Distributed Database*(preliminary)*

Infinite Delta Corp, InfiniteDelta.com, Ty@InfiniteDelta.com
2460 Maple Valley Dr SE, Kentwood, MI 49512

April 24, 2020

1 Introduction

The Distributed Database (DDB) is a simple record/field oriented distributed database. It minimizes the need to be **Network Connected** at all times. The goal is to achieve the highest functionality in degraded modes.
DDB addresses the problem of system down time by keeping sufficient information on local computers to perform the majority of its tasks. This particularly supports **Real Time** and **Embedded Systems** since the relevant data is now local. It also reduces security risks, since only the relevant data is maintained, so a stolen computer, or a malicious attacker has little to gain.

1.1 Identification

April 24, 2020
Ty Zoerner, Infinite Delta Corp, `InfiniteDelta.com`
`ty@InfiniteDelta.com` Infinite Delta Corp, 2460 Maple Valley Dr SE, Grand Rapids, MI 49512-3801

Contents

1 Introduction **1**
1.1 Identification 1

2 Goals **3**
2.1 An Engineering Distributed Database 3
2.2 Degraded Systems 3
2.3 Low development costs 3
2.4 High performance 3
2.5 Relational 3
2.6 International 3
2.7 Hardware Interfaces 3
2.8 Security 3
2.8.1 Multiple Levels of Security 4
2.8.2 User/Group style local protections 4
2.8.3 Network 4
2.8.4 Need to Know 4
2.9 Data Integrity 4

2.10 Record Variation 4

3 Architecture **4**

4 Engineering Data Specifications **5**

5 Organization **5**
5.1 Records 6
5.2 Tables 6
5.3 String Enumeration 6
5.4 No Locking 6
5.5 Optimized functions 6
5.6 Capabilities and Dependencies 6

6 Copying **6**

7 GNU LESSER GENERAL PUBLIC LICENSE **7**
7.1 References 11

2 Goals

This section covers specific DDB goals including the primary goal to support complex engineering problems, complete with highest application functionality in degraded systems.

2.1 An Engineering Distributed Database

DDB primarily targets the distributed engineering environments supporting most levels of computing platforms. DDB understands standard engineering types, performance restrictions, and safety issues.

2.2 Degraded Systems

Achieve the highest application functionality in degraded systems.
The cost of system down time can be greatly reduced with minor planning. Many solutions to the down time problem is to use expensive networks and/or servers, when only planning is required. When software application designers are involved with degraded planning, unique and sometimes expensive solutions to maintain are used.

2.3 Low development costs

Low development costs can be achieved by separating the application developer from the tedious administration associated with a distributed database. DDB allows application developer to work (mostly) without knowledge of the final system data distribution architecture. The developer need only to include their dependencies on the system.

2.4 High performance

Typical **Real Time** and/or **Embedded Systems** require numerically compressed values for actions and data decisions and avoid string parsing. DDB only uses these numerically compressed values when communicating between systems, fully supporting these performance requirements.
However, to support user interactions, an local lookup dictionary can be maintained by DDB to support user interactions.

2.5 Relational

Records containing the same field can be logically joined. An index is kept local on the system to remove any network traffic or delays. Fields specified to be unique, aka record-index, allow some applications to maintain only the latest updated information. Applications not declaring a unique field can have a log of updates.

2.6 International

Multiple language support can be provided by the DDB required dictionary for user interaction. Unlike **gettext**, this could be dynamic and context sensitive.

2.7 Hardware Interfaces

DDB design shall support multiple hardware interfaces, but is optimized around TCP/IP UDP and multicast.

2.8 Security

DDB includes several security features.

2.8.1 Multiple Levels of Security

Encryption can be enabled and assigned on a field bases, allowing partitioning with multiple levels of security within the same record. Data can be distributed to many systems, but only the applications with the specific keys and passwords to the field can access it.
This support the highest functionality and minimizes the effects of stolen or compromised systems.

2.8.2 User/Group style local protections

DDB fully uses Unix/GNU-Linux user/group protections for reporting data. DDB treats field separately and can be assigned to a specific group. Allowing multiple groups to coexist on the same system. It can be used to determine the data to be maintained on the specific system.

2.8.3 Network

DDB can work behind a VPN or use SSH piping when entering or leaving a protected network. Within the protected network, DDB filters packets only relevant to the applications and users on the system.

2.8.4 Need to Know

DDB can be configured to distribute only the required data to specific systems. Separating personal identification, medical, and accounting information is now required from most governments. A stolen lab computer may contain minimal identification and medical information, but not enough for identity theft. A context sensitive DDB dictionary can further be used to filter sensitive into approved releasable media.

2.9 Data Integrity

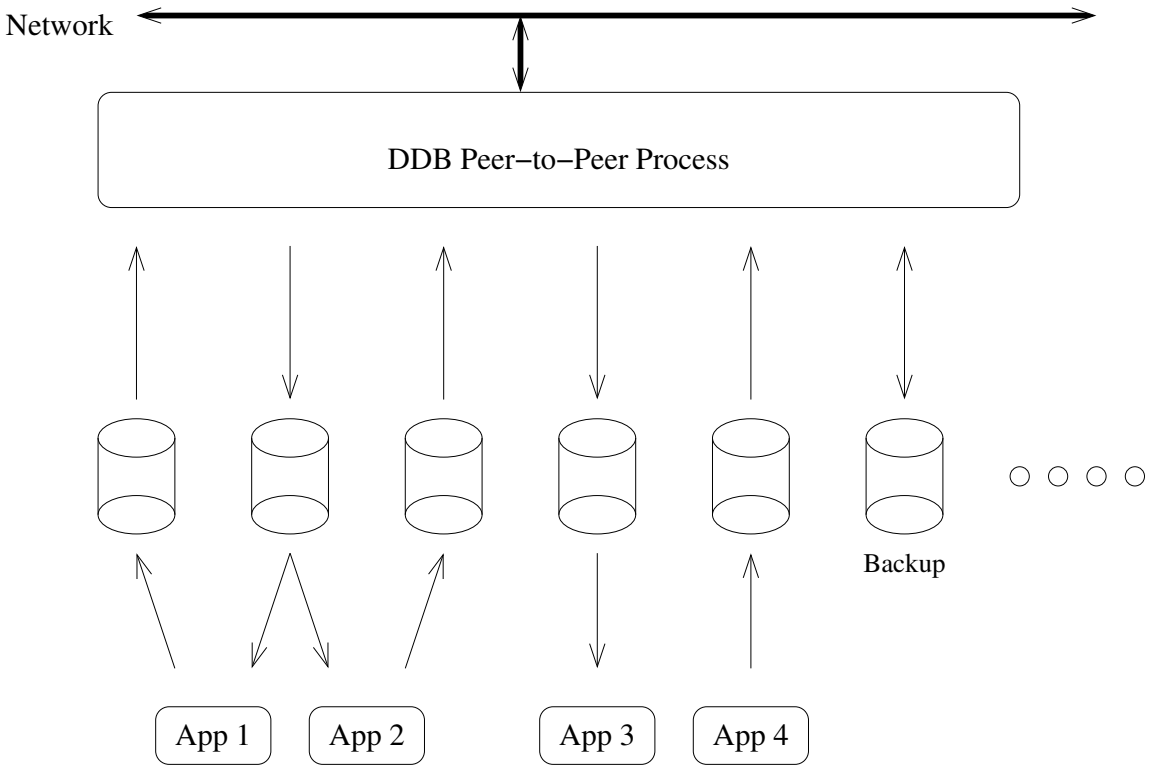
DDB can guarantee limited data integrity after a suspected network disconnect or lost packet. Each DDB requests updates from the network.

2.10 Record Variation

DDB shall support records with varying fields and units. Thus allowing embedded applications free of non-relevant fields and conversions.

3 Architecture

The basic DDB architecture consists of a network peer-to-peer process feeding one way tables. Systems including embedded systems, may use restricted shared memory or files for these tables. With proper permissions, the applications can read and monitor one or more of these tables. Conversely, if an application has data to report, it creates and maintains an out going data file. In all cases, only one process (application or network process) is responsible for maintaining a given file.



$Id: architecture.fig 1 2013–01–07 03:11:41Z ty $

4 Engineering Data Specifications

DDB uses the Electronic Capabilities Manager (ECM) method of specifying DDB requirements within the application source code. Typically within Doxygen's comment strings /** or ---. Example:

```
/**
  Calculates: energy W/hrs = windmill(wind miles/hr, model);
  */
```

Informs DDB that the application supplies an energy table for a windmill based on the speed of the wind in the units miles per hour.
An associated include file is generated for the application to fill in an energy table. It also publishes this capability to the current and future users.

5 Organization

A very important distinction to note: **DDB does not use multiple individual tables** as many databases. DDB only has and maintains records which can be joined or tightly coupled. DDB does support logical or virtual Tables. DDB defines *Table* as an array of records containing a specific attribute. The above windmill example would create a windmill *Table*.
Each field within a record can be owned by different processes on different systems. Multiple sources for a field are flagged by DDB as a **warning** and are treated as unique records.
Every record has a unique local *rec_id*. DDB defines *rec_id* as an array of pointers to the actual record data.

Keys can be used for sorted or creating groups of records. DDB defines *Key* as a pre-selected set of records. SQL users may consider a pre-compiled select statement.

5.1 Records

All records are referenced by rec_id. To preserve full database atomic during updates, the reference table address is replaced pointing to a completed record structure. The old record shall be garbage cleaned up only after a specified amount of time to allow slower applications to complete their task.

5.2 Tables

Tables (shared memory or files) are of three base types:

History records are only appended. Some history tables are a circular queue, others a maintained as daily files. History records are used to generate and update data table records. Note that data tables are not always required. The life of the history table depends on the need for long term records.

Data tables are an indexed lists of data records. Generally updated from history table updates. A data tables can report change amounts by keeping available a previous record.

Key is a sorted table containing index values into data tables. Key table are only generated on local systems if required by a local application.

5.3 String Enumeration

DDB uses string enumeration for the following:

Dynamic Multilingual support for immediate updates between international groups.

Embedded applications can have the strings removed from the application.

Performance gains are seen when string manipulations are removed from real time or embedded applications.

Space gains are seen when applications and databases only store enumerated values.

5.4 No Locking

DDB does not support record locking, only field ownership. Locking is dangerous in most prioritized and distributed systems where high functionality is important. The same functionality can typically be achieved via *request* and *current* data organization. An example camera manager application can take a prioritized list of *requests* from the system, but will only report the *current* or actual camera configuration.

5.5 Optimized functions

Some functions can be compile time optimized, such as *sum()*, *count()*, *select()*, and unit conversions.

5.6 Capabilities and Dependencies

Capabilities and *Dependencies* are electronically available for complete system analysis.

6 Copying

7 GNU LESSER GENERAL PUBLIC LICENSE

```
   GNU LESSER GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.


  This version of the GNU Lesser General Public License incorporates
the terms and conditions of version 3 of the GNU General Public
License, supplemented by the additional permissions listed below.

  0. Additional Definitions.

  As used herein, "this License" refers to version 3 of the GNU Lesser
General Public License, and the "GNU GPL" refers to version 3 of the GNU
General Public License.

  "The Library" refers to a covered work governed by this License,
other than an Application or a Combined Work as defined below.

  An "Application" is any work that makes use of an interface provided
by the Library, but which is not otherwise based on the Library.
Defining a subclass of a class defined by the Library is deemed a mode
of using an interface provided by the Library.

  A "Combined Work" is a work produced by combining or linking an
Application with the Library.  The particular version of the Library
with which the Combined Work was made is also called the "Linked
Version".

  The "Minimal Corresponding Source" for a Combined Work means the
Corresponding Source for the Combined Work, excluding any source code
for portions of the Combined Work that, considered in isolation, are
based on the Application, and not on the Linked Version.

  The "Corresponding Application Code" for a Combined Work means the
object code and/or source code for the Application, including any data
and utility programs needed for reproducing the Combined Work from the
Application, but excluding the System Libraries of the Combined Work.

  1. Exception to Section 3 of the GNU GPL.

  You may convey a covered work under sections 3 and 4 of this License
without being bound by section 3 of the GNU GPL.

  2. Conveying Modified Versions.
```

```
  If you modify a copy of the Library, and, in your modifications, a
facility refers to a function or data to be supplied by an Application
that uses the facility (other than as an argument passed when the
facility is invoked), then you may convey a copy of the modified
version:

   a) under this License, provided that you make a good faith effort to
   ensure that, in the event an Application does not supply the
   function or data, the facility still operates, and performs
   whatever part of its purpose remains meaningful, or

   b) under the GNU GPL, with none of the additional permissions of
   this License applicable to that copy.

  3. Object Code Incorporating Material from Library Header Files.

  The object code form of an Application may incorporate material from
a header file that is part of the Library.  You may convey such object
code under terms of your choice, provided that, if the incorporated
material is not limited to numerical parameters, data structure
layouts and accessors, or small macros, inline functions and templates
(ten or fewer lines in length), you do both of the following:

   a) Give prominent notice with each copy of the object code that the
   Library is used in it and that the Library and its use are
   covered by this License.

   b) Accompany the object code with a copy of the GNU GPL and this license
   document.

  4. Combined Works.

  You may convey a Combined Work under terms of your choice that,
taken together, effectively do not restrict modification of the
portions of the Library contained in the Combined Work and reverse
engineering for debugging such modifications, if you also do each of
the following:

   a) Give prominent notice with each copy of the Combined Work that
   the Library is used in it and that the Library and its use are
   covered by this License.

   b) Accompany the Combined Work with a copy of the GNU GPL and this license
   document.

   c) For a Combined Work that displays copyright notices during
   execution, include the copyright notice for the Library among
   these notices, as well as a reference directing the user to the
   copies of the GNU GPL and this license document.
```

```
   d) Do one of the following:

       0) Convey the Minimal Corresponding Source under the terms of this
       License, and the Corresponding Application Code in a form
       suitable for, and under terms that permit, the user to
       recombine or relink the Application with a modified version of
       the Linked Version to produce a modified Combined Work, in the
       manner specified by section 6 of the GNU GPL for conveying
       Corresponding Source.

       1) Use a suitable shared library mechanism for linking with the
       Library.  A suitable mechanism is one that (a) uses at run time
       a copy of the Library already present on the user's computer
       system, and (b) will operate properly with a modified version
       of the Library that is interface-compatible with the Linked
       Version.

   e) Provide Installation Information, but only if you would otherwise
   be required to provide such information under section 6 of the
   GNU GPL, and only to the extent that such information is
   necessary to install and execute a modified version of the
   Combined Work produced by recombining or relinking the
   Application with a modified version of the Linked Version. (If
   you use option 4d0, the Installation Information must accompany
   the Minimal Corresponding Source and Corresponding Application
   Code. If you use option 4d1, you must provide the Installation
   Information in the manner specified by section 6 of the GNU GPL
   for conveying Corresponding Source.)

  5. Combined Libraries.

  You may place library facilities that are a work based on the
Library side by side in a single library together with other library
facilities that are not Applications and are not covered by this
License, and convey such a combined library under terms of your
choice, if you do both of the following:

   a) Accompany the combined library with a copy of the same work based
   on the Library, uncombined with any other library facilities,
   conveyed under the terms of this License.

   b) Give prominent notice with the combined library that part of it
   is a work based on the Library, and explaining where to find the
   accompanying uncombined form of the same work.

  6. Revised Versions of the GNU Lesser General Public License.

  The Free Software Foundation may publish revised and/or new versions
of the GNU Lesser General Public License from time to time. Such new
```

versions will be similar in spirit to the present version, but may
differ in detail to address new problems or concerns.

  Each version is given a distinguishing version number. If the
Library as you received it specifies that a certain numbered version
of the GNU Lesser General Public License "or any later version"
applies to it, you have the option of following the terms and
conditions either of that published version or of any later version
published by the Free Software Foundation. If the Library as you
received it does not specify a version number of the GNU Lesser
General Public License, you may choose any version of the GNU Lesser
General Public License ever published by the Free Software Foundation.

  If the Library as you received it specifies that a proxy can decide
whether future versions of the GNU Lesser General Public License shall
apply, that proxy's public statement of acceptance of any version is
permanent authorization for you to choose that version for the
Library.

7.1 References

A. Integrated Modular Avionics (IMA)
http://en.wikipedia.org/wiki/Integrated_modular_avionics
https://www.sto.nato.int/publications/STO%20Educational%20Notes/RTO-EN-SCI-176/EN-SCI-176-02
pdf

B. Delivering Capabilities through Requirements Management a process investigation
http://infinitedelta.com/wp/electronic_capabilities_management.pdf

C. DO-178C, Software Considerations in Airborne Systems and Equipment Certification
http://en.wikipedia.org/wiki/DO-178C

D. Modular Tickless Prioritized Preemptive RTOS http://infinitedelta.com/wp/avionics_rtos.
pdf

E. Multiple Security Levels (MLS): http://en.wikipedia.org/wiki/Multilevel_security

F. Dr. Tom Herald ‘‘Affordable Architectures’’, Oct, 2011
https://www.incose.org/products-and-publications/papers-presentations-library

G. Single Event Upsets (Soft Errors) http://en.wikipedia.org/wiki/Soft_error