-516
- [19] McHugh, J. et al., Lore: A Database Management System for Semi-structured Data, SIGMOD Record, Volume 26, No 3, 54-66, 1997
- [20] Papakonstantinou, Y. et al., Object Exchange Across Heterogeneous Information Sources, 11th Conference on Data Engineering, IEEE Computer Society, 251-260, 1995
- [21] Abiteboul, S., Querying Semi-Structured Data, ICDT, Lecture Notes in Computer Science, Volume 1186, 1-18, 1997
- [22] Date, C., An Introduction to Database Systems 8th Ed, Addison-Wesley, 2004
- [23] RDF Primer, W3C Recommendation, Feb 2004, <http://www.w3.org/TR/rdf-primer/>
- [24] Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C Recommendation, Feb 2004, <http://www.w3.org/TR/rdf-concepts/>
- [25] RDF Semantics, W3C Recommendation, Feb 2004, <http://www.w3.org/TR/rdf-mt/>
- [26] RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, Feb 2004, <http://www.w3.org/TR/rdf-schema/>
- [27] OWL Specification Development, W3C Recommendations, Feb 2004, <http://www.w3.org/2004/OWL/#specs>
- [28] OWL Web Ontology Language Overview, W3C Recommendation, Feb 2004, <http://www.w3.org/TR/owl->

[features/](#)

- [29] Resource Description Framework (RDF), Semantic Web Technologies in Mozilla, <http://www.mozilla.org/rdf/doc/>
- [30] Open Directory RDF Dump, <http://rdf.dmoz.org/>
- [31] SPARQL Query Language for RDF, W3C Working Draft, current version (4 Oct 2006), <http://www.w3.org/TR/rdf-sparql-query/>
- [32] Wood, D., Scaling the Kowari Metastore, in Dean, M., et al. (Eds.): WISE 2005 Workshops, LNCS 3807, pp. 193-198, 2005
- [33] Vitter, J., External Memory Algorithms and Data Structures, External Memory Algorithms and Visualisation, 1-38, American Mathematical Society Press, 1999
- [34] Arge, L., Efficient External-Memory Data Structures and Applications, PhD Dissertation, University of Aarhus, 1996
- [35] Ruemmler, C. and Wilkes, J., An Introduction to disk drive modeling, IEEE Computer, Volume 27, No 3, 17-28, 1994
- [36] Okasaki, C., Purely Functional Data Structures, Cambridge University Press, 1998
- [37] Pugh, W., Skip Lists: A Probabilistic Alternative to Balanced Trees, CACM, Volume 33, Issue 6, 668-676, 1990
- [38] Bernstein, P. Concurrency Control and Recovery in Database Systems, 1-45, Addison-Wesley, 1987
- [39] Henschen, L. and Lee, J., A Highly Concurrent Transaction Management Model, IEEE international conference on computing and information, 1994, also <http://citeseer.ist.psu.edu/350718.html>

FACTORS AFFECTING ADOPTION OF OPEN SOURCE SOFTWARE BY INDIVIDUALS

Young, Alison, Griffith University, Nathan 4111 Brisbane, Australia,
Alison.Young@student.griffith.edu.au

Abstract

This research aims to identify factors that affect an individual when adopting open source software. An examination of the literature includes a definition of open source software and brief accounts of its use within governments and organisations. The research question is to identify what factors influence the decision to adopt open source software in end-users, or individuals. The diffusion of innovation model was used to interpret findings from previous research to identify factors. The primary factor identified was regarding ease of use and complexity of open source software. Research that utilises the multi-level framework in addition to a MIS discipline theory will provide a sound basis for continuing research on this topic.

Keywords: adoption, end-users, open source software.

1 INTRODUCTION

This paper presents an investigation of literature demonstrating that end-user adoption of open source software has been stalled by an assortment of issues. Issues affecting the rate of end-user adoption of open source software include factors such as accessibility, ease of use, and knowledge of its existence. The paper will also explain routes for future research to better understand whether any change in the rate of adoption or issues affecting rates of adoption have occurred in the period since the literature was published.

The purpose of this paper is to inspire investigation into the opportunities that use of open source software can bring to everyday end-users. With such opportunity freely available, information systems researchers need to explore why end-users are not utilising this source of software. The purpose of this paper is to provide insight into what factors are attributed to the rate of adoption of open source software among end-users. The research question being addressed in this paper is what are the factors that affect end-user adoption of open source software? With knowledge of these factors in hand, it will be possible for developers and distributors of open source software to find ways to circumvent issues and thus enable their end-user numbers to grow.

The scope of this paper is to present findings from research and articles from literature that describe and explain why the rate of adoption of open source software is lagging behind that of proprietary software. The background literature on open source software will provide some factors, or issues, that can be used to analyse the affects on the rate of end-user adoption.

Preliminary literature synthesis will then enable further research to be done on this topic which may include development of a case study to analyse the current situation and make conclusions as to whether factors identified in the literature are still present. Explanations from open source software literature will provide some background details about end-user adoption.

Use of a case study in association with the theoretical framework will allow factors and issues to be identified that are relevant to the adoption of open source software. It is important to remember that for the purpose of this paper, open source software will be deemed an innovation.

The following paper will contain a literature review comprising background information about open source software and where it has been used in governments and organisations. Adoption of open source software within those environments will also be addressed and then related to individual adoption. Theoretical frameworks will be used to analyse case studies and introduce the use of a multi-level framework for researching open source.

2 LITERATURE REVIEW

Due to the nature of this research topic, much of the literature that has been published does not fall under the classification of an academic quality publication. This is primarily due to the nature and actions of the open source community and the belief that information should be freely available.

Another reason for the lack of academic papers may be that there has simply been a lack of academic level research on this particular topic. This research will serve to highlight the importance and interest of this topic. An exploratory study has revealed that there is a recent but growing interest in open source software and how it may be used to provide software in developing countries due to its low or zero cost to use . This is a growing interest area and an important topic, however only very recently have approaches been devised to study open source as a wider concept by use of a multi-level framework . Further details pertaining to this framework and how levels can be used to examine different aspects of open source will be discussed in the research framework and method section . In addition different discipline theories and their relevance to researching open source will be discussed, in particular diffusion of innovation which is the theoretical framework used to examine this topic .

2.1 What is Open Source Software?

The first and most important thing to do for this research topic is to establish a definition for open source software. The Open Source definition version 1.9 contains 10 clauses which specify the criteria which must be met if software is to be classed as open source .

The clauses are as follows :

1. Free Redistribution – There shall be no restriction to any party from selling or giving away the software and also not require payment of royalties or fees for the sale.
2. Source Code – Source code must be included, and must allow distribution in source code as well as in compiled form; if code is unavailable there must be a well advertised way of obtaining code that is able to be modified by the programmer.
3. Derived Works – Modification of software must be allowed and the modified program must be distributed under the same license as the original software.
4. Integrity of the Author's Source Code – There may be restrictions on source-code from being distributed in modified form *only* if the license allows the distribution of "patch files", else modified works may be required to change the name or version number from the original.
5. No Discrimination Against Persons or Groups – No groups or persons may be discriminated against.
6. No Discrimination Against Fields of Endeavour – There must be no restrictions on the field in which the software is used.
7. Distribution of License – the license must apply to all who receive the program through redistribution without the need to agree to another license.

8. License Must Not Be Specific to a Product – the license must not be specific to a program that is part of a particular distribution.
9. License Must Not Restrict Other Software – no restrictions on software that is distributed with the licensed software must exist.
10. License Must Be Technology-Neutral – the licence must not be dependant on any particular technology or interface style.

While the details of the criteria that ensures software is in fact open source may appear exhaustive. It is ensuring that not just the developers of software have access to the source code and the right to make modifications that will enhance the software for every user. This freedom to alter, customise and redistribute is often in association with the freedom of obtaining the product without a monetary exchange taking place.

2.2 Organisations and Open Source Software

Organisations have long been aware of open source software and its benefits. Use of open source software is increasing in organisations and organisational awareness is growing. In particular recent literature is discussing how open source software may be used in organisations and what benefits can be found from development and use of software in-house in conjunction with sharing of developed software between organisations .

How open source software is adopted and used in organisations is outside the focus of the research. The acknowledgement of organisational involvement and use of open source software is present to demonstrate that the advantages of open source software have been discovered and utilised by the corporate sector.

2.3 Governments and Open Source Software

Governments are increasingly adopting open source software for their departments. This is especially prevalent in European countries which are adopting open standards and deliberately avoiding proprietary software developers .

Evans and Reddy (2003) surveyed government proposals and initiatives that concern open source software and found that many countries were adopting the use of open source software. Their findings revealed that Germany and France were at the forefront of promoting open source . In particular the French public sector is moving towards a complete open-source infrastructure . Brazil, Italy, Spain and Venezuela have all passed resolutions that either force or encourage the government of each nation to use open source software .

Demonstration of government involvement and public resolution for their governments to use open source software is a promising sign that individuals are becoming increasingly aware of open source software.

2.4 Adoption of Open Source Software

Literature shows that organisations are adopting open source software but more interestingly lists have been produced of technical and management requirements that the open source software must satisfy . While such lists may be useful for determining important factors within an organisation, similar factors will likely not arise when analysing adoption on an individual scale, however using organisational requirements allows for comparison of individual equivalents. Other studies have used a grounded theory approach to investigate adoption, but again this has taken place on an organisational level .

While quite a number of conference papers have focussed on organisational adoption of open source software, there are a few papers which have taken an individual perspective. An end-user is also known as an individual user, somebody much like you or me. The needs, perceptions and capabilities of an end-user differ from those of an organisation. It is for this reason that organisational based research can not be directly applied to end-users.

The article by Sullivan (2001) presents an end-user perspective piece on the perceived factors that are affecting adoption of open source software. This type of paper is useful as research in this topic as while it contains author bias, it does give a list of issues and concerns . Using such a list as a starting point it is then possible to seek out similar factors in related literature which will assist in standardising factors and aiding to remove bias. The advantage of end-user authored articles is also that they present an interesting picture of the perceptions that some end-users may have towards open source software.

Research conducted by Hussein (2003) in a Malaysian university sought to discover why, despite the growing phenomenon of open source software in tertiary level institutions across a number of countries, the rate of adoption among Malaysian computer science students was minimal. A survey was conducted and the results show that while many students have used open source software operating systems, only approximately 30% of students surveyed enjoyed it . Most students who owned a computer didn't have an open source operating system installed, those who did were doing so as it was required for school assignments . The factor identified that describes why students didn't like open source software is that it is hard to use and not user-friendly, the next most stated factor was that the interfaces were unattractive .

While the research by Hussein (2003) was a very small study, it does give insight into what an end-user finds most important in software that they are using. Benefits of open source software identified by students were primarily related to learning by source code analysis. This may not be indicative of a typical end-user whom is not a student.

3 RESEARCH FRAMEWORK AND METHOD

The research method utilised for this paper is that of a literature review. The results and findings of the research conducted by others will be used to introduce and substantiate the importance of the research topic. The topic of research is analysis of documents commenting on the adoption of open source software by individuals and factors affecting adoption. Due to the nature of open source software development and use, there has been little research on its use outside of organisational contexts. Individual use and adoption has not been a topic with much accompanying research. Research that has been conducted on this topic rarely falls under the classification of 'academic research' and thus demonstrates a need for academic level research to be conducted on this topic.

Most recently a series of papers have been published that describe a research agenda for studying open source . This series of three papers commences by describing a multi-level framework of which the discrete levels can be used to analyse information systems through open source software . There are five levels described in the framework; the artifact, the individual, the group/ the project/ the community, the organisation and the broader societal perspective . Of these levels the one addressed in this paper will be that of the individual, subsequent studies may wish to address how the individual corresponds to the broader societal perspective. Niederman et al (2006a) state that examining a level individually lends insight to that level but to gain the bigger picture, relationships between the levels must be examined also. For this particular research paper, the level of the individual will be focussed on. However within a single level descriptive studies may be done that "examine the nature of the variables at that level" . It must be noted that in some studies the role of an individual is typically that of a developer and not a user . Research is easier when focussing on developers rather than users, however using a diffusion and adoption model allows for consideration of other research topics as given in Niederman et al. (2006a). These research topics include such areas as examination of patterns of the number of users changing over time and what decision making process does an individual undertake when deciding between proprietary or open source software .

With a level of analysis in mind it is then possible to use the second paper to determine which discipline theories from information systems are most relevant to the level in question. The theories that have been included in the paper by Niederman et al. (2006b) include adaptive structuration theory, agency theory, complexity theory, diffusion theory, game theory, social network theory and transaction cost theory. Each of the abovementioned theories was selected based on "the potential for examining open source issues and their existing base of application within the MIS literature". The theory selected for use in this research paper is diffusion theory which was defined by Rogers (1995) and the suitability of using this theory to analyse open source has been explained by Niederman et al. (2006b). While there have been no studies from classical diffusion literature, the concept of innovation has enabled IS researchers to adopt alternate methods of examining innovation . From the literature differentiation in communication of innovations occurs between open source and so called typical approaches; where in open source software the communication travels in a bottom-up style from the users towards the developers .

In attempt to gain an understanding of the topic in light of the theoretical framework, a similar case study will be used and comparisons will be drawn from the case study and research topic. This will aim to show relevance of the research topic to the theoretical framework and allow areas of further investigation to be identified. The theoretical framework section contains a description of the diffusion of innovation framework by Rogers (1995) and the following method section demonstrates how the framework is used to interpret a case study from a university environment.

3.1 Theoretical Framework

The theoretical framework used in this research is Diffusion of Innovation (DOI) theory . This theory was selected due to the ability to assess both individual and prior condition variables which affect the innovation decision and to also trace factors that affect adoption or rejection through the four phases shown in the model. A more in depth description of this model follows.

Other models that were considered but later rejected include the Technology Acceptance Model , and DeLone and McLean's Information Systems Success Model . The reasons for rejection of these two models are given in the following paragraphs.

The Technology Acceptance Model investigates user acceptance of information systems based on perceptions about ease of use and usefulness then later the actual ease of use and usefulness . While user acceptance is related to the research topic, information systems specifically nor simply focussing on the ease of use and usefulness encompasses the full scope of the study.

The DeLone and McLean Information Systems Success Model was considered due to its user satisfaction components and how net benefits affect user satisfaction which in turn drives use of the system . While some aspects of the model may be relevant to the research topic, the focus of the model is not on what factors drive adoption. This model may be useful in future studies that examine how successful open source software is among end-users following adoption.

3.2 Innovation-Decision Process Model

The Innovation-Decision Process Model shows the process an individual goes through when deciding whether to accept or reject an innovation as found by diffusion scholars . A diagram of the stages in the Innovation-Decision process can be found in Figure 1 has been based on the model found in Rogers (1995, pg.163).

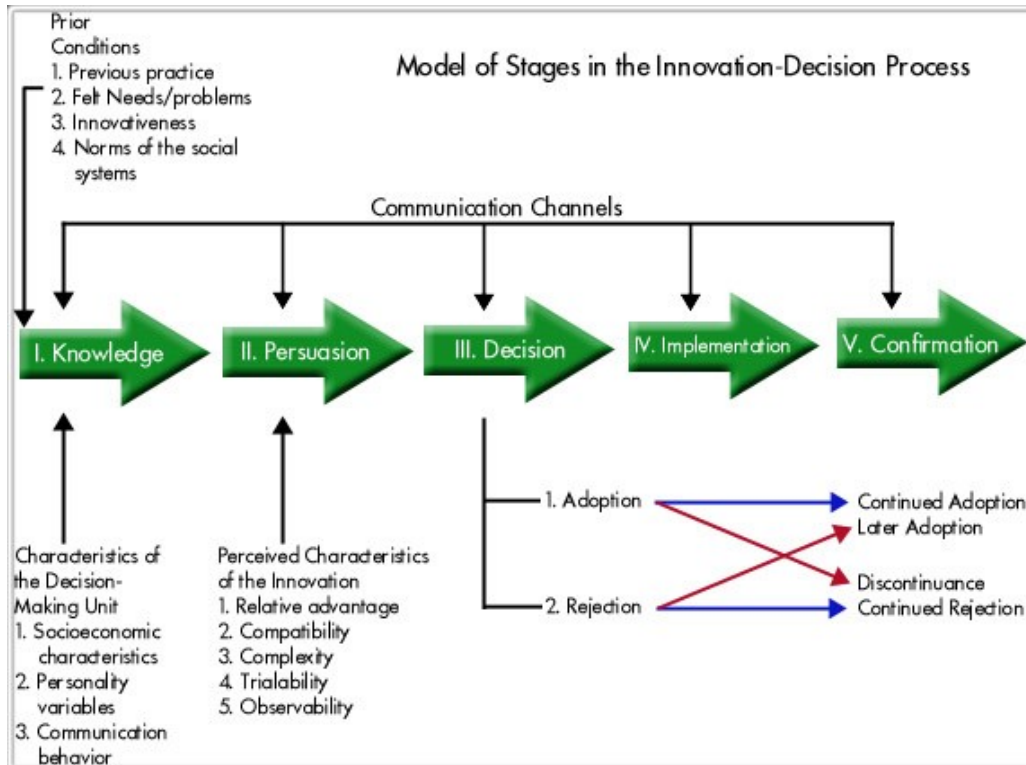


Figure 1. A Model of Stages in the Innovation-Decision Process

This model was deemed appropriate for the research as the focus is on an individual end-user and the factors that play a role in determining whether that individual will accept or reject an innovation. Despite the model encompassing all processes of the innovation acceptance/rejection decision, only the first and second processes will be directly applicable to the research topic.

Factors from the model that are expected to coincide with factors found in the literature are the prior conditions, characteristics of the decision-making unit and perceived characteristics of the innovation.

4 METHOD

The research question of this paper is what factors from the literature that can be used to explain end-user adoption or rejection of open source software. Using an aspect from the multi-level framework it is possible to focus the research in such a way to gain maximum benefit and relevance. For this research the individual level has been selected as the primary focus however Niederman et al. (2006) do stress that to gain a complete understanding of open source, the entire multi-level framework is required. This ensures that all subtleties that occur between the levels are acknowledged and the effects and influences each level has on the others is understood.

The article by Sullivan (2001) presented a list of factors that have been personally and socially identified describing issues and concerns about open source software, specifically the Linux operating system. The most important issue from Sullivan's (2001) article is that Linux is presently quite difficult and complex for new users, in particular as most users converting to Linux are likely to be Windows users. The paper by Hussein (2003) discussed research that was conducted among students and presented results describing actual use statistics and issues that have negatively affected the adoption of open source software by students.

The common factor between both papers is the importance of ease of use, in both articles this is actual ease of use as participants have actually used open source software and were not merely stating their perceptions. The factor about aesthetics while not mentioned by Sullivan (2001) may play an important part to other end-users who are not so technically minded. Whereas some of the more specialist issues raised by Sullivan (2001), such as Windows networking and cohesion of vision and standards, may not be mentioned if a variety of end-users are surveyed. Whereas an end-user survey revealed that the lack of end-user support and aesthetic features of open source software were genuine issues raised by end-users .

The theoretical framework selected is deemed to be appropriate as factors identified in the literature are able to be mapped onto the Model of Stages in the Innovation-Decision Process . This is demonstrated in the factors associated with usability and how they can be attributed to characteristics of the decision-making unit, in this example, the end-user. These characteristics form part of the first stage in the model, knowledge. Usability is also linked to complexity, which is a perceived characteristic of the innovation in the persuasion stage of the model . Based on feedback from end-user research complexity is a key issue of the persuasion stage where individuals are receptive to the innovation and will use particular characteristics to evaluate the innovation prior to moving to the decision stage . Of the existing attributes within the persuasion stage of the DOI model additional attributes have been identified in the study of technology diffusion .

These attributes include "critical mass, cost and social approval" . Consideration of the additional attributes is important for this research as issues of cost and social approval may prove to play a larger part than previously anticipated. Factors such as social approval may depend heavily on the software aesthetics for it to be deemed acceptable by society. For each of the stages within the DOI model, each must be examined using an individual basis for analysis. In purely considering an individual, certain elements of the DOI framework may become more or less important to that particular level from the multi-level framework. Due to the complexity of comprehending multiple layers of analysis Niederman et al. (2006b) recommended that research of the impact of features and complexity be done for specific open source software applications on the consequences of implementing software in varying contexts. By cumulating several separate studies it will be possible for researchers to understand successful adoption of open source software and the preconditions required for that success .

5 CONCLUSIONS

Based on the literature review and analysis of the findings it can be concluded that the research topic investigating end-user adoption of open source software is important and interesting. Due to the lack of academic quality literature there is much room for additional research to be conducted. Research by Niederman et al. (2006a, 2006b) has highlighted the levels in which can be used to study open source software and have provided seven discipline theories which may be used to research and analyse findings based on open source software. The theoretical framework selected for this research is one that has been identified as being suitable for further research in this topic area.

This research has identified the primary factor that is important to end-users when deciding to adopt open source software as a new innovation. This factor is related to complexity and ease of use from a end-user perspective and has been identified following analysis of a case study and also a critique article. Open source software is a relatively new innovation and research on this topic is valuable for not only developers and current end-users of open source software but also users who are considering the opportunities and benefits open source software may allow them.

This research has demonstrated that identification of factors influencing the adoption decisions of end-users is a topic that is worth pursuing. Diffusion of the open source innovation is growing and developing a more thorough and complete analysis of factors affecting the rate of diffusion will provide the means to increase growth. The publication of research by Niederman et al (2006a, 2006b) demonstrates that research in this area is worth pursuing and that by using a number of existing theories, each of the levels within the multi-level framework in addition to the relationships between levels may be fully understood. Further research in this domain will increase understanding and also clarify and expand on several approaches to investigating open source software.

References Dedrick, J. & West, J. (2003) An Exploratory Study into Open Source Platform Adoption. HBS - MIT Sloan Free/Open Source Software Conference: New Models of Software Development. Boston.

[DeLone](#), W. H. & [McLean](#), E. R. (1992) Information Systems Success: The Quest for the Dependent Variable. Information Systems Research, 3,1, 60-95.

Evans, D. S. & Reddy, B. J. (2003) Government Preferences for Promoting Open Source Software: A Solution in Search of a Problem. Michigan Telecommunications and Technology Law Review, 9,2, 313-394.

Hussein, N. (2003) The Acceptance of Open Source Among Computer Science Students In Universiti Sains Malaysia. Free & Open Source Software Conference 2003. Selangor, Malaysia.

Niederman, F., Davis, A., Greiner, M. E., Wynn, D. & York, P. T. (2006a) A Research Agenda for Studying Open Source I: A Multi-Level Framework. Communications of the Association for Information Systems, 18, 129-149.

Niederman, F., Davis, A., Greiner, M. E., Wynn, D. & York, P. T. (2006b) Research Agenda for Studying Open Source II: View Through the Lens of Referent Discipline Theories. Communications of the Association for Information Systems, 18, 150-175. Open Source Initiative (2006) The Open Source Definition version 1.9. Open Source Initiative. Rogers,

E. M. (1995) Diffusion of innovations, Free Press New York. Rogers, E. M. (2006) Chapter 5: The Innovation-Decision Process, Diffusion of Innovations: Part I. Good Ideas do not Sell themselves.

Skidmore, D. (2005) The future of software as a business artefact. IN SCOTTO, M. & SUCCI, G. (Eds.) First International Conference on Open Source Systems. Genova.

Sullivan, P. (2001) What's Holding Linux Back? Anandtech.com. Szajna, B. (1996) Empirical Evaluation of the Revised Technology Acceptance Model. Management Science, 42,1, 85-92.

Wang, H. & Wang, C. (2001) Open Source Adoption: A Status Report. IEEE Software. Conference Information

Open Source Software for Computational Modelling of Fluid Flow

O. M. Nielsen* S. G. Roberts†

28 December 2006

Abstract

Modelling the effects on the built environment of natural hazards such as riverine flooding, storm surges and tsunami is critical for understanding their economic and social impact on our urban communities. Geoscience Australia and the Australian National University have developed a hydrodynamic inundation modelling tool called ANUGA to help simulate the impact of these hazards. The core of ANUGA is a PYTHON implementation of a finite-volume method for solving the conservative form of the Shallow Water Wave equation. In this paper we describe the model, the architecture and some applications. ANUGA has recently been released as Open Source to enable free access to the software and allow the scientific community to use, validate and contribute to the software in the future.

1 Introduction

The Indian Ocean tsunami on 26 December 2004 demonstrated the potentially catastrophic consequences of natural hazards. While the scale of the impact from such events is not common, smaller-scale tsunami regularly threaten coastal communities around the world. Earthquakes which occur in the Java Trench near Indonesia (e.g. [8] or [1]) and along the Puysegur Ridge to the south of New Zealand (e.g. [3]) have potential to generate tsunami that may threaten Australia's northwestern and southeastern coastlines. In

*Risk Assessment Methods Project, Geospatial and Earth Monitoring Division, Geoscience Australia, Symonston, AUSTRALIA. <mailto:Ole.Nielsen@ga.gov.au>

†Department of Mathematics, Australian National University, Canberra, AUSTRALIA. <mailto:stephen.roberts@anu.edu.au>

addition, the preferential development of Australian coastal corridors means that inundation from hydrological disasters such as tsunami or storm-surge of even a few hundred metres beyond the shoreline has increased potential to cause significant disruption and loss. The extent of inundation is critically linked to the event, tidal conditions, bathymetry and topography and it not feasible to make impact predictions using heuristics alone.

Hydrodynamic modelling allows impacts from flooding, storm-surge and tsunami to be better understood, their impacts to be anticipated and, with appropriate planning, their effects to be mitigated. Geoscience Australia in collaboration with the Mathematical Sciences Institute, Australian National University, is developing a software application called ANUGA to model the hydrodynamics of floods, storm surges and tsunami. These hazards are modelled using the conservative shallow water equations which are described in section 2. In ANUGA these equations are solved using a finite volume method as described in section 3. A more complete discussion of the method can be found in [5] where the model and solution technique is validated on a standard tsunami benchmark data set. Section 4 describes the software implementation and the API while section 5 presents some validation results.

2 Model

The shallow water wave equations are a system of differential conservation equations which describe the flow of a thin layer of fluid over terrain. The form of the equations are:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = \mathbf{S}$$

where $\mathbf{U} = [h \quad uh \quad vh]^T$ is the vector of conserved quantities; water depth h , x -momentum uh and y -momentum vh . Other quantities entering the system are bed elevation z and stage (absolute water level) w , where the relation $w = z + h$ holds true at all times. The fluxes in the x and y directions, \mathbf{E} and \mathbf{G} are given by

$$\mathbf{E} = \begin{bmatrix} uh \\ u^2h + gh^2/2 \\ uvh \end{bmatrix} \quad \text{and} \quad \mathbf{G} = \begin{bmatrix} vh \\ vuh \\ v^2h + gh^2/2 \end{bmatrix}$$

and the source term (which includes gravity and friction) is given by

$$\mathbf{S} = \begin{bmatrix} 0 \\ -gh(z_x + S_{fx}) \\ -gh(z_y + S_{fy}) \end{bmatrix}$$

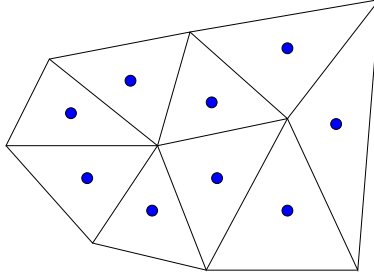


Figure 1: Triangular mesh used in our finite volume method. Conserved quantities h , uh and vh are associated with the centroid of each triangular cell.

where S_f is the bed friction. The friction term is modelled using Manning's resistance law

$$S_{fx} = \frac{u\eta^2\sqrt{u^2 + v^2}}{h^{4/3}} \text{ and } S_{fy} = \frac{v\eta^2\sqrt{u^2 + v^2}}{h^{4/3}}$$

in which η is the Manning resistance coefficient.

As demonstrated in our papers, [5, 9] these equations provide an excellent model of flows associated with inundation such as dam breaks and tsunamis.

3 Finite Volume Method

We use a finite-volume method for solving the shallow water wave equations [9]. The study area is represented by a mesh of triangular cells as in Figure 1 in which the conserved quantities of water depth h , and horizontal momentum (uh, vh) , in each volume are to be determined. The size of the triangles may be varied within the mesh to allow greater resolution in regions of particular interest.

The equations constituting the finite-volume method are obtained by integrating the differential conservation equations over each triangular cell of the mesh. Introducing some notation we use i to refer to the i th triangular cell T_i , and $\mathcal{N}(i)$ to the set of indices referring to the cells neighbouring the i th cell. Then A_i is the area of the i th triangular cell and l_{ij} is the length of the edge between the i th and j th cells.

By applying the divergence theorem we obtain for each volume an equation which describes the rate of change of the average of the conserved quantities within each cell, in terms of the fluxes across the edges of the cells and the effect of the source terms. In particular, rate equations associated with

each cell have the form

$$\frac{d\mathbf{U}_i}{dt} + \frac{1}{A_i} \sum_{j \in \mathcal{N}(i)} \mathbf{H}_{ij} l_{ij} = \mathbf{S}_i$$

where

- \mathbf{U}_i the vector of conserved quantities averaged over the i th cell,
- \mathbf{S}_i is the source term associated with the i th cell, and
- \mathbf{H}_{ij} is the outward normal flux of material across the ij th edge.

The flux \mathbf{H}_{ij} is evaluated using a numerical flux function $\mathbf{H}(\cdot, \cdot; \cdot)$ which is consistent with the shallow water flux in the sense that for all conservation vectors \mathbf{U} and normal vectors \mathbf{n}

$$H(\mathbf{U}, \mathbf{U}; \mathbf{n}) = \mathbf{E}(\mathbf{U})n_1 + \mathbf{G}(\mathbf{U})n_2.$$

Then

$$\mathbf{H}_{ij} = \mathbf{H}(\mathbf{U}_i(m_{ij}), \mathbf{U}_j(m_{ij}); \mathbf{n}_{ij})$$

where m_{ij} is the midpoint of the ij th edge and \mathbf{n}_{ij} is the outward pointing normal, with respect to the i th cell, on the ij th edge. The function $\mathbf{U}_i(x)$ for $x \in T_i$ is obtained from the vector \mathbf{U}_k of conserved average values for the i th and neighbouring cells.

We use a second order reconstruction to produce a piece-wise linear function construction of the conserved quantities for all $x \in T_i$ for each cell (see Figure 2. This function is allowed to be discontinuous across the edges of the cells, but the slope of this function is limited to avoid artificially introduced oscillations.

Godunov's method (see [7]) involves calculating the numerical flux function $\mathbf{H}(\cdot, \cdot; \cdot)$ by exactly solving the corresponding one dimensional Riemann problem normal to the edge. We use the central-upwind scheme of [2] to calculate an approximation of the flux across each edge.

In the computations presented in this paper we use an explicit Euler time stepping method with variable timestepping adapted to the observed CFL condition.

4 Software Implementation

ANUGA is mostly written in the object-oriented programming language PYTHON with computationally intensive parts implemented as highly optimised shared objects written in C.

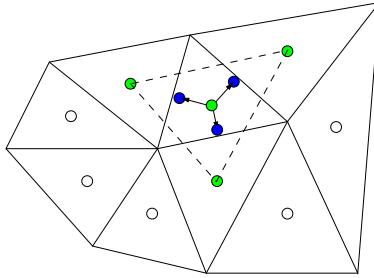


Figure 2: From the values of the conserved quantities at the centroid of the cell and its neighbouring cells, a discontinuous piecewise linear reconstruction of the conserved quantities is obtained.

PYTHON is known for its clarity, elegance, efficiency and reliability. Complex software can be built in PYTHON without undue distractions arising from idiosyncrasies of the underlying software language syntax. In addition, PYTHON's automatic memory management, dynamic typing, object model and vast number of libraries means that software can be produced quickly and can be readily adapted to changing requirements throughout its lifetime.

The fundamental object in ANUGA is the `Domain` which inherits functionality from a hierarchy of increasingly specialised classes starting with a basic structural `Mesh` to classes implementing the finite-volume scheme described in section 3. Other classes are `Quantity` which represents values of one variable across the mesh along with their associated operations, `Geospatial_data` which represents georeferenced elevation data and a collection of `Boundary` classes which allows for a 'pluggable' way of driving the model. The conserved quantities updated automatically by the numerical scheme are stage (water level) w , x -momentum uh and y -momentum vh . The quantities elevation z and friction η are quantities that are not updated automatically but can be changed explicitly during run-time if the user wishes to do so.

To set up a scenario the user specifies the study area along with any internal regions where increased mesh resolution is required. External edges may be labelled using symbolic tags which are subsequently used to bind boundary condition objects to tagged segments of the mesh boundary. The mesh is then generated using ANUGA's built-in mesh generator and converted into the `Domain` object which provides all methods used to setup and run the flow simulation. Figure 3 shows an example of a mesh generated by ANUGA.

Next step is to setup initial conditions for each `Quantity` object. For

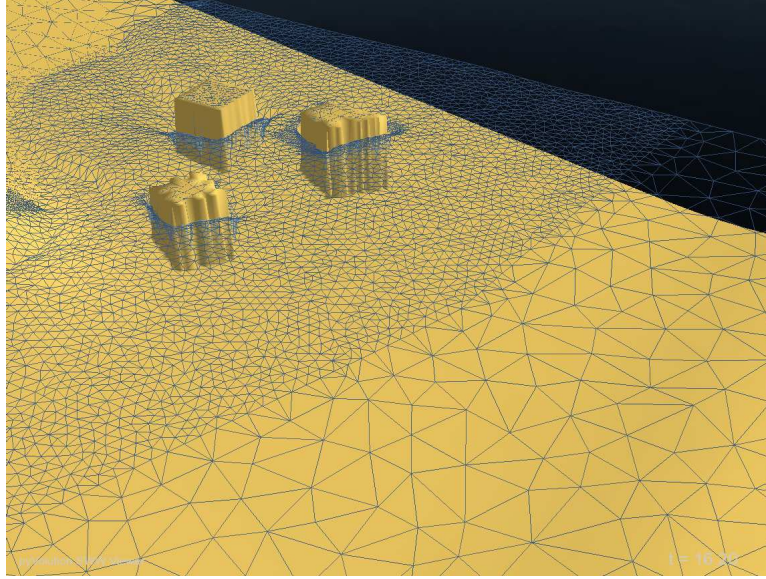


Figure 3: Triangular mesh used in our finite volume method. Conserved quantities h , uh and vh are associated with each triangular cell.

the elevation z this is typically obtained from bathymetric and topographic data sets. Setting initial values for quantities is done through the method `domain.set_quantity(name, X, location, region)` where `name` is the name of the quantity (e.g. `'stage'`, `'xmomentum'`, `'ymomentum'`, `'elevation'` or `'friction'`). The variable `X` represents the source data for populating the quantity and may take one of the following forms:

- A constant value as in `domain.set_quantity('stage', 1)` which will set the initial water level to 1 m everywhere.
- Another quantity or a linear combination of quantities. If `q1` and `q2` are two arbitrary quantities defined within the same domain, the expression `domain.set_quantity('stage', q1*(3*q2 + 5))` will set the stage quantity accordingly. One common application of this would be to assign the stage as a constant depth above the bed elevation.
- An arbitrary function (or a callable object), `f(x, y)`, where `x` and `y` are assumed to be vectors. The quantity will be assigned values by evaluating `f` at each location within the mesh.
- An arbitrary set of points and associated values (wrapped into a `Geospatial_data` object). The points need not coincide with triangle vertices or

centroids and a penalised least squares technique is employed to populate the quantity in a smooth and stable way. Since the least squares technique can be time consuming for large problems, `set_quantity` employs a caching technique which automatically decides whether to perform the computations or retrieve them from a cache. This will typically speed up the build by several orders of magnitude after each computation has been performed once.

- A filename containing points and attributes.
- A Numerical Python array (or a list of numbers) ordered according to the internal data structure.

The parameter `location` determines whether the values should be assigned to triangle edge, midpoints or vertices and `region` allows the operation to be restricted to a region specified by a symbolic tag or a set of indices.

Boundary conditions are bound to symbolic tags through the method `domain.set_boundary` which takes as input a lookup table (implemented as a Python dictionary) of the form `{tag: boundary_object}`. The boundary objects are all assumed to be callable functions of vectors `x` and `y`. Several predefined standard boundary objects are available and it is relatively straightforward to define problem-specific custom boundaries if needed. The predefined boundary conditions include Dirichlet, Reflective, Transmissive, Temporal, and Spatio-Temporal boundaries.

Forcing terms can be written according to a fixed protocol and added to the model using the idiom `domain.forcing_terms.append(F)` where `F` is assumed to be a user-defined callable object.

When the simulation is running, the length of each time step is determined from the maximal speeds encountered and the sizes of triangles in order not to violate the CFL condition which specifies that no information should skip any triangles in one time step. With large speeds and small triangles, time steps can become very small. In order to access the state of the simulation at regular time intervals, ANUGA uses the method `evolve`:

```
For t in domain.evolve(yieldstep, duration):  
    <model interrogation and modification>
```

The parameter `duration` specifies the time period over which `evolve` operates, and control is passed to the body of the for-loop at each fixed time step called `yieldstep`. The internal workings of the numerical scheme and its variable time stepping are thus decoupled from the fixed time stepping of the `evolve` loop. This means that the user of the API may access the model at fixed

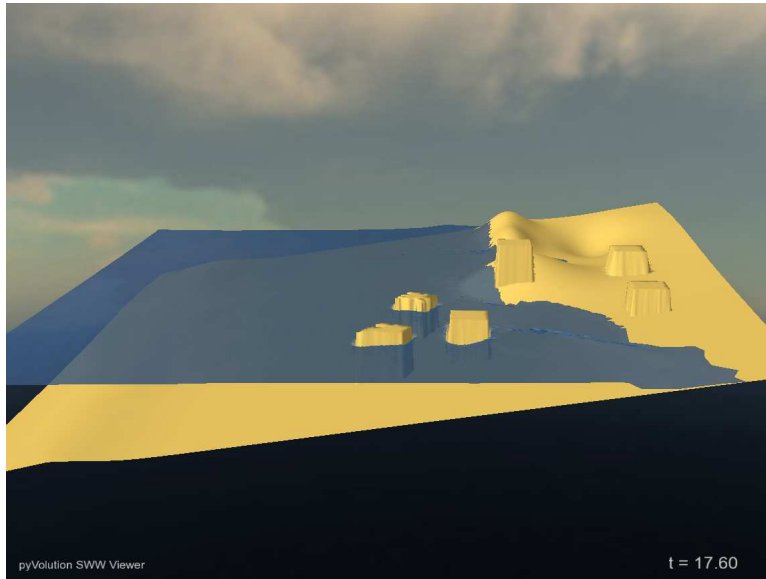


Figure 4: A hypothetical runup scenario.

timesteps to e.g. store model outputs, interrogate quantities or change the model itself at runtime. The evolve method has been implemented using a Python generator hence the reference to 'yield' in the parameter name.

Figure 4 shows a simulation of water flowing onto a hypothetical beach with obstacles. A number of complex patterns are captured in this example including a shock where water reflected off the wall far (at the right hand side) meets the main flow. Other physical features are the standing waves and interference patterns. See the ANUGA User Manual at <http://sourceforge.net/projects/anuga> for more details and examples.

5 Validation

The process of validating the ANUGA application is in its early stages, however initial indications are encouraging.

As part of the Third International Workshop on Long-wave Runup Models in 2004 (<http://www.cee.cornell.edu/longwave>), four benchmark problems were specified to allow the comparison of numerical, analytical and physical models with laboratory and field data. One of these problems describes a wave tank simulation of the 1993 Okushiri Island tsunami off Hokkaido, Japan [4]. A significant feature of this tsunami was a maximum run-up of 32 m observed at the head of the Monai Valley. This run-up was not uniform

along the coast and is thought to have resulted from a particular topographic effect. Among other features, simulations of the Hokkaido tsunami should capture this run-up phenomenon.

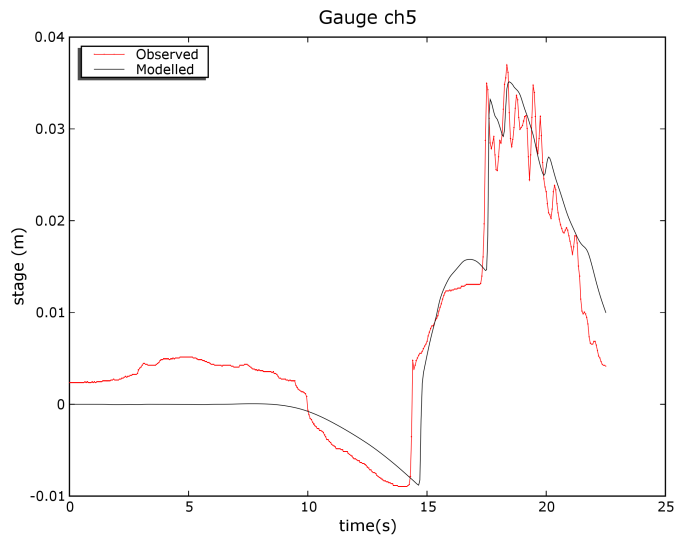


Figure 5: Comparison of wave tank and ANUGA water stages at gauge 5.

The wave tank simulation of the Hokkaido tsunami was used as the first scenario for validating ANUGA. The dataset provided bathymetry and topography along with initial water depth and the wave specifications. The dataset also contained water depth time series from three wave gauges situated offshore from the simulated inundation area.

Figure 5 compares the observed wave tank and modelled ANUGA water depth (stage height) at one of the gauges. The plots show good agreement between the two time series, with ANUGA closely modelling the initial draw down, the wave shoulder and the subsequent reflections. The discrepancy between modelled and simulated data in the first 10 seconds is due to the initial condition in the physical tank not being uniformly zero. Similarly good comparisons are evident with data from the other two gauges. Additionally, ANUGA replicates exceptionally well the 32 m Monai Valley run-up, and demonstrates its occurrence to be due to the interaction of the tsunami wave with two juxtaposed valleys above the coastline. The run-up is depicted in Figure 6.

This successful replication of the tsunami wave tank simulation on a complex 3D beach is a positive first step in validating the ANUGA modelling capability. Subsequent validation will be conducted as additional datasets



Figure 6: Complex reflection patterns and run-up into Monai Valley simulated by ANUGA and visualised using our netcdf OSG viewer.

become available.

6 Conclusions

ANUGA is a flexible and robust modelling system that simulates hydrodynamics by solving the shallow water wave equation in a triangular mesh. It can model the process of wetting and drying as water enters and leaves an area and is capable of capturing hydraulic shocks due to the ability of the finite-volume method to accommodate discontinuities in the solution. ANUGA can take as input bathymetric and topographic datasets and simulate the behaviour of riverine flooding, storm surge, tsunami or even dam breaks. Initial validation using wave tank data supports ANUGA's ability to model complex scenarios. Further validation will be pursued as additional datasets become available. ANUGA is already being used to model the behaviour of hydrodynamic natural hazards. This modelling capability is part of Geoscience Australia's ongoing research effort to model and understand the potential impact from natural hazards in order to reduce their impact on Australian communities (see [6]). The ANUGA source code is available at <http://sourceforge.net/projects/anuga>.

References

- [1] S. Baldwin. Our aussie tsunami terror. *Take 5*, 32:18–19, August 2006.
- [2] A. Kurganov, S. Noelle, and G. Petrova. Semidiscrete central-upwind schemes for hyperbolic conservation laws and hamilton-jacobi equations. *SIAM Journal of Scientific Computing*, 23(3):707–740, 2001.
- [3] J.F. Lebrun, G.D. Karner, and J.Y. Collot. Fracture zone subduction and reactivation across the puysegur ridge/trench system, southern new zealand. *Journal of Geophysical Research*, 103:7293–7313, 1998.
- [4] M. Matsuyama and H. Tanaka. An experimental study of the highest run-up height in the 1993 hokkaido nansei-oki earthquake tsunami. In *National Tsunami Hazard Mitigation Program Review and International Tsunami Symposium (ITS)*, pages 879–889. U.S. National Tsunami Hazard Mitigation Program, 2001.
- [5] O. Nielsen, S. Roberts, D. Gray, A. McPherson, and A. Hitchman. Hydrodynamic modelling of coastal inundation. In A. Zenger and R.M. Argent, editors, *MODSIM 2005 International Congress on Modelling and Simulation*, pages 518–523. Modelling and Simulation Society of Australia and New Zealand, December 2005. <http://www.mssanz.org.au/modsim05/papers/nielsen.pdf>.
- [6] O. Nielsen, J. Sexton, D. Gray, and N. Bartzis. Modelling answers tsunami questions. *AusGeo News*, 83, September 2006. <http://www.ga.gov.au/ausgeonews/ausgeonews200609/modelling.jsp>.
- [7] E. F. Toro. Riemann problems and the waf method for solving the two-dimensional shallow water equations. *Philosophical Transactions of the Royal Society, Series A*, 338:43–68, 1992.
- [8] Y. Tsuji, S. Matsutomi, F. Imamura, and C.E. Synolakis. Field survey of the east java earthquake and tsunami. *Pure and Applied Geophysics*, 144(3/4):839–855, 1995.
- [9] C. Zoppou and S. Roberts. Catastrophic Collapse of Water Supply Reservoirs in Urban Areas. *ASCE J. Hydraulic Engineering*, 125(7):686–695, 1999.

A limited review of the available Free / Libre and Open Source Academic Literature, as of late 2006.

Skidmore, Darren d.skidmore@unimelb.edu.au

Abstract

This paper is a compilation of the various academic references and a timeline of significant F/LOSS events. It is intended as a resource for people beginning in their research of F/LOSS or who are researching in F/LOSS to refer to in various sections to look at the research available. It is not claiming to be the ultimate guide, but does make a contribution to the research area as a repository of knowledge.

It should also be said upfront, that although the author has read widely in the Free / Libre¹ and Open Source Software (F/LOSS) literature, they have not read close to everything, and certainly have not been able to remember, record or re-find everything they have read, so if critical work is missing, the author apologises, and would be grateful for feedback. Also although there is a plethora of writings about F/LOSS, which include many web sites, and articles, this paper will try and concentrate on works which are academic in nature. The inclusion (or exclusion) of a paper from this paper should not be seen as either endorsement nor included, a rejection of the paper. This is a list of papers that the author has found or had pointed out, and for the most part, reflect the research interests focused on Information Systems research. This paper wishes to concentrate on references primarily to do with F/LOSS so references to classic software engineering e.g. The Mythical Man Month (Brooks, 1995) or discussions on the business of software e.g. (Cusumano, 2004) are not generally given; the emphasis being on papers and articles that are focused on F/LOSS specifically, this of course leaves out excellent supporting papers on issues on business value, software engineering and other issues.

The paper starts the beginnings of Open Source and some critical stages in its development, and some seminal starting places for research into F/LOSS. This is followed by a small series of repositories of either general information or a wider range of information, a list of publications where academic research into F/LOSS is also given. The paper then lists out a series of headings and the works under those topics that can be found on Open Source.

The beginning.

Although not the first usage or conceptualisation of what would become Free / Libre and Open Source, in 1983, Richard Stallman, often referred to as RMS, decided to create the GNU (GNU is Not Unix) project, shortly after he founded the Free Software Foundation (FSF). The GNU project was created with the plan to create a Free Operating System called GNU Operating System. In May of 1985 RMS released the GNU Manifesto, detailing his plans for GNU and asking for help and support. The philosophical ideas and aims of the Free Software Foundation can be found in the Free Software Definition (Free Software Foundation). This is the document which contains the four freedoms which form the basis of the FSF's philosophy. Richard Stallman also created the GNU General Public License (GNU GPL) (Free Software Foundation, 1991). The GNU GPL is the practical enforcement of the FSF philosophy; this is also concept referred to as CopyLeft which is the effective result of the GNU GPL. The concept of copyleft (Free Software Foundation, 2005c), which the GNU GPL enacts, requires that where a new software application which builds upon or uses part of a work (which is the term used in copyright law) that is licensed under the GNU GPL, then the new work must and can only be licensed under the GNU GPL (even if the section taken is only a tiny piece). This result of using the GNU GPL is also why some critics refer to the GNU GPL as being viral, a reciprocal license (Rosen, 2004), or also as a propagating license (New Zealand State Services Commission, 2006).. The expression "Free as in Speech, not as in Beer", was also coined by the FSF in the Free Software Definition, remembering this is a US centric view, where the guarantee of Free Speech is the first amendment to their constitution, and has therefore a special meaning and weight to citizens of the US.

Although the FSF, the GNU and the GNU GPL have a high profile as the initiators of what is now referred to as Open Source (discussed shortly), independent of, and previous to, the FSF, AT&T which owned UNIX at the time, fought against some users of Unix, particularly several

¹ Libre is the French word for Free (as in Freedom) which is differentiated from the French word meaning no monetary cost.

Universities, in a long and complicated battle to be able to use and share the source code to UNIX. This led to the other major movement in Open Source around the same time as the formation of GNU, being that of the Berkeley Software Distribution (BSD), where members of the University released the source code to the 'UNIX like' Operating System, which they had developed. The license under which this was done is referred to as the BSD license, the main condition which was that the authors of any code used must be acknowledged in the new code. The precursor to the BSD style license, which was the license for the [BSD] Networking Release 1, was created in June of 1989² (Weber, 2004, Pg 40), however Berkeley Software Distributions' had been put together since 1977 (Wayner, 2000, pg 94) and distributed to others in 1978 (Weber, 2004, pg 31).

The two licenses contrast each other, and this has led to some heated exchanges between the two sides, both are Open Source, but the GNU GPL, requires that the derivative code and applications remain Open Source, whereas this is not required by BSD style licenses, which see themselves as being completely free. Apple's OS X is an example of the outcome of the BSD style licenses, Apple developed OS X taking code from the Mach 3 microkernel, and FreeBSD 3.2 (Weber, 2004, pg 201), although Apple have released the core code for OS X, called Darwin, under a BSD style license, they have kept OS X closed source, not contributing all of the improvements to the original code.

In May of 1997 Eric Raymond wrote an essay titled the Cathedral and the Bazaar, which was later, compiled with a series of other essays in a book (Raymond, 2001)³. In the series of essays, Raymond has listed out the historical development of Open Source, what he sees as the advantages, and argues the benefits of hackers and Open Source. The Cathedral and the Bazaar, is credited with being influential as a contributor to the starting point for business acceptance of Open Source. The essay also lists out and describes nineteen "lessons" which have been regularly quoted. One of these, Raymond dubbed "Linus's Law" being "Given enough eyeballs all bugs are shallow" (Raymond, 2001, p 30: Lesson number 8). The name of the book and essay refers to perceived development style; that of the Cathedral, all planned and constructed to the plan, compared to a Bazaar where everything is created by individuals. Netscape in response to competitive pressures made an announcement in January of 1997, that they would release the source code to the Netscape Browser, which eventually led to the Mozilla browsers. This is also seen as the first large vendor announcement of a move to allow access to the source code. Partially in response to the Netscape announcement, in February of 1998 several people⁴ met and coined the term "Open Source", (Open Source Initiative, 2005). This meeting also resulted in the formation of the organisation the Open Source Initiative (OSI), and the writing of the Open Source Definition (Open Source Initiative, 2004a), which was based upon the Debian Free Software Guidelines, written by Bruce Perens. Previous to this meeting the term Open Source was not used, instead the concept was referred to as Free Software. The OSI could not trademark the term Open Source, and instead in addition to being an advocate organisation, will certify that a License conforms to the Open Source Definition. Part of the reason to use the term Open Source was that the term Free Software and the advocacy of the FSF, which insisted upon a strict GNU view of the world, which was seen as hindering the adoption of F/LOSS by businesses. To a larger extent the FSF, has a philosophy, of which the GNU GPL is the practical implementation, whereas the OSI takes a more pragmatic, and engineering view of Open Source. These differences have led to several heated exchanges in the past, and although both camps see Open Source as the best option for software development, they disagree about the licensing methods by which this is enforced and the ultimate societal aims surrounding F/LOSS.

In April of 1991, Linus Torvalds posted a note to the usenet list comp.os.minix which announced the first public call to participate to help Linus develop the Operating System Linux. Although Linux is a major development both in terms of the impact of Open Source on organisations and the development of Open Source, alternative Operating Systems, such as the xBSD's (e.g. FreeBSD, OpenBSD, and NetBSD) have also been F/LOSS successes. The growth and development in the Internet at the same time as the F/LOSS movement was also possibly a large factor in its success. Certainly, the killer app of F/LOSS (which gave Linux and the xBSD

² the GNU GPL v 1 was possibly created a few months before this in 1989 as the EMACS General Public License González, A. G. (2006) GPL v3: A Legal Analysis. IN Özel, B., Çilingir, C. B. & Erkan, K. (Eds.) *Towards Open Source Software Adoption: Educational, Public, Legal, and Usability Practices. OSS 2006 tOSSad workshop proceedings*. Como, Italy, TÜBİTAK (The Scientific & Technology Research Council of Turkey). http://www.tossad.org/tossad/events/tossad_2006/proceedings.

³ The essays are also available at the web site: <http://www.catb.org/~esr/writings/cathedral-bazaar/>

⁴ including Todd Anderson, Chris Peterson, John "maddog" Hall, Larry Augustin, Sam Ockman and Eric Raymond.

Operating Systems a reason to be installed) has been described as the Apache Web Server (Moore et al., 2003), in 1995 several Systems Administrators made contact with each other, to maintain and then build on the National Center for Supercomputing Applications HTTP daemon (or the NCSA HTTPd) Web Server which had fallen into disarray after the creator left the project. The first public release of Apache (HTTP Server 0.6.2) was in April of 1995, with HTTP Server 1.0 released in December of 1995.

An alternate book on the history of Open Source movement is (Wayner, 2000)⁵, which also discusses the personalities behind Linux and the Open Source movement, including Richard Stallman, and Eric Raymond, as well as several others. The book gives an account of the major milestones, schisms and forks in some of the more important Open Source applications and distributions. (Weber, 2004), has an account of the history of Open Source, including a description of the UNIX development, in the initial chapters of the book. A DVD documentary has been made of the history of Linux (Moore et al., 2003), "Revolution OS", which interviews many people in the F/LOSS domain, also documenting the story of Open Source Software, and Linux.

Collections

The collections listed here are repositories which have published information on F/LOSS in several areas, some sites such as the OSI have not been listed as they will be listed in other sections, i.e. in the case of the OSI then Licensing, in the case of SourceForge Engineering and Implementations and Case Studies. Another review of the research literature on F/LOSS is (Overby, 2003), as is First Mondays special issue on Open Source (Krishnamurthy, 2005b). Some works have not been included because either other sources were found, or they have not been read, such as the collection of selected essays of Richard Stallman (Gay, 2002), or on business and economics such as Martin Finks book (Fink, 2003).

Web sites

There are more than several web sites which are useful in research in Open Source, however to list a few in brief, many of the sites also have mailing lists which provide ongoing information on Open Source.

Free Software Foundation	http://www.fsf.org/
MIT Free / Open Source Research Community	http://opensource.mit.edu/
EU IDABC Open Source Observatory	http://europa.eu.int/idabc/oso
Center of Open Source and Government	http://www.egovos.org/
ASK-OSS (Australia)	http://ask-oss.mq.edu.au

Books

An collection of articles on Open Source issues is (Feller et al., 2005), other books which have a general overview of F/LOSS, but generally have a concentration on specific issues are (DiBona et al., 2006, DiBona et al., 1999, Golden, 2004, Raymond, 2001, Wayner, 2000, Weber, 2004). With a reflection on the comments of Raymond in the Cathedral and the Bazaar (Bezroukov, 1999).

Conferences / Journals

Some Academic conferences which are aimed at and have dedicated tracks in Open Source Software are the International Conference on Open Source Systems, and the European Conference of Information Systems. Journals which have published papers on Open Source are also listed, although they are not dedicated to F/LOSS.

First International Conference on Open Source Systems	http://oss2005.case.unibz.it/
Second International Conference on Open Source Systems	
HICCS	
AMCIS	
European Conference on Information Systems	http://is.lse.ac.uk/asp/aspecis/

First Monday	http://firstmonday.org/index.html
First Monday Special Issue on Open Source 2005 October 3	http://firstmonday.org/issues/special10_10/

⁵ A copy of this book is available under a Creative Commons license at <http://www.wayner.org/books/ffa/>

Software, IEEE	
Information & Management	
Organization Science	
Journal of Industrial Economics	
Research Policy	
Information System Journal	

FUD

The definition of FUD is Fear, Uncertainty and Doubt (The on-line hacker Jargon File, 2003), the meaning of which is to create the FUD to prevent persons or organisations from swapping to a rival system. Infamous in F/LOSS is the Halloween Document, October 1998⁶.

SCO

In March of 2003, SCO started legal proceedings against IBM, alleging a raft of infringements to do with the transfer of source code from UNIX to LINUX, in an ongoing and dragged out pretrial proceedings. This has also had a limited effect on the arguments in the adoption of F/LOSS by companies. The OSI has a history of the events⁷, there are also other works on this case (Omar, 2005, Moglen, 2003).

Legal Issues

The foundation of F/LOSS, in practice is the use of licenses which authorise the use of and access to the source code. Therefore many works have been dedicated or comment on licensing issues. In addition there are issues in Patent law given the effect and increasing use of Patents in software. In 2003 a conference on the topic of Open Source legal issues discussed these topics (Fitzgerald, 2003), there was also a roundtable organised at the University of Washington, in 2006⁸.

Licensing

Although the Open Source Initiative and the FSF take a leadership and coaching role in F/LOSS, in all of the discussion about F/LOSS, it should be remembered that the term Open Source is not a Trade Mark nor a protected name, therefore there are licenses which say they are Open Source, but which are not compatible with the Open Source Definition, and / or which the FSF has a serious disagreement. The Hacktivismo HESSLA license for instance states that source code covered under the license may not be used to violate human rights (Hacktivismo, Free Software Foundation, 2005a), Microsoft in their shared source program, state that the resulting application must only be used on Windows platform (Microsoft, 2005). There are several resources to go to for lists of various Open Source Software Licences (Open Source Initiative, 2004b, Free Software Foundation, 2005b, ifrOSS, 2005, Commonwealth of Massachusetts, 2004).

Since the source code is available in F/LOSS, there is the possibility of companies using the source code in proprietary programs and not returning this code to the community (Ciffolilli, 2004). One organisation which is dedicated to finding and then bringing such practices to task is the GPL-Violations.org⁹, which claims success in forcing organisations to share or remove infringing code. A Munich district court in Germany, in April of 2004 has upheld the GNU GPL under German law¹⁰.

Works that specifically deal with issues in Open Source Licences are books by Rosen¹¹ (Rosen, 2004), and St. Laurent (St. Laurent, 2004) as well as several papers (Lerner and Tirole, 2005b, Lin et al., 2006, McGowan, 2005, Skidmore, 2007, Välimäki, 2005, Vetter, 2004, Webbink, 2003, AGIMO, 2005). Discussion on the choices available to an organisation (Barnett and Titterington, 2003), on the use of Dual Licenses (Olson, 2006, Välimäki, 2003) and a firms' choices of Open Source Licenses (Bonaccorsi and Rossi, 2003). Numbers and types of licenses used in Open Source projects can be found (Weiss, 2005, Rusovan et al., 2005). Comment on the GNU GPL version 3 (Free Software Foundation, 2006), González (2006) and. The issue of license proliferation also has comment (Rosen, 2005, Skidmore, 2006). In an analysis of the licenses used (Weiss, 2005) found that the GNU GPL was used for 45% of the projects with 7%

⁶ <http://www.catb.org/~esr/halloween/>

⁷ <http://www.opensource.org/sco-vs-ibm.html> visited 2006 January 16.

⁸ Beyond the Basics: Advanced Legal Topics in Open Source and Collaborative Development in the Global Marketplace, available at <http://www.law.washington.edu/lct/Events/FOSS/>

⁹ <http://www.gpl-violations.org/>

¹⁰ http://www.jbb.de/urteil_lg_muenchen_gpl.pdf (Unofficial English translation): http://www.jbb.de/judgment_dc_munich_gpl.pdf

¹¹ Available under a Creative Commons License at: <http://www.rosenlaw.com/oslbook.htm>

using the GNU LGPL, 5% BSD, with 22 other licenses sharing the remaining 43%, this is partly explained by the GPL forcing obligations of people as well as the nature of the GNU GPL. Also see the section of Researching into Open Source for the FLOSSmole project.

Patent

The issue of patents, is becoming an issue in general in Software Engineering, but has been prominent in the F/LOSS literature and as a outgrowth of the legal landscape. The Open Source Risk Management (OSRM) group analysis of the patent risk to Linux (Ravicher, 2004) finding there were issues and discussed options around the issue. Options to mitigate risk are available, from the OSRM¹², several companies have combined to contribute their patents for use by F/LOSS projects as well several companies have released policies to use their patent portfolios to protect against patent attack (IBM, 2005b, IBM, 2005a, HP News Release, 2004, Novell, 2004b, Novell, 2004a). There is also an initiatives to permit Patents for use in Open Source Projects as well as assisting in dealing with issues in Patents (Open Invention Network, 2005).

Engineering

An overview of the current state of F/LOSS engineering has been compiled (Koch and Gonzalez-Barahona, 2005). Decisions about which licenses to use are required, for this there is a perspective of different stakeholders and the choice of F/LOSS license to their needs (Michaelson, 2004). The use of F/LOSS licenses also has an effect on the competition, and may effect choices available to others (Välimäki and Oksanen, 2005). Defining the success of an F/LOSS project is given in (Crowston et al., 2003). Issues in the creation of an Open Source project, is discussed in (Fogel, 2005). Mixing of proprietary and open source (West, 2003). For an article which analyses and discuss the components, both in lines of code, modules and license types in Red Hat Linux 7.1, (Wheeler, 2002). Discussion of a module in Linux and the quality (or lack of it) can be found in (Rusovan et al., 2005).

Security and the robustness of F/LOSS code verses Closed Source code works are (Anderson, 2005). Use of F/LOSS in Software Engineering or programming teams, (Lussier, 2004, Madanmohan and De', 2004, Bonaccorsi and Rossi, 2005), looking specially in ASIA and Japan (Shimizu et al., 2004), and in Italy (Bonaccorsi and Rossi, 2003, Bonaccorsi and Rossi, 2004), as well as developing countries (Kshetri, 2004). Other papers dealing with development and engineering are (Serrano et al., 2004, Scacchi, 2004, Samoladas et al., 2004, Rusovan et al., 2005, Ruffin and Ebert, 2004, Robbins, 2005, Spinellis and Szyperski, 2004, Neumann, 2005, German, 2005, Mockus et al., 2005, Jørgensen, 2005). Considerations of non code artefacts (Robles et al., 2006). Internationalisation issues have also been discussed (Ghosh, 2003, Souphavanh and Karoonboonyanan, 2005). (Moon and Sproull, 2000), looks at the development of the Linux Kernel from three different perspectives, that of the individual, the group and the community. The divisions of labour (Torrise et al., 2005).

Codebase repositories of F/LOSS

Below are a list of repositories for F/LOSS projects or software, of them the largest is SourceForge, which owned by the Open Source Technology Group (OSTG). SourceForge is also linked to or has influenced several other repositories (e.g. Government Forge, eclipse.techforge.com). SourceForge is a major F/LOSS development repository for the codebases of F/LOSS.

¹² URI [OSRM]: <http://www.osriskmanagement.com/> visited 2005 January 14th.

Repository	URI	Comment
SourceForge	http://sourceforge.net/	Largest source of F/LOSS
Savannah	http://savannah.gnu.org/	GNU only Repository
Free Software Directory	http://directory.fsf.org/	A directory of Free Software
FreshMeat	http://freshmeat.net/	
Government Forge	http://governmentforge.org/	Repository of software aimed at Governments.
EU's IDABC Software taxonomy	http://europa.eu.int:80/idabc/en/document/t/3499	A list of F/LOSS applications by business function.

Usability

There has been limited research in the area of Usability of Open Source Software the best reference is (Nichols and Twidale, 2003). However there is an initiative to assist projects with increasing the usability of Open Source called OpenUsability¹³

Reward / Motivations

Consideration of why individuals or organisations choose F/LOSS (Bonaccorsi and Rossi, 2003, Rossi and Bonaccorsi, 2005). In a much quoted survey paper, the Boston Consulting group conducted research to look at the motivations of Open Source Developers (Lakhani et al., 2002).

Business / Organisational

Implementations and Case Studies

The EU IDABC site has information and case studies on F/LOSS in Member countries¹⁴, as well as expert studies conducted for the EU¹⁵. Papers on F/LOSS implementation (Fitzgerald and Kenny, 2004, Fitzgerald and Kenny, 2003).

Government use

Open Source issues in Government, (Gartner, 2004, Center for Strategic and International Studies (CSIS), 2004, Fitzgerald and Suzor, 2005, Hahn et al., 2002, Office of Government Commerce, 2004, Government Information Officers' Council (GITOC), 2003, AGIMO, 2004, AGIMO, 2005, Schmitz, 2001, Berlecon Research GbmH, 2002b, Aigrain, 2005, Wong, 2004). The EU proposes a license build for the EU, not only for government use, but also by organisations in Europe, license (CeCILL, 2005a, CeCILL, 2005b), discussion paper (Dusollier et al., 2004), a newer version of the license (version 0.2) has also been published (European Community, 2005). NICTA in Australia also created a license for that jurisdiction (National ICT Australia, 2004).

Support

The support of software is seen as a critical business requirement, articles on the support that users can get are (Lakhani and von Hippel, 2003).

Considering Open Source

Research into the barriers and constraints considered by organisations (McCabe, 2004, Goode, 2005). Books which build business cases and consideration of what an organisation should use or consider (Woods and Guliani, 2005, Golden, 2004, Lahti and Peterson, 2005). When deciding to move to F/LOSS organisations should require a feasibility study, such as the EU IDA study (Schmitz and Castiaux, 2002, Berlecon Research GbmH, 2002a), as well considering risk assessment (Drozdik and Kovács, 2005). Alternatives as well as inclusive factors in the decision to adopt might be that of COTS and architecture considerations (Holck et al., 2005). Decisions may also need to be made which licenses to use (Bonaccorsi and Rossi, 2003, Bonaccorsi and Rossi, 2004). An initiative to create a rating of the Open Source software, and how mature or ready for business adoption has been created with the Business Readiness Rating (BRR)¹⁶.

¹³ <http://www.openbrr.org/wiki/index.php/Home>

¹⁴ <http://europa.eu.int/idabc/en/document/1666/471>

¹⁵ <http://europa.eu.int/idabc/en/document/3879/471>

¹⁶ BRR: <http://www.openbrr.org/>

Business Models

Business models in Open Source are looked at by sources such as (Krishnamurthy, 2005a). An example and building of a possibly model, using JBOSS as an example is given in (Watson et al., 2005).

Specific Business Issues

A book looks at Open Source tools that companies can use to comply with the Sarbanes-Oxley act (Lahti and Peterson, 2005).

F/LOSS as an Innovation method

Some papers which look to see if F/LOSS can assist in innovation for firms (von Hippel, 2005, von Hippel and von Krogh, 2003).

Economics

Sources which concentrate on the economics of F/LOSS are (Lerner and Tirole, 2002, Lerner and Tirole, 2005b, Lerner and Tirole, 2005a, Benkler, 2002, Frost et al., 2005, Iannacci, 2002), of these the seminal are possibly Benkler (2002) and Lerner (2002). A paper which looks at economic externalities and costs of F/LOSS use is (Edwards, 2005) analysing three license types, that of the GNU GPL, BSD and Microsoft EULA.. Looking at how F/LOSS can reduce transaction costs (Soares, 2004). (Schmidt and Schnitzer, 2003), looks at the economic merits of direct and indirect public subsidies for Open Source projects, as well as in terms of Gift economic use (Zeitlyn, 2003). A comparison with the historical perspectives of the coal mine steam engines is (Nuvolari, 2005).

Wider useage of F/LOSS techniques

Comment on the wider usage of F/LOSS techniques (Shirky, 2005). Taking Open Source techniques and applying them to other organisational development (Ljungberg, 2000)

Future of F/LOSS

Some articles which discuss the future of F/LOSS, (Appelbe, 2003), in particular the issues which must be addressed or may confront F/LOSS in the coming years (Fitzgerald, 2005). A paper in MIS quarterly has discussed the changes that have occurred in the Open Source domain distinguishing what it now refers to as OSS 2.0 from FOSS (Fitzgerald, 2006).

Researching into Open Source

Researching into Open Source Software, particularly in the aspects of licenses and analysis of usage has in most cases depended on the use of SourceForge data, papers on this are depending on the analysis that needs to be made (Howison et al., 2005). Data on SourceForge for academic research is available under certain conditions from the University of Notre Dame¹⁷, in the Research Project on the Free/Open Source Software Development Phenomenon. There is also an initiative on the collection of F/LOSS project data information with the FLOSSmole¹⁸ (Conklin et al., 2005).

TimeLine

Below is a brief timeline of events in the Open Source development. Some of the events are extremely hard to track and therefore it is difficult to get this information completely accurately.

Date	Event	Description
1969/04/07	RFC 1	Request For Comment 01, "Host Software"
1969/06/01	UNICS (precursor to UNIX)	UNICS was later renamed UNIX
1977/01/01	Berkeley Software Distribution put together	Bill Joy first put the BSD together
1979/01/01	Printer Jam	Sometime in 1979 a Printer Jams and Xerox will not release the source code so that they can play with it, This upsets RMS
1983/09/27	Initial Announcement of the GNU Project	Richard Stallman sent initial email announcing the GNU idea
1985/03/01	GNU Manifesto	Richard Stallman wrote the GNU Manifesto
1989/01/01	GNU GPL v1	

¹⁷ <http://www.nd.edu/~oss/Data/data.html>
<http://www.nd.edu/~oss/index.html>

¹⁸ <http://ossmole.sourceforge.net/>

1989/06/01	[BSD] Network Release 1 (BSD style License)	Berkeley group releases Networking Release 1
1991/04/25	Linux	Initial Post to the comp.os.minix newsgroup for first Linux contributions
1991/06/01	GNU GPL v2	
1991/09/17	Linux 0.01	Linux Kernel 0.01 released to the Internet
1991/12/01	Linux 0.02	
1992/03/01	Linux 0.95	First Linux Kernel capable of running X-Windows
1994/03/14	Linux 1.0.0	
1994/06/01	4.4BSD-Lite	
1995/01/01	Apache	Apache Initial Formation.
1995/04/01	Apache HTTP Server 0.6.2	First public Release of Apache Server
1995/08/01	Apache HTTP Server 0.8.8	
1995/12/01	Apache HTTP Server 1.0	
1996/06/09	Linux 2.0.0	
1997/05/18	The Cathedral and the Bazaar	Eric Raymond released first version of paper "The Cathedral and the Bazaar"
1997/06/06	Apache HTTP Server 1.2	
1998/01/22	Netscape Announcement	Announcement by Netscape that they will release the source code for Netscape Navigator Browser
1998/02/03	Open Source Initiative Formed	
1998/03/31	Navigator Source Code Released	Netscape Navigators Source is released under License Netscape Navigator
1998/06/06	Apache HTTP Server 1.3.0	
1998/10/01	Halloween Documents	
1999/02/01	Lesser General Public License v2.1	
2000/01/01	Apache License v1.1	
2000/01/01	SourceForge	Source Forge was running in Dec of 1999, but announced in Jan of 2000
2000/04/04	US Court rules Source Code is Speech	
2000/06/01	MySQL goes to Dual License	
2000/07/01	Sun Announces Star Office to be Open Source	
2001/01/04	Linux 2.4.0	
2002/04/06	Apache HTTP Server 2.0	
2003/03/06	SCO	SCO files suit in 3rd District Court of UTAH against IBM
2003/12/17	Linux 2.6.0	
2004/01/01	Apache License v2.0	
2005/10/18	Apache HTTP Server 1.3.34	
2005/12/01	Apache HTTP Server 2.2.0	

Bibliography

- Agimo (2004) Guide to ICT Sourcing for Australian Government Agencies. IN Department of Finance and Administration (Ed.) Canberra, Australian Government Information Management Office,, <http://www.sourceit.gov.au/sourceit/ict>
- Agimo (2005) Guide to Open Source Software for Australian Government Agencies. IN Department of Finance and Administration (Ed.) Canberra, Australian Government Information Management Office,, <http://www.sourceit.gov.au/sourceit/oss>
- Aigrain, P. (2005) Libre Software Policies at the European Level. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Anderson, R. (2005) Open and Closed Systems Are Equivalent (That Is, in an Ideal World). IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.)

Perspectives on Free and Open Source Software. Cambridge, Massachusetts, The MIT Press.

- Appelbe, B. (2003) The Future of Open Source Software. *Journal of Research and Practice in Information Technology*, 35, 227-236.
- Barnett, G. & Titterington, G. (2003) Software Licensing and open source: the enterprise in the maze. Ovem Research. <http://www2.ovum.com/secure/p,,34671>
- Benkler, Y. (2002) Coase's Penguin, or, Linux and the Nature of the Firm. *The Yale Law Journal*, 112, 369-446.
- Berlecon Research Gbmh (2002a) FLOSS Final Report, Free/Libre and Open Source Software: Survey and Study. IN Berlecon Research Gbmh (Ed.) *Free/Libre and Open Source Software: Survey and Study*. Berlin, Germany, Berlecon Research GbmH, University of Maastricht, The Netherlands.
<http://www.infonomics.nl/FLOSS/report/>
- Berlecon Research Gbmh (2002b) FLOSS Final Report, Free/Libre and Open Source Software: Survey and Study: Part 0 Table of Contents and Executive Summary. IN Berlecon Research Gbmh (Ed.) *Free/Libre and Open Source Software: Survey and Study*. Berlin, Germany, Berlecon Research GbmH,.
<http://www.infonomics.nl/FLOSS/report/Final0.htm>
- Bezroukov, N. (1999) A Second Look at the Cathedral and Bazaar. *First Monday*, 4.
- Bonaccorsi, A. & Rossi, C. (2003) Licensing schemes in the production and distribution of Open Source software. An empirical investigation. Institute for Informatics and Telematics. <http://opensource.mit.edu/papers/bnaccorsirossilicense.pdf>
- Bonaccorsi, A. & Rossi, C. (2004) Altruistic individuals, selfish firms? The structure of motivation in Open Source software. *First Monday*, 9.
- Bonaccorsi, A. & Rossi, C. (2005) Open Source Software, intrinsic motivations and profit-oriented firms. Do firms practice what they preach? IN Scotto, M. & Succi, G. (Eds.) *The First International Conference on Open Source Systems*. Genova, Italy, ECIG.
- Brooks, F. P. (1995) *The mythical man-month : essays on software engineering*, Reading, Mass., Addison-Wesley Pub. Co.
- Cecill (2005a) CeCILL Free Software License Agreement version 2.0 - English. CeCILL. http://www.cecill.info/licences/Licence_CeCILL_V2-en.html
- Cecill (2005b) Contrat de licence de logiciel libre CeCILL version 2.0. CeCILL. http://www.cecill.info/licences/Licence_CeCILL_V2-fr.html
- Center for Strategic and International Studies (Csis) (2004) Government Open Source Policies. IN Center for Strategic and International Studies (Csis) (Ed.) Washington DC, Center for Strategic and International Studies (CSIS).
<http://www.csis.org/tech/OpenSource/>
- Ciffolilli, A. (2004) The economics of open source hijacking and the declining quality of digital information resources: A case for copyleft. *First Monday*, 9.
- Commonwealth of Massachusetts (2004) Open Source Licenses - Quick Reference Chart. IN <Http://Www.Mass.Gov/ItD/Legal/Quickrefchart.Xls> (Ed.), Commonwealth of Massachusetts. Open Source Licenses - Quick Reference:
- Conklin, M., Howison, J. & Crowston, K. (2005) Collaboration using OSSmole: a repository of FLOSS data and analyses. *Proceedings of the 2005 international workshop on Mining software repositories*. St. Louis, Missouri, ACM Press.
<http://doi.acm.org.ezproxy.lib.unimelb.edu.au/10.1145/1083142.1083164>
- Crowston, K., Annabi, H. & Howison, J. (2003) Defining Open Source Software project success. *Proceedings of the International Conference on Information Systems (ICIS 2003)*. Seattle, WA, USA.
<http://dissertation.martinaspeli.net/papers/success/crowston-et-al-2003-defining-open-source-software-project-success/crowston-defining-success.pdf>

- Cusumano, M. A. (2004) *The business of software: what every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad*, New York, Free Press.
- Dibona, C., Cooper, D. & Stone, M. (2006) *Open Sources 2.0 : The Continuing Evolution*, Sebastopol, Calif., O'Reilly.
- Dibona, C., Ockman, S. & Stone, M. (1999) *Open sources : Voices from the Open Source Revolution*, Beijing ; Sebastopol, CA, O'Reilly.
- Drozdzik, S. & Kovács, G. L. (2005) Risk Assessment of an Open Source Migration Project. IN Scotto, M. & Succi, G. (Eds.) *The First International Conference on Open Source Systems*. Genova, Italy, ECIG.
- Dusollier, S., Laurent, P. & Schmitz, P-E. (2004) Open Source Licensing of software developed by the European Commission. IN Unisys (Ed.) *IDA/GPOSS Encouraging Good Practice in the use of Open Source Software in Public Administrations*. European Commission.
<http://europa.eu.int/idabc/servlets/Doc?id=19296>
- Edwards, K. (2005) An economic perspective on software licenses—open source, maintainers and user-developers. *Telematics and Informatics*, 22, 97-110.
- European Community (2005) Draft-European Union Public Licence. 0.2 ed., European Commission. PDF: <http://ec.europa.eu/idabc/servlets/Doc?id=24720>
- Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (2005) *Perspectives on Free and Open Source Software*, Cambridge, Massachusetts, The MIT Press.
- Fink, M. (2003) *The Business and Economics of Linux and Open Source*, Upper Saddle River, Prentice Hall.
- Fitzgerald, B. (2003) *Legal Issues Relating to Free and Open Source Software*, Brisbane, Queensland, Australia, Queensland University of Technology School of Law.
- Fitzgerald, B. (2005) Has Open Source Software a Future? IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Fitzgerald, B. (2006) The Transformation of Open Source Software. *MIS Quarterly*, 30, 587-598.
- Fitzgerald, B. & Kenny, T. (2003) Open Source Software in the Trenches: Lessons from a Large-Scale OSS implementation. *Twenty-Fourth International Conference on Information Systems*. Seattle Washington, USA.
- Fitzgerald, B. & Kenny, T. (2004) Developing an information systems infrastructure with open source software. *Software, IEEE*, 21, 50-55.
- Fitzgerald, B. & Suzor, N. (2005) Legal Issues for the Use of Free and Open Source Software in Government. *Melbourne University Law Review*, 29, 412-447.
- Fogel, K. (2005) *Producing open source software : how to run a successful free software project*, Beijing ; Sebastopol, CA, O'Reilly.
- Free Software Foundation The Free Software Definition.
<http://www.gnu.org/philosophy/free-sw.html>
- Free Software Foundation (1991) GNU General Public License Version 2.
<http://www.gnu.org/copyleft/gpl.html#SEC1>
- Free Software Foundation (2005a) The HESSLA's Problems. IN Free Software Foundation (Ed.) *Licenses*. Free Software Foundation.
<http://www.gnu.org/licenses/hessla.html>
- Free Software Foundation (2005b) Licenses. IN Free Software Foundation (Ed.) *Licenses*. Free Software Foundation. <http://www.fsf.org/licensing/licenses/>
- Free Software Foundation (2005c) What is Copyleft? IN Free Software Foundation (Ed.), Free Software Foundation,. <https://www.fsf.org/licensing/essays/copyleft.html>
- Free Software Foundation (2006) GPLv3 First Discussion Draft Rationale. Free Software Foundation. <http://gplv3.fsf.org/rationale>

- Frost, J. J., Leffler, P. K., Gomulkiewicz, P. R. & Laster, P. D. (2005) Some Economic & Legal Aspects of Open Source Software. <http://opensource.mit.edu/papers/frost.pdf>
- Gartner (2004) Where Government should use Open-Source Software. IN Harris, R. (Ed.) *Decision Framework*. Gartner. <http://lib-resources.unimelb.edu.au/mate.lib.unimelb.edu.au/gartner/research/119400/119491/119491.html>
- Gay, J. (Ed.) (2002) *Free Software, Free Society: Selected Essays of Richard M. Stallman*, Free Software Foundation.
- German, D. M. (2005) Software Engineering Practices in the GNOME Project. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Ghosh, R. A. (2003) Licence Fees and GDP per capita: The case for open source in developing countries. *First Monday*, 8.
- Golden, B. (2004) *Succeeding with open source*, Boston, Addison-Wesley.
- González, A. G. (2006) GPL v3: A Legal Analysis. IN Özel, B., Çilingir, C. B. & Erkan, K. (Eds.) *Towards Open Source Software Adoption: Educational, Public, Legal, and Usability Practices. OSS 2006 tOSSad workshop proceedings*. Como, Italy, TÜBİTAK (The Scientific & Technology Research Council of Turkey).
- Goode, S. (2005) Something for nothing: management rejection of open source software in Australia's top firms. *Information & Management*, 42, 669-681.
- Government Information Officers' Council (Gitoc) (2003) Using Open Source Software in the South African Government. Government Information Officers' Council (GITOC),. http://www.oss.gov.za/docs/OSS_Strategy_v3.pdf
- Hacktivismo The Hacktivismo Enhanced-Source Software License Agreement. Hacktivismo. <http://www.hacktivismo.com/about/hessla.php>
- Hahn, R. W., Bessen, J., Evans, D. S., Lessig, L. & Smith, B. L. (Eds.) (2002) *Government policy toward open source software*, Washington, D.C, AEI-Brookings Joint Center for Regulatory Studies.
- Holck, J., Pedersen, M. K. & Larsen, M. H. (2005) Open Source Software Acquisition: Beyond the Business Case. IN Bartmann, D., Rajola, F., Kallinikos, J., Avison, D., Winter, R., Ein-Dor, P., Becker, J., Bodendorf, F. & Weinhardt, C. (Eds.) *Thirteenth European Conference on Information Systems*. Regensburg, Germany. <http://is.lse.ac.uk/asp/aspecis/20050130.pdf>
- Howison, J., Conklin, M. & Crowston, K. (2005) OSSmole: A collection repository for FLOSS research data and analyses. IN Scotto, M. & Succi, G. (Eds.) *The First International Conference on Open Source Systems*. Genova, Italy, ECIG.
- HP News Release (2004) HP First Major Vendor to Certify, Integrate and Support Open Source Software for Customers. HP. <http://www.hp.com/hpinfo/newsroom/press/2004/040601a.html>
- Iannacci, F. (2002) The Economics of Open-Source Networks. *Communications & Strategies*, 48, 119-138.
- IBM (2005a) IBM Statement of Non-Assertion of Named Patents Against OSS. IBM. <http://www.ibm.com/ibm/licensing/patents/pledgedpatents.pdf>
- IBM (2005b) New IBM Initiative Advances Open Software Standards In Healthcare and Education. <http://www-1.ibm.com/press/PressServletForm.wss?MenuChoice=pressreleases&TemplateName=ShowPressReleaseTemplate&SelectString=t1.docunid=7938&TableName=DataheadApplicationClass&SESSIONKEY=any&WindowTitle=Press+Release&STATUS=publish>
- Ifross (2005) License Center. IN Software, I. F. R. D. F. U. O. S. (Ed.), *Institute für rechtsfragen der freien und Open Source Software*. http://www.ifross.de/ifross_html/lizenzcenter-en.html
- Jørgensen, N. (2005) Incremental and Decentralized Integration in FreeBSD. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.

- Koch, S. & Gonzalez-Barahona, J. M. (2005) Open Source Software engineering – The state of research. *First Monday*, Special Issue.
- Krishnamurthy, S. (2005a) An Analysis of Open Source Business Models. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Krishnamurthy, S. (2005b) The elephant and the blind men - Deciphering the Free/Libre/Open Source puzzle. *First Monday*, Special Issue.
- Kshetri, N. (2004) Economics of Linux adoption in developing countries. *Software, IEEE*, 21, 74-81.
- Lahti, C. & Peterson, R. (2005) *Sarbanes-Oxley IT Compliance Using COBIT and Open Source Tools*, CA, Syngress.
- Lakhani, K. R. & Von Hippel, E. (2003) How open source software works: "free" user-to-user assistance. *Research Policy*, 32, 923-943.
- Lakhani, K. R., Wolf, B., Bates, J. & Dibona, C. (2002) The Boston Consulting Group Hacker Survey (Release 0.73). IN Group, B. C. (Ed.), Boston Consulting Group. <http://www.ostg.com/bcg/BCGHACKERSURVEY-0.73.pdf>
- Lerner, J. & Tirole, J. (2002) Some Simple Economics of Open Source. *Journal of Industrial Economics*, 50, 197-234 (38).
- Lerner, J. & Tirole, J. (2005a) Economic Perspectives on Open Source. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Lerner, J. & Tirole, J. (2005b) The Scope of Open Source Licensing. *Journal of Law, Economics, and Organization*, 21, 20-56.
- Lin, Y.-H., Ko, T.-M., Chuang, T.-R. & Lin, K.-J. (2006) Open Source Licenses and the Creative Commons Framework: License Selection and Comparison. *Journal of Information Science and Engineering*, 22, 1-17.
- Ljungberg, J. (2000) Open Source Movements as a Model for Organizing. IN Hansen, H. R., Bichler, M. & Mahrer, H. (Eds.) *8th European Conference on Information Systems*. Vienna University of Economics and Business Administration, Austria. <http://www.viktoria.informatik.gu.se/groups/KnowledgeManagement/Documents/ecis2000.pdf>
- Lussier, S. (2004) New tricks: how open source changed the way my team works. *Software, IEEE*, 21, 68-72.
- Madanmohan, T. R. & De', R. (2004) Open Source reuse in commercial firms. *Software, IEEE*, 21, 62-69.
- Mccabe, B. (2004) Open Source Software and Australian IT Decision-Makers. S2 Intelligence.
- Mcgowan, D. (2005) Legal Aspect of Free and Open Source Software. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Michaelson, J. (2004) There's no such thing as a Free (software) lunch. *ACM Queue*, 2.
- Microsoft (2005) Shared Source Licenses. *Microsoft Shared Source Initiative*. Microsoft. <http://www.microsoft.com/resources/sharedsource/licensingbasics/sharedsourcelicenses.aspx>
- Mockus, A., Fielding, R. T. & Herbsleb, J. D. (2005) Two Case Studies of Open Source Software Development: Apache and Mozilla. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Moglen, E. (2003) SCO: Without Fear and Without Research. <http://www.gnu.org/philosophy/sco/sco-without-fear.html>
- Moon, J. Y. & Sproull, L. (2000) Essence of Distributed Work: The Case of the Linux Kernel. *First Monday*, 5.

- Moore, J. T. S., Wonderview Productions (Firm) & Seventh Art Releasing (Firm) (2003) *Revolution OS*, [S.l.] [Los Angeles, Calif.], Wonderview Productions ; Seventh Art Releasing [distributor].
- National Ict Australia (2004) Australian Public Licence B Version 1-1. National ICT Australia. http://nicta.com.au/director/commercialisation/open_source_licence.cfm
- Neumann, P. G. (2005) Attaining Robust Open Source Software. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- New Zealand State Services Commission (2006) Guide to Legal Issues in Using Open Source Software v2. State Services Commission. <http://www.e.govt.nz/policy/open-source/open-source-legal2/index.html>
- Nichols, D. M. & Twidale, M. B. (2003) The Usability of Open Source Software. *First Monday*, 8.
- Novell (2004a) Novell Statement on Patents and Open Source Software. Novell. <http://www.novell.com/company/policies/patent/>
- Novell (2004b) Novell Supports Innovation, Competition in Open Source with Patent Policy. Novell. <http://www.novell.com/news/press/archive/2004/10/pr04069.html>
- Nuvolari, A. (2005) Open Source Software development: Some historical perspectives. *First Monday*, 10.
- Office of Government Commerce (2004) Open Software Trials in Government: Final Report. United Kingdom. http://www.ogc.gov.uk/embedded_object.asp?docid=1002367
- Olson, M. (2006) Dual Licensing. IN Dibona, C., Cooper, D. & Stone, M. (Eds.) *Open Sources 2.0 : The Continuing Evolution*. Sebastopol, Calif., O'Reilly.
- Omar, I. (2005) The Penguin in Peril: SCO's Legal Threats to Linux. *First Monday*, 10.
- Open Invention Network (2005) Open Invention Network formed to promote linux and spur innovation globally through access to key patents. IN Open Invention Network (Ed.), Open Invention Network,. <http://www.openinventionnetwork.com/press.html>
- Open Source Initiative (2004a) The Open Source Definition. http://www.opensource.org/docs/definition_plain.php
- Open Source Initiative (2004b) Open Source Initiative OSI - Licensing. Open Source Initiative,. <http://www.opensource.org/licenses/>
- Open Source Initiative (2005) Open Source Initiative OSI - OSI History. IN Open Source Initiative (Ed.). <http://www.opensource.org/docs/history.php>
- Overby, E. (2003) Open Source Software- A Review of the Research Literature. Goizueta Business School, Emory University. http://userwww.service.emory.edu/~eoverby/files/overby_open_source_lit_review.pdf
- Ravicher, D. (2004) OSRM Position Paper: Mitigating Linux Patent Risk. IN Managment, O. S. R. (Ed.) 1.1 ed., Open Source Risk Managment. http://www.osriskmanagement.com/pdf_articles/linuxpatentpaper.pdf
- Raymond, E. S. (2001) *The cathedral and the bazaar : musings on Linux and Open Source by an accidental revolutionary*, Beijing ; Cambridge, Mass., O'Reilly.
- Robbins, J. (2005) Adopting Open Source Software Engineering (OSSE) Practices by Adopting OSSE Tools. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Robles, G., Gonzalez-Barahona, J. M. & Merelo, J. J. (2006) Beyond source code: The importance of other artifacts in software development (a case study). *Journal of Systems and Software - Selected papers from the fourth Source Code Analysis and Manipulation (SCAM 2004) Workshop*, 79, 1233-1248.
- Rosen, L. (2004) *Open Source Licensing Software Freedom and Intellectual Property Law*, Upper Saddle River, NJ, Prentice Hall.

- Rosen, L. (2005) License Proliferation. Open Source Developers Lab,.
<http://www.rosenlaw.com/LicenseProliferation.pdf>
- Rossi, C. & Bonaccorsi, A. (2005) Intrinsic Motivations and Profit-Oriented Firms in Open Source Software. Do Firms Practise What They Preach? , 32.
- Ruffin, C. & Ebert, C. (2004) Using open source software in product development: a primer. *Software, IEEE*, 21, 82-86.
- Rusovan, S., Lawford, M. & Parnas, D. L. (2005) Open Source Software Development: Future or Fad? IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Samoladas, I., Stamelos, I., Angelis, L. & Oikonomou, A. (2004) Open Source Software development should strive for even greater code maintainability. *Communications of the ACM*.
<http://doi.acm.org.mate.lib.unimelb.edu.au/10.1145/1022594.1022598>
- Scacchi, W. (2004) Free and open source development practices in the game community. *Software, IEEE*, 21, 59-66.
- Schmidt, K. M. & Schnitzer, M. (2003) Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market. *Harvard Journal of Law and Technology*, 16.
- Schmitz, P.-E. (2001) Study into the use of Open Source Software in the Public Sector :Part 2 Use of Open Source in Europe. IN Belgium, U. (Ed.) *Study into the use of Open Source Software in the Public Sector*. European Union :IDA.
<http://europa.eu.int/ISPO/ida/export/files/en/837.pdf>
- Schmitz, P.-E. & Castiaux, S. (2002) Pooling Open Source Software: An IDA Feasibility Study. IN Ida (Ed.), *European Communities: IDA*.
<http://europa.eu.int/ISPO/ida/export/files/en/1115.pdf>
- Serrano, N., Calzada, S., Sarriegui, J. M. & Ciordia, I. (2004) From proprietary to open source tools in information systems development. *Software, IEEE*, 21, 56-58.
- Shimizu, H., Iio, J. & Hiyane, K. (2004) Realities of Free/Libre/Open Source Software developers in Japan and Asia. *First Monday*, 9.
- Shirky, C. (2005) Epilogue: Open Source outside the Domain of Software. IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Skidmore, D. (2006) Too many Open Source Licenses! But do the existing licenses adequately encompass the diverse needs and concerns of particular stakeholders? IN Özel, B., Çilingir, C. B. & Erkan, K. (Eds.) *Towards Open Source Software Adoption: Educational, Public, Legal, and Usability Practices. OSS 2006 tOSSad workshop proceedings*. Como, Italy, TÜBİTAK (The Scientific & Technology Research Council of Turkey).
- Skidmore, D. (2007) (*Forthcoming*) Free / Libre and Open Source Software: Describing some Legal, and Software Engineering terms, and a taxonomy for classifying licenses. IN St.Amant, K. & Still, B. (Eds.) *Handbook of Research on Open Source Software: Technological, Economic, and Social Perspectives*. Idea Group.
- Soares, M. V. B. (2004) Reducing transaction costs in information infrastructures using FLOSS. *First Monday*, 9.
- Souphavanh, A. & Karoonboonyanan, T. (2005) *Free/Open Source Software: Localization*, New Delhi, Elsevier.
- Spinellis, D. & Szyperski, C. (2004) How is open source affecting software development? *Software, IEEE*, 21, 28-33.
- St. Laurent, A. M. (2004) *Understanding Open Source and Free Software Licensing*, O'Reilly.
- The on-Line Hacker Jargon File (2003) FUD. IN Raymond, E. (Ed.) *The on-line hacker Jargon File*. 4.4.7 ed. <http://www.catb.org/~esr/jargon/html/F/FUD.html>

- Torrise, S., Giuri, P., Ploner, M. & Rullani, F. (2005) Skills and Division of Labor in an Ecology of Floss Projects: Implications for Performance. *DRUID Tenth Anniversary Summer Conference 2005: Dynamics of Industry and Innovation: Organizations, Networks and Systems*. Copenhagen, Denmark.
<http://www.druid.dk/ocs/viewpaper.php?id=501&cf=3>
- Välimäki, M. (2003) Dual Licensing in Open Source Software Industry. *Systèmes d'Information et Management*, 8, 63-75.
- Välimäki, M. (2005) *The Rise of Open Source Licensing - A Challenge to the Use of Intellectual Property in the Software Industry*, Turre Publishing.
- Välimäki, M. & Oksanen, V. (2005) The impact of free and open source licensing on operating system software markets. *Telematics and Informatics*, 22, 97-110.
- Vetter, G. R. (2004) The Collaborative Integrity of Open-Source Software. *UTAH Law Review*, 2004, 563-700.
- Von Hippel, E. (2005) Open Source Projects as "User Innovation Networks". IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software*. Cambridge, Massachusetts, The MIT Press.
- Von Hippel, E. & Von Krogh, G. (2003) Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science. *Organization Science*, 14, 209-223.
- Watson, R. T., Wynn, D. & Boudreau, M.-C. (2005) JBOSS: The Evolution of Professional Open Source Software. *MIS Quarterly Executive*, 4, 329-341.
- Wayner, P. (2000) *Free for all : how Linux and the free software movement undercut the high-tech titans*, New York, Harper Business.
- Webbink, M. H. (2003) Understanding Open Source Software. *Computers and Law Journal*.
- Weber, S. (2004) *The success of open source*, Cambridge, MA, Harvard University Press.
- Weiss, D. (2005) Quantitative Analysis of Open Source Projects on SourceForge. IN Scotto, M. & Succi, G. (Eds.) *The First International Conference on Open Source Systems*. Genova, Italy, ECIG.
- West, J. (2003) How open is open enough?: Melding proprietary and open source platform strategies. *Research Policy*, 32, 1259-1285.
- Wheeler, D. A. (2002) More Than a Gigabuck: Estimating GNU/Linux's Size.
<http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html>
- Wong, K. (2004) *Free/Open Source Software Government Policy*, New Delhi, Elsevier.
- Woods, D. & Guliani, G. (2005) *Open Source for the Enterprise*, Sebastopol, O'Reilly.
- Zeitlyn, D. (2003) Gift economies in the development of open source software: anthropological reflections. *Research Policy*, 32, 1287-1291.

Putting the ‘X’ into XMDS

Clinton Roy

January 7, 2007

Abstract

Differential equations are used in a wide variety of disciplines, XMDS (eXtensible multi-dimensional Simulator) helps non-programmers generate correct, fast code to calculate numerical solutions to them. The current version of XMDS implements a variety of integration methods, can handle stochastic processes, can generate code to run on parallel environments and has a wealth of other features; however it is not as easy to extend as it could be. This paper will look at the current design of XMDS, the restrictions this design entails, a new design to allow better extensibility, and some possible future directions of the software.

1 Overview

Differential equations are used in a wide variety of disciplines:

- mathematics
- physics
- engineering
- finance
- economics
- chemistry
- theoretical biology

XMDS can solve ordinary or partial differential equations, as well as their stochastic forms. XMDS implements several types of integration algorithms:

- Runge-Kutta (explicit and implicit modes, adaptive step modes)
- Runge-Kutta Fehlberg (explicit and implicit modes, adaptive step modes)
- semi-implicit method

XMDS can easily generate code to solve problems in a parallel manner for parallel processing environments.

XMDS has been licenced under the GPL since inception.

The current implementation generates code that can solve a wide variety of problems but is not very extensible, this paper will discuss the current design of XMDS, and will explore approaches to improve the extensibility in the next major release.

Problems to be tackled include notation parsing, extensibility, maintainability and blue sky plans.

2 Mathematical details

This section courtesy of Paul Cochrane.

XMDS is designed to integrate PDEs of the general form:

$$\frac{\partial}{\partial x^0} a(x) = \aleph(x, a(x), p(x), b(x), \xi(x))$$
$$p^i(x) = \mathcal{F}^{-1} \left[\sum_j \mathcal{L}^{ij}(x^0, k_\perp) \mathcal{F}[a^j(x)] \right]$$

$$\frac{\partial}{\partial x^c} b(x) = \mathcal{H}(x, a(x), b(x))$$

Where

x spatial dimension

$a(x)$ main field

$b(x)$ cross-propagating field

$p(x)$ field defined in Fourier space

$\xi(x)$ noise terms

3 Motivation

The initial motivation for XMDS was the need of the ARC Centre of Excellence for Quantum-Atom Optics (<http://acqao.org>) requirement to solve differential equations without reinventing the wheel every time.

More widely, the numerical results of simulations are used as the main evidence in support of a theory in a lot of papers, but as there is no standard way of representing the solution it is often difficult to reproduce results. XMDS was written to solve this, and a critical part of that solution is the licence.

Also, it is important for the syntax used to describe the differential equations to XMDS be as similar as possible to that used in the paper, in order to ease the conceptual jump and to reduce the likely hood of errors.

4 Current Implementation

The current implementation takes its input from an ad hoc XML input design. While using XML for the input is fine, there are a lot of tags where a restricted range of values would make parsing easier. The new XMDS design will handle the current input format, as well as allowing XMDS to move forward and have a stricter input structure.

The current code that is generated by XMDS outputs the result in XSIL format, an XML format that allows XMDS to capture the problem and solution in one place.

The current implementation is written in C++ in an OO way, each class representing an integration or stochastic algorithm. Although this is a good way of splitting the problem up, this design is not granular enough to allow easy extensibility. Each class representing an algorithm also has to understand how to parse XML, how it interacts with other algorithms, and how all the other options interact with it; the code is too tightly coupled.

5 Tools

5.1 C++

As well as being the current target language C++ is also the current implementation language. Although fine for solving mathematical problems quickly, C++ is not a great choice for what is in effect a string manipulation and generation problem.

5.2 Autoconf

Autoconf is currently used to configure the compilation aspects of the generated programs. It detects the libraries and compilers that are available.

There is currently a lot of negative sentiment against Autoconf as it is seen to be too difficult to use. Most of the blame here falls on authors using the Autoconf macros incorrectly and generating `configure` scripts that are difficult to pass options to or override variables like the install `prefix`. Autoconf solves a vast number of terribly complicated problems that none of the so-called alternatives currently even attempt to solve, so XMDS will not be moving away from autoconf.

5.3 MPI

We use MPI to implement the parallel implementations, it is the de facto standard for communication in parallel programs and has many implementations.

5.4 FFTW

FFTW is the library used to do Fourier transforms. It is fast, flexible and easy to use.

Unfortunately at the moment there is a disjoint set of features across major versions: version 2 supports MPI operations while version 3 has CPU specific support to speed up transforms. It would be well worth spending some time on implementing MPI for major version 3, greatly simplifying the configuration of FFTW as well as gaining speed improvements.

6 New Design

The new design of XMDS should allow new integration methods to be added without modifying any existing code. It should allow much easier generation and stitching together of code fragments. The code templates should be removed from the XMDS core and placed in easy to write user libraries.

The following outlines the execution steps of the new design:

1. parse input XML file
2. parse the library files
3. traverse the input XML locating the matching handler from the libraries
4. run each matching handler, building up the output program

6.1 Parsing the Input

The current implementation of XMDS uses an XML library to parse the input. Unfortunately, there is then another step required to parse some elements, take the `domain` tag for example:

```
<domain> (1, 2) (3, 4) (5, 6) </domain>
```

Hand coded parsing is used to parse the above, where it would be preferable to parse the following input completely with an XML parser:

```
<domains>
  <domain min="1" max="2">
  <domain min="3" max="4">
  <domain min="5" max="6">
</domains>
```

It will be possible to write a fairly simple translator to convert the ad hoc XML format into the newer XML format, such that the handling of backwards compatibility is confined to one place. In future, if it desirable to have some automatic editor for the input, a much stricter and form on the input with an accompanying DTD will be required.

In order to confine the parsing of XML to one module it is necessary to provide a syntax for accessing XML nodes:

```
domains.domain[0].min = 1
domains.domain[0].max = 2
domains.domain[1].min = 3
domains.domain[1].max = 4
domains.domain[2].min = 5
domains.domain[2].max = 6
```

Further syntax provided will be explored further.

6.2 Parsing the Libraries

The libraries are written in a simple macro based language that allows certain macros to be associated with particular tags, the values of nodes in the input XML file to be accessed, and some basic flow control (decisions, loops).

The substitution approach for the next version of XMDS is to provide a simple templating language that allows substitution akin to shell scripting, so that code like:

```
fprintf(outfile, "%s(:) = %s%i(:,%i);\n", variableName(i)->c_str(), datFileNameBase, iD, i)
```

into

```
$i(:) = $datFileNameBase$iD(:,$i)
```

leading to much easier to read code. As well as simple substitution the templating language has simple macros that can take typed parameters (strings, XML node names, boolean expressions and numerical numbers):

```
\foo(arg:one, arg:two) {
  \bar(one, two, three)
}
```

6.3 Traversing the Input

The particular technique of traversing the input tree is a depth first traversal: children are visited before their parents.

This model lets the XML file drive XMDS, rather than the library files proscribing what happens, as in the current XMDS. This removes a lot of control code currently sprinkled all the way through XMDS, removes a lot of dispatching code and makes things much more flexible.

The process of finding a matching handler is perhaps the most important part of the new XMDS . One matching handler is called for each tag in the XML file. The matching can be done on an arbitrary boolean expression that can look at the value of any of XML input file:

```
\match(tag:simulation.algorithm,
      expr:simulation.algorithm == "Runge-Kutta") {
% Runge-Kutta integration implementation
}
```

A common operation is to loop across instances of the same tag, some pseudo elements can ease the creation of generated code:

```
\iterate(domains.domain, i){
  """
  printf("Domain $i._i of $i._n: ($i.min, $i.max)\n");
  """
}
```

6.4 Building up the Output

To ease the creation of the output file, XMDS has the flexible idea of "sections". If a handler creates output, the output must be directed to one or more sections. At the end of the XML traversal the sections are weaved together to form the whole program. As the handlers in the library are responsible for the layout of these sections, the ultimate structure of the generated code is under the complete control of the XML script.

The top node of the tree needs to be specially named (`main`), and the demarcation of sections is most easily done inside output strings:

```
\match(tag:simulation){
  \output(section:main)
  """
  \section(includes)
  \section(defines)
  \section(prototypes)
  \section(impls)
  \section(main)
  """
}
```

7 Unsolved Problems

7.1 Default values

One possible way is to have a `default()` function that can be called from the matching expressions. This would change the handling code to:

```
\handle(tag=simulation.errorChecking,
        expr=default()) {
  % Default to enable error checking.
}
```

7.2 Poor Parsers

Already two different parser generators have been trialled and discarded: `python-parsing` and `simpleparser`. While both are quick and easy to use for simple lan-

guages they don't support a powerful enough form of backtracking, precedence has to be done by hand and their error recovery is next to useless.

The author is working towards a Python target for version 3 of Antlr. Antlr generated parsers are quick, easy to follow, have powerful backtracking and great error recovery. They can also build the parse tree automatically, rather than having to be built up by XMDS itself.

8 Looking forward

8.1 Easier Input

Although great for portability, XML isn't always the easiest way to input numerical equations. Some sort of flexible forms system along with an equation editor could make XMDS much easier to use. Even a \LaTeX parser would be possible.

8.2 Graphical Output

The round trip for numerical analysis ends with graphics, either images, graphs or animations of numerical results. At the moment XMDS does not directly support the generation of graphics, instead external programs are used to convert the raw numerical data into graphical output.

In order to tighten the analysis loop it could be useful for XMDS to handle generating graphical output.

8.3 Web frontend

It's conceivable that the easiest way of solving both the input and output problems is through the web; using smart forms (XForms if they're ever actually implemented, or failing that an Ajax front end) plus a ECMAScript equation editor.

The form could contact a backend server to perform the calculation, as well as give the user feedback about how long a job will take, as well as control over the task.

Results, in graphical form, could be fed back to the web browser and viewed via a Java client application, which would allow dynamic tweaking of display parameters.