CTX Tanker & Bulk Carrier Casualty Database

Version 4.6

Jack Devanney, Patrick Doyle

Sisyphus Beach

Tavernier, Florida

2011

2011-05-23
This is an incomplete draft for discussion.
Please send any comments to cdb@c4tx.org.

Contents

Chapter 1

Introduction

1.1 Goal

The goal of the CTX CDB is to record all *major* casualties involving ocean going tankers or bulk carriers. Ships with a size of less than 5000 dwt are excluded as are barges and integrated tug barges. Casualties in repair yards, during demolition, or while being towed to demolition are excluded. A casualty is *major* if any of the following are true:

- Someone was killed or seriously injured.
- Oil was spilled, either cargo or bunkers.
- The ship suffered a total loss of power for more than one hour, or a loss of power requiring tugs to be mobilized.
- The casualty involved a fire, collision or grounding.
- The casualty involved a structural failure requiring tugs to be mobilized, or cargo transfered. or the ship diverted.
- The casualty resulted in the ship being lost, scrapped, or declared a Contructive Total Loss.

Currently, the database falls far, far short of this goal; but this is our target. The database may also contain non-major casualties which the CTX has become aware of.

1.2 Purpose

Just about all the reasonably complete ship casualty databases are proprietary. Many countries maintain more or less complete databases of the tanker casualties and spills that occur in their own waters, and make this data freely available to the public. These include Australia, the UK and the

USA. However, only private for profit groups have stepped in to combine the data worldwide.

To access these private databases, one must

- pay a substantial amount of money,
- much worse, accept restrictions on disclosure.

No legitimate researcher can accept restrictions on disclosure. This violates one of the most basic principles of science. If anyone publishes some sort of summary or analysis of the casualty data, then anyone else should be able to go to the data and reproduce the same summary. If he can't, we don't have science; we have advertising. Intertanko, the tanker owner lobbying group, claims that spills dropped dramatically between the 70's and the 80's. It turns out that this is true; but, since the data on which this claim is based is not public, the claim is in a strict sense meaningless.

This is not a theoretical issue. Over 2003-2005, the Safedor POP&C project spent millions of European taxpayer Euros developing a proprietary tanker casualty database. One of their major results is that structural failure is not a particularly important cause of tanker deaths or spillage. The CTX database claims that structural failure is by far the single most important cause of both tanker deaths and spillage. Why this fundamental, critically important difference? Anyone can examine the CTX database and see if they agree with the cause assignment casualty by casualty. No third party can do the same for the Safedor data.[1]

Secondly, if the casualty data is publicly available anyone can question it, correct it, and add to it; and the quality of the data, which in the private databases is often execrable, will be improved. Many seafarers, salvors, spill responders, etc know a great deal about a handful of casualties; but there is no easy way for them to contribute that knowledge. And it certainly makes little sense to contribute if the recipient then takes this contribution and turns it into his/her private property.

Thirdly, if the data is publicly and easily available, maybe, just maybe, the people writing tanker and bulk carrier regulation might take a look at it. IMO is passing tanker regulation left and right without apparently even looking at the casualty data. Misdirected regulation with unintended consequences is the inevitable result.

The CTX Casualty Database (CDB) is a bit different from any of the existing databases in several respects.

[1] The CTX guesses that the Safedor database calls many structural failures, fires or explosions. Many, if not most, major tanker structural failures result in a fire or explosion. But since we cannot examine the Safedor data, this is only a guess.

Public for Use Most importantly, all the individual casualty data is available to anyone without charge, under the Gnu Free Documentation License (GFDL) The CDB can be downloaded by anyone who wishes to do their own analysis of the data. The GFDL in no way restricts anyone from doing legitimate analysis, nor passing the data on. Its goal is to ensure that any derivative documents are free in the same sense that the CDB is free. The last thing we want is for the CDB data to fall into proprietary hands.

Do we need a change name if you change data warning

Public for contribution The CTX is painfully aware that our casualty data is incomplete and is bound to be incorrect in places. We know that behind every casaulty, every spill, there is a story; and in many, if not most, cases we know we do not have that story, certainly not the complete story. Especially when it come to cause, the CDB is replete with blanks and hopefully informed guesses. We also know that somebody out there knows what really happened. The CTX will be happy to receive any corrections, additions, comments, etc which contributions, if used, will be fully credited if desired. The CTX will also protect the identities of contributors who wish to be so protected. We are particularly interested in receiving the offical investigation reports. See How to Contribute for details.

Focus on Real Cause Almost all the current spill databases, public or private, focus on what happened after the spill. The CTX data base focuses on what happened before the spill, rather than the post spill impact and attempts to ameliorate those impacts, although links to the post-spill material are included when available. All the existing databases, private and public, are sadly deficient when it comes to cause. Calling "grounding" the cause of a casualty is like blaming the earth for an airplane crash. Not only is "grounding" never the cause of a casualty, but by talking about grounding as if were the cause, we focus attention away from measures that could have prevented the grounding.

The CTX database tries to get to the real causes, which are never grounding, fire/explosion, or collision. Ships don't ground themselves, nor all of sudden blow up, nor run into each other for no reason.[2]

[2] This is hardly a new idea. The UK Board of Trade going back at least to the 1870's published tables dividing the various causes of wrecks and collisions into three classes. The causes themselves were such things as "Want of Lights or Buoys", "Bad lookout", "Improper Stowage", etc. [Martin, The History of Lloyds, p 392-395]

Something always happens first. Instead of "a" cause, for each casualty, the CDB has a series of *events*. Some of these events can be causal. ***But to be causal an event must be an error, a defect, or a failure.*** Grounding, fire/explosion, collision and the like can never be causal. If the causal event(s) are not known, often the case, we explicitly set cause to UNKNOWN. The CTX hopes that contributors will be able to fill in these all important blanks

Allow Multiple Causality Multiple causality is a commonplace in casualties, often taking the form of the casualty would not have occured if any of A, B, or C had not happened. Any database which cannot handle multiple causality is hopelessly crippled. In the CTX database each event in the sequence may be either a primary cause, a necessary cause, a probable cause, a secondary cause, or non-causal.

Allow Any Combination of Events and Ships Databases which attempt to divide casualties into collisions, or groundings, or whatever not only ignore the real cause(s), but also face an intractable problem. In the real world, casualties don't partition themselves into neat categories. Many ship casualties involve combinations of structural failure, fire, grounding, etc. The sequence: collision, fire, grounding is not uncommon. The NASSIA casualty in the Bosporus which killed 42 includes this sequence.[3] Is the Nassia casualty a collision or a fire or a grounding? The correct answer is "all of the above". And the correct answer for cause is "none of the above". The primary cause was a black out on the bulk carrier BC Shipbroker. Without electrical power, the Shipbroker had no steering, and turned into the Nassia.

Moreover, some casualties involve multiple occurances of the same event. Some such as the HYDE PARK involve multiple collisions (and allisions) and some involve multiple groundings. In both the massive URQUIOLA and SEA EMPRESS spills, the great bulk of the spillage occurred in the secondary groundings. The SEA EMPRESS grounded at least four separate times. Roughly 2500 tons of cargo were spilled in the initial grounding. Over 69,000 additional tons were lost in the subsequent groundings. All sorts of combinations are possible. Well after the PARACAS hit and sank the Texaco Caribbean, two other ships hit the wreck, killing an additional 43 people. The CTX database

[3] If you are reading this manual on-line, and a ship name occurs in small caps LIKE THIS, then by clicking on the name, your browser should display the CDB Precis file (See Section 3.1) for that casualty.

views a casualty as a sequence of events. A casualty can have any combination of events including multiple structural failures, machinery failures, collisions, and groundings.

Some casualty databases are built around the premis that a casualty involves A ship. Multiple ship casualties are treated as an exception. Often the second ship cannot be described as completely as the "first" ship. And often the database can't handle more than two ships. In the CTX database, a casualty can involve any number of ships all of which are treated equally. In fact a *ship*ship doesn't even have to be a ship. If a ship runs into an offshore platform, that platform becomes a *ship* for the purposes of the CDB.

Allow detailed description of casualty Most ship casualty databases are designed to allow straightforward analysis of the data: spill volume by year or region for example. The CTX CDB also allows this sort of extraction and aggregation, albeit with far more exphasis on the real causes. But normally if the ship survives, someone (Class, owner, yard) has much more detailed data on the casualty. This includes the location and extent of structural damage, machinery sub-systems involved, course and speed for collisions, etc.

The CTX database is set up so that, if such data is available, it may be included and publicly recorded in machine readable form. Among other things, this allows real analysis of residual strength and spillage and flooding.[4] The hope here is that tanker and bulk carrier regulation and design can be based on data available for scrutiny and correction, and not on sanitized, unauditable, unanalyzable Class secrets.[5]

[4] To handle situations in which only partial data is available, the data base is at times redundant. For example, in the load pattern description, there is a field for the total cargo deadweight and fields for the contents of each tank. If you know and enter the contents of each tank, the cargo deadweight field is redundant, since it can be calculated from the more detailed data. But put such fields in anyway. Some applications may be too lazy or not smart enough to compute the redundant data.

[5] A particularly bizarre result of Class confidentiality is *non-dimensionlization* of hull penetration data. In the past, the only entities that have a reasonably complete database on hull penetration are the Classification Societies collectively. But when IMO turned to the Classes for that information, they ran up against the confidentiality clauses in the contracts between Class and individual owner. To get around this, Class non-dimensionalized the data by ratioing it to the size of the ship. For example, in the case of side damage, they divided the depth of penetration by the beam of the damaged ship. This effectively hid the ship's identity, and along the way made it impossible to check the data for accuracy, completeness, etc. Even the IMO delegates who write the regulation do not see the real penetration data. Much worse, in using the non-dimensionlized data in evaluat-

Another hope is that the ability to record detailed data will result in the data actually being recorded in a publicly accessable fashion. Since about 2003, all big ships must be fitted with Voyage Data Recorders, black boxes which record position, course, speed, etc. But to CTX's knowledge, the only time this data has been used after casualties is in court cases. Otherwise it seems to just disappear.

Separate objective facts from subjective judgements In so far as possible, a casualty database should separate supposedly objective facts, such as time, location, and the like, from necessarily subjective judgements, such as assignment of cause, causal factors, and the effectiveness of Risk Control Options (RCO). Since data may be missing or inaccurate, this is not always possible; but the CTX CDB attempts to avoid compounding this problem in several ways.

Most importantly, an event is not automatically a cause. For an event to be considered a cause, the coder must separately and explicitly claim it is a cause; and he must also code the quality of evidence behind this claim. The existence of an event and its causalness are separated.

Secondly, causal factors are maintained in a separate category from other more objective fields. Applications are encouraged to display or treat these fields in a different manner from the ordinary fields. For example, the CTX web interface to the database uses a different background color when displaying causal factor fields. The same thing is true of RCO assessment fields.

Thirdly, all cause assignment, causal factor, and RCO assessment fields are coded in a manner that rates the degree of confidence we have in the accuracy of the judgement,

As a result the CTX CDB can be used by people who may not agree with our more subjective judgements, If they wish, they can simply ignore the more subjective fields, or at least the more speculative such judgements,

Needless to say, the fact that a field is "objective" does not guarantee its accuracy. For example, the location where a fire started may

ing new designs, IMO had no choice but to assume that the penetration is proportional to the size of the struck ship. For example, in the same collision, IMO assumes that a narrow ship suffers less penetration than a wide ship. This of course is total nonsense. The depth of penetration depends on the size of the striker, not the size of the struck. The whole system is not only totally opaque, but ridiculously biased against bigger ships. Non-dimensionalized penetration data is useless.

be a guess, inferred from whatever data we have. Conversely, there are many situations in which the CTX causual judgements are indisputably accurate.

Expandable The CTX Casualty database is by design easily expandable. Primarily this is accomplished by the use of XML which allows the introduction of new elements and new fields without affecting code based on the existing database. In addition, the description of the various fields is almost completely table driven. We have already had two major expansions to the database. In 2.6, we added a description of war/piracy attacks. In 2.7, we expanded the database so it could handle all bulk carrier casualties, not just tankers.[6] Only modest changes to the code were required. And we have made numerous more minor changes in which no change at all of the existing code was required. It would not be difficult to further expand the database to include say LNG and LPG carriers, or even container ships. And it would be easy to add elements describing such thing as spill response and cost.

[6] Since the CTX database started out life as a tankers only database, some of the terminology is tanker oriented.

1.3 History

Version 1 The CDB started out in 2005 as a simple table, plus a set of precis files. The data was organized around the concept that a casualty involved a combination of date and ship. The initial data base was quite limited, containing little more than date, ship name, deaths, volume spilled, and a set of event codes. Each casualty could only have a single cause. Many new fields were added to the table in Versions 1.1 through 1.5. It became clear that the number of columns — most of which were blank for lack of information — in the table was getting out of hand.

Version 2 In 2007, the CDB was converted to XML which offered a far more flexible, hierarchical structure. Sub-elements were introduced to allow recording the casualty in far more detail, when such details were available. But a casualty was still viewed as a combination of date and ship. Multi-ship casualties were treated as an awkward special case.

Version 3.0 In mid-2008, the <ship> sub-element was introduced. This allowed a casualty to have any number of ships and all such ships were treated more or less equally. But the structure was still ship centric with most of the casualty details being contained in ship sub-elements

Version 3.2 In Fall, 2008, the event code field was replaced with the <event> element. This was a seminal change, allowing both multiple causality, and removing the automatic association between event and cause. But most of the data remained in the ship sub-elements, and the link between event and ship was awkward, sometimes redundant, and in some cases ambiguous.

Version 4.0 In February, 2009, Version 4.0 completed the CDB's evolution from a ship centric data base to an event centric data base. All the ship sub-elements describing the casualty as opposed to the ship were moved into the event elements. This resulted in a simpler, much more flexible structure allowing the CDB to really tell the story (when we had the data) in an easily machine readable fashion. The `sid` field was introduced to eliminate ambiguities between event and ship(s).

Version 4.3 Version 4.3 released in late 2009 focused on event details, allowing the database to record in machine readable form a great deal more information, such as course and speed changes. 4.3 also intro-

duced a far more flexible `area` field to allow charting applications to correctly display casualties by region and body of water.

Version 4.4 Version 4.4 released in March, 2010 adopted a naming convention for separating the fields into three categories

1. Factual: data which at least in theory requires no subjective judgement.
2. Causal: assignment of cause and causal factors.
3. RCO: fields which make a judgement about how effective a particular RCO would have been/was in preventing or ameliorating this casualty.

This change makes it easier for both human and computer users to distinguish the "objective" data from the more subjective.

Version 4.5 Version 4.5 released in April, 2010 greatly reduced the number of event codes which were over-lapping with causal factors. Much better separation of the occurence of an event from the causal factors surrounding the event. Result is more flexible and more scalable.

Version 4.6 Version 4.6 released in February, 2011 added lifeboat/raft and rescue event codes. Also split deaths/injuries into crew, passenger, by-stander, and attacker categories.

1.4 Acknowledgements

The CTX Casualty Database is dedicated to two men: Norman Hooke and Richard Cahill.

Norman Hooke took on the prodigous task of documenting all maritime casualties in which a ship was lost or declared a constructive total loss between 1967 and 1996. The result was 600 pages of small type with nothing but facts and more facts. Without Hooke's painstaking work many tanker casualties involving scores of deaths would already have been forgotten by an industry more than willing to quickly and quietly consign its dead to the deep. Without Hooke's work, the CDB would consist largely of tanker casualties which have caught the public's attention, mainly due to an oil spill on a high profile beach.

Captain Richard Cahill in his two seminal works *Collisions and their Causes* and *Strandings and their Causes* focuses on a small subset of Hooke's casualties. But for that subset he digs deep for the real cause. As the titles of the two books show, Cahill knew that collisions and groundings don't cause spills. In order to progress, we need to understand what caused the collision, what caused the grounding. And for several hundred casualties, he determined that cause despite in many cases the best efforts of owner, Class, and flag state. The CTX CDB uses Captain Cahill's definition of cause.

Needless to say, the CTX CDB borrows liberally from Hooke and Cahill. In fact, the CDB can be regarded in part as an effort to preserve a small portion of their work in a machine readable and analyzable form. The problem is we no longer have Mr. Hooke and Captain Cahill to do our work for us. We must find replacements.

The CDB owes a big debt to the Equasis organization (www.equasis.org). equasis is a ship database focussing on recording port state detentions. But, for currently and recently active ships, equasis has also taken on the very difficult task of maintaining the ship's owner, manager, class and flag history. Other than Class and flag at time of casualty, the CTX database makes no attempt to do this. If you need owner, manager stuff for a fairly recent casualty, you will probably find it by going to equasis with the ship's IMO number.[7]

Port states are big contributors to the CDB, most importantly, the USCG CGMIX and MSIS/MISLE databases, the Australian Maritime Safety, CEDRE, REMPEC, and the ACOPS summaries of UK spills. The Canadian Environmental Technology Center made a big contribution by preserving

[7] Perhaps in the future we can automate this process.

data on some 750 tanker spills between 1973 and 1997.

The idea of using an event sequence rather than categorizing spills as collisions, groundings, etc goes back at least to a 1983 report called List of Reported Serious Tanker Casualties by Shipping Information Services of Lloyds Register; and was used again by Psraftis et al in 1998, An Analysis of Marine Risk Factors.

The CDB also needs to thank Roger Haworth and the New Zealand Ship and Marine Society for maintaining the Miramar Ship Index, the most complete publicly available ship database, and the source of a good deal of the ship data in the CTX database. The CDB uses the Miramar Ship Index(MSI) ID as a unique ship key. The MSI ID is the ship's IMO number for those ships for which an IMO number has been assigned, but also handles ships which pre-date the IMO system.

1.5 Overview

The CTX Casualty Data base is in XML format. Since XML is non-proprietary, human readable and self-identifying, any one can examine the raw data files without any specialized software. There are no secrets.[8] And unlike a relational database, all the information on a casualty is localized in one place.

At the same time, the data is machine readable. Any XML aware computer can easily transform and access the data in a variety of ways. The interface on the CTX web site www.c4tx.org is a simple example. See also Appendix D.

And as we shall see, the hierarchical nature of XML meshes well with our goals which require us to record detail when we have it, but recognize that often we don't have or don't yet have, the detailed data we desire.

The CTX Casualty Database consists of

`ctx_core.xml` This file contains machine readable data on each casualty. Each casualty is keyed by as ID which is made up from the date of the casualty and an index number. For example, `19890324_001` is the first casualty in the data base with a date of `19890324`, `19800324_002` would be the second, etc. `ctx_core.xml` is described in Chapter 2. It is not only machine readable, but human readable as well.

The precis files For each casualty in `ctx_core.xml`, there is a `precis` file. The name of this file is the casualty ID, *YYYYMMDD_NNN*. This XML file contains free form text descriptions of the casualty abstracted from a variety of sources and/or links to those sources. The format of a `precis` file is described in Chapter 3

The pics files In some cases, the CTX has obtained photos or drawings related to a casualty. These images are filed in the casualty's *photo folder*. The name of each casualty's photo folded once again is the casualty ID. See Chapter 4.

The ship folders `ctx_core.xml` contains very little permanent ship data. The CTX CDB is designed to be used with any ship database which uses the ship's IMO number as a key. However, the CTX maintains its own ship database. For each ship involved in a CTX casualty, there is

[8] Don't worry. You don't have to know anything about XML to read this manual. The format is so obvious, that the simplest way to learn XML is just to look at an example, such as the CTX CDB.

a ship folder containing a description of that ship.[9] The name of this folder is *NNNNNNN* where *NNNNNNN* is the ship's IMO number. The CTX `ship` folders are described in Chapter 7.

Style sheets The CTX database contains a number of style sheets including the style sheets used to produce the web interface on the CTX web site. Theses style sheets which are written in XSL rely on a number of XML support files. For example, files for translating event sequence codes to a more long-winded description. These files are described in Appendix C.

Processing Scripts The database includes a number of scripts for doing statistical analysis, prepraring pretty tables, and the like. These scripts are written in Perl. These files are described in Appendix D. Appendix D not only allows interested parties to critique CTX's analysis methods, but also serves as easily modifiable templates, which can be customized by others to do whatever analysis they require.

Documentation Currently, the only documentation on the data base is this manual. But users may also want to peruse the CTX paper Uses and Abuses of Ship Casualty Data.

The CTX Casualty Database format is actually a sub-set of standard XML. This format obeys all the normal XML rules plus:

1. All the actual data is in attributes. The content of an element is either another element or an attribute. In particular, this implies no mixed content, which simplifies processing greatly.
2. Attributes are either free-form text or coded. A coded attribute in the core file, `ctx_core.xml`, cannot have the same name as another attribute, elsewhere in the core database file unless it is coded in exactly the same manner. In plain English, if it has the same name, it has the same meaning. `tod` (time of day) has exactly the same coding whether it is in a <collision> element or a <grounding> element. Put another way, you cannot reuse an attribute name say `type` to mean two different things in two different contexts.
3. In addition to the characters outlawed by XML (<, >, and &), free-form text may not include any double quotes. Use "and", "gt", and "lt" and "" instead.

[9] This is a goal which is not always realized. Processing software must be able to gracefully handle the situation where there is no ship folder for a particular ship.

Chapter 2

The ctx_core.xml File

2.1 The <casualty> element

ctx_core.xml consists of a outer <cdb> element enclosing a number of <casualty> elements, one casualty element for each casualty in the database.[1] So the overall document looks something like Figure 2.1.

The first line is XML boilerplate. Ignore it. The outer <cdb> has three attributes, all required.

version The data base version number.

update The GMT time the file was last modified in ISO8601 (short) format.

author The user ID of the person responsible for the last modification.

[1] In XML jargon, an *element* consists of an opening tag <*element_name*>, a closing tag < /*element_name*> and some stuff in between. The stuff in-between or *content* can be either be some text or another element. Elements can also have *attributes* which are included in the opening tag in the form *name*="*value*". In the CDB core file, all the actual data is in these self-identifying attributes. These attributes correspond to the fields or columns of a normal table.

If an XML element, say <tank>, consists only of attributes, you may use a bit of XML shorthand for the element's closing tag.

```
<tank code="3C" flood="Y">
</tank>
```

can be shortened to

```
<tank code="3C" flood="Y"/>
```

For our purposes, white space is almost always insignificant. The following two lines are the same as either of the forms above.

```
<tank   code="3C    "
flood="Y "></tank>
```

That's just about all there is to XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<cdb  version="4.5" update="20010417T143405" author="djw1">
...... lots of casualty elements .......
<casualty
  id="19721219_001" date="19721219" Edu="8"  site="Gulf of Oman          "
  locale="O"        tod="0400"      Acc="C"  lat=" 25.300" long="  57.567"
  coastal="OM"      area="in,WI,ARS,     "   weather="GD"         vis="GD"
  note="collision Gulf Oman, one Port to Port, one stbd to stbd          ">
  <event ec="VD" sid="2" Cause="N" Sure="5" detect_mi="16" rule="A"
         note="Perfect weather Gulf of Oman. Both detect at 14 plus miles.
               CPA 1 miles. Horta Barbosa assumes starboard to starboard.">
      <alter tod="    " course="322" spd="16W" helm="M" engine="FA"/>
      <alter tod="0358" course="322" spd="16W" helm=" " engine="FS"/>
  </event>
  <event ec="VD" sid="1" Cause="P" Sure="5" detect_mi="14" rule="A"
         note="At a distant of 1 to 2 miles, loaded Sea Star goes stbd">
      <alter tod="    " course="142" spd="16W" helm="S" engine="FA"/>
      <alter tod="0330" course="145" spd="16W" helm="S" engine="FA"/>
      <alter tod="0355" course="160" spd="16W" helm="S" engine="  "/>
  </event>
  <event ec="VL" sid="2" Cause="M" Sure="2"
         note="Horta Mate in chart room for 6 or 7 mins plotting"/>
  <event ec="CN" sid="1" cid="1" tod="0358" enc="H" talk="N" VHF="Y" GPS="M"
         nuc="N" impact="AS" spd="  " dop="+10" angle="90"
         note="Ballast HB maintains, expecting stbd to stbd"
  </event>
  <event ec="CN" sid="2" cid="1"
         nuc="N" impact="B"  spd="  "
         note="HB T-bones Sea Star forward of bridge"
  </event>
  <event ec="HL" sid="1" DS="N"  DB="N"
         note="'40 ft hole', photo of HB in dock says 10m+ penetration"/>
  <event ec="XT" sid="1" dead="12" hurt="8" Igs="M"
         note="Loaded, non-inerted Sea Star explodes, 12 killed, 8 burned"/>
  <event ec="X_" sid="2" hurt="-2" IGS="M"
         note="Fire spreads to Horta which is abandoned without loss."/>
  <event ec="SK" sid="1" date="19721224" tod="1045" vol="141100000" mat="C"
         note="More explosions, Sea Star sinks. Horta repaired"/>
  <ship sid="1"  imo="6829721"  class="  "  name="sea star"
        st="TC"  dwt="120300"   yob="1968"  flag="KR" tnks="
        ht="SP   grt=" 63989"   status="L"  cgo="PC"  sts="N"
        ns="1"   crew="39"      ig="N"      pob="N">
  </ship>
  <ship sid="2"  imo="7001261"  class="  "  name="horta barbosa"
        st="TC"  dwt="116750"   yob="1969"  flag="BR" tnks="  "
        ht="SP"  grt=" 62619"   status="B"  cgo="  "  sts="N"
        ns="1"   crew="36"      ig="N"      pob="N">
  </ship>
</casualty>
...... lots more casualty elements .......
</cdb>
```

Figure 2.1: Make up of the Core File

2.2 The Casualty Common Fields

Each <casualty> element begins with a number of fields which are common to all events and ships in the casualty. These fields are always required in every casualty; but, unless the description says otherwise, they may be blank. All non-freeform fields must be coded exactly as show below. The codes are case sensitive. `Y` is not the same as `y`.

id Local date in *yyyymmdd* plus an index number. `19721219_001` is the first casualty in the database with a date of `19721219`. `19721219_002` would be the second and so on. This is the primary key for this casualty. `id` is immutable. If we discover that the date we used in the ID was incorrect, the `date` field (see below) will change, but not the `id`. This field is always required, and it must be unique.

date Local date on which the casualty occurred in *yyyymmdd* form. This should be the date of the casualty's primary cause event (see below). If day of month not known, use `00`. If you do not know month, comment out the casualty until you do. Always required; may not be blank.

should we not relax this?

Edu The CTX feels strongly that, while ***carefully done*** overall statistical analyses can yield useful insights, one can learn at least as much about ship casualties by studying a representative sample of individual casualties in detail. That is one reason why each casualty in the database is supported by a *precis* file, and, where available, photos, charts, and drawings. Unfortunately, we have a reasonably complete story for only a small minority of our casualties. The `Edu` field is intended to be in aid in using the precis files. It ranks all our casualties on a scale of 1 to 5 by how complete and instructive our information is on the casualty. 1 means we have essentially no useful data on the casualty. 5 means we think we have the full story and it is extremely instructive. `Edu` is coded as follows:

5 Extremely Instructive
4 High Educational Value
3 Modest Educational Value
2 Little Educational Value
1 Nil Educational Value

A low `Edu` does not mean the casualty was unimportant.

Anyone seriously interested in ship safety should study all the casualties with an `Edu` of 7 or higher individually.

site Free-form text describing location of casualty. Can include status, eg "anchored Khor Fakkan" No more than 22 characters,

acc Position accuracy grade. The CTX philosophy is that it is better to put in an approximate position rather than none at all. However, we need to keep track of the accuracy or lack of same of our estimate of position. This field is coded as follows.

A	Data is thought to be accurate to tenths of a minute.
B	Data is thought to be accurate to one-minute.
C	We have real location data, but could be approximate (+/-10 minutes).
D	We have only ctx guess, could be off (+/-0.5 degrees).
E	We have only ctx wild guess, could be off (+/-2 degrees)
F	Could not even make a guess, position field is blank.

Unless the position accuracy grade is A or B, the data should not be relied on to accurately locate the casualty. The other grades are aimed at applications which wish to do regional or area-wide analyses. Mapping applications must distinguish between the accuracy grades and display inaccurate positions (if at all) in a manner that warns the user of the level of accuracy.

The fact that all positions are shown to a thousanth of a degree has absolutely no implications for accuracy. All applications must combine the `lat` and `long` fields with this position accuracy grade in displaying this data.

Mapping applications should be aware that in many cases all we know about the location of a casualty is that it occurred in a particular port or body of water, e.g. Milford Haven. Such casualties will usually be assigned a position which is the nominal location of the area according to a maritime atlas. In other words, all these (for example) Milford Haven casualties will erroneously appear to have occurred at exactly the same place. Often, in such cases, the position accuracy grade will be C.

If you do not code a `lat` and `long` for the casualty, then mapping applications cannot display a token for the casualty, which could be misleading for users, who use a map display to pick out casualties.

lat Estimate of latitude of casualty. Otherwise blank. This should be the position where the primary cause occurred, not the position where the ship subsequently sank or grounded. But sometimes the data forces us to use the latter as a proxy for the former.

The coding is decimal degrees with three digits to the right of the decimal point. North latitude is positive, South is negative.

long Estimate of longitude of casualty. Otherwise blank. The coding is decimal degrees with three digits to the right of the decimal point. East longitude is positive, West is negative.

locale The kind of water the ship was in at the time of casualty. This field may be

D	At repair yard, including mooring/unmooring
T	At fixed berth, including mooring/unmooring
S	At SBM, including mooring/unmooring
L	Lightering, including mooring/unmooring lighter
H	Harbor or Harbor entrance, anything inside the Seabuoy
R	Restricted waters
O	Open water
	Dont know

Use the most specific locale appropriate. Canals and rivers are an H locale. Restricted waters should be interpreted broadly. The entire English Channel is restricted waters. Certainly, any TSS area is restricted waters.

area This field allows casualties to be grouped geographically in a number of ways. See Section 2.2.1

tod Local time of day when casualty occurred which may be

hh	24 hrs
hhmm	24 hrs, minutes
hhmmss	24 hrs, minutes, seconds
NGHT	Night
DAY	Day
DUSK	Dusk
DAWN	Dawn
NOON	Near midday
	Dont know

Use the *hhmm* form if you have the data. Ideally this should be the time of the primary cause (see below). The *hhmmss* form is almost never used in the <casualty> element, but is common in the more detailed sub-elements,

weather Weather at time of casualty, coded as follows:

Bn	Beaufort force $n <= 9$ if known
10	Beaufort 10
11	Beaufort 11
12	Beaufort 12
BD	bad but no other info
CM	Calm but not foggy
FG	Foggy
GD	Good
GL	Gale
HW	High Winds
RN	Rain
TH	Thunderstorms
TY	Hurricane/typhoon
	Dont know

vis Visability at time of casualty, coded as follows.

An	About $1 <= n <= 9$ miles
L1	Less than 1 mile
M9	9 plus miles
GD	Good, but no other info
BD	Bad (3 mi or less) but no other info
VB	Very Bad (1 mi or less), but no other info
	No info.

coastal The coastal state most affected by the casualty. Use the ISO-3166 two character code, all caps. If two countries are equally affected, lean toward the country that does the best job of investigating the casualty. Every casualty anywhere near land should have a coastal state. If a casualty is mid-way between two or more coastal states, pick one. Processing software interested in a particular coastal state is expected to be smart enough to check neigboring states.

note Freeform text describing the casualty as descriptive as possible. Many applications need a one-line summary of the casualty. This string should be no more than 64 characters. Applications such as the web site interface are allowed to truncate this string to 64 characters. Anything longer than 64 characters will not be seen by most users. If this is a multi-ship casualty, try to get the names of the second, etc ships into this field. Some applications which show only one line per casualty have no way of displaying the other ship names. If you have room, you can repeat the location info in `site`. Some applications may display only this element. Remember to replace any double quotes with single

quotes. You may use the following abbreviations, but only if necessary to make the text fit.

ER	Engine room	bdh	Bulkhead
PR	Pump Room	cgo	Cargo
FP	Forepeak	iwo	in way of
FO	Fuel oil		
fwd	Forward		
stbd	Starboard		

Just about any element in the CDB can have a `note` field. But be aware that any information contained in these fields is unavailable to statistical analyses. The most an application can do is display these fields. For data to be machine readable, it must be in one of the non-free form fields.

2.2.1 The area field

Users need to group casualties geographically for a wide range of reasons. In some cases, the user is interested in very broad areas, for example, the entire North Atlantic Ocean. In other cases, the interest is limited to a particular port. In still others, he may be interested in a particular body of water or perhaps a canal. Mapping applications, inter alia, have to operate at these various levels in a manner that ensures that all casualties within a desired region are displayed including those whose location is specified at a lower level of detail. Thus, if a user asks to see all casualties that occurred in the Gulf of Mexico, then the resulting casualties should include casualties that occurred in Galveston Bay. Finally, the mapping application must be able to repond to the user's requirements with a minimum of computational effort. The `area` field attempts to meet these requirements; but in so doing it is one of the more complicated fields in the data base.

The format for the `area` field is *ee*,*rr*,*sss*,*ppppp* where *ee* is a two character empire code, *rr* is a two character region code, *sss* is a three character sub-region code, *ppppp* is a five character place code (usually a port). The commas must always be coded, even if a sub-field is blank.

All the known sub-codes must be explicitly coded. For example, if a casualty occured at Valdez, the area field must be coded: `area="np,UW,PWS,USVDZ"`. Do not assume that processing software knows that the place Valdez is in Prince William Sound sub-region, in the eastern North Pacific region, in the North Pacific empire.[2]

This restriction together with the explicit coding allows mapping applications to respond to a user's geographical query with almost no computation.[3]

Do not confuse the `area` field with the `coastal` field. The `area` field is aimed at identifying a body of water, not the country where the casualty occurred. If is quite common for a coastal state, to be bounded by multiple `area` codes.

[2] There are two reasons for this. a) The data structure, as we shall see, is not a tree. b) This "redundancy" saves lots of computational effort.

[3] The algorithm is: if the query code, xx, is two char lower case, search area field for xx,: if the query code is two char upper case, search area field of ,xx,: if the query code is three char, search area field of ,xxx,: if the query code is five char, search the area field for ,xxxxx. Note the use of the comma(s).

The Empire codes

There are seven empire codes:

Table 2.1: Empires

na	North Atlantic empire	North Atlantic ocean inclusive
sa	South Atlantic empire	South Atlantic ocean inclusive
np	North Pacific empire	North Pacific ocean inclusive
ss	Seven Seas empire	Oceania
sp	South Pacific empire	South Pacific ocean inclusive
in	Indian empire	Indian Ocean inclusive
me	Mediterranean empire	Mediterranean Sea inclusive
ar	Arctic empire	Arctic Ocean inclusive

The empire codes are mutually exclusive and exhaustive. ***Every casualty for which we have any location data must be assigned an empire code.*** Everything that is connected to the ocean is part of that empire.

The North Atlantic empire includes Baltic, Caribbean Seas, the Gulf of Mexico, the Gulf of St Lawrence, and all the rivers that drain into these bodies of water including the Great Lakes. It excludes Baffin Bay and the Barents Sea.

The Indian empire includes the Red Sea, Straits of Malacca (but see below)). South of Australia it extends east to the Bass Strait. It excludes the Seven Seas.

The North Pacific empire the East China Sea, the Yellow Sea, the Sea of Okhotsh, but not the South China Sea. It extends south to the equator and north to the Bering Strait.

The Seven Seas empire includes the South China Sea (including all but the west end of Malacca) Sulu Sea, Celebes Sea, Java Sea, Banda Sea, Arafura Sea, Timor Sea, but not the Coral Sea. It extends east to the Torres Strait and north to the Formosa Strait. It includes all Philippine inland waters, but not the east coast of the Philippines.

The South Pacific empire includes the Coral Sea, the Tasman Sea, the Solomon Sea, and the Bismark Sea. It extends east to the West Coast of South America and north to the equator. South of Australia it extends west to include the Bass Strait.

The Mediterranean empire includes the Black Sea and even the Caspian Sea.

The Arctic empire includes the Bering Strait, Baffin Bay, and the Barents Sea.

The Region codes

The region codes are

Table 2.2: Regions

NP	Northwest Pacific	Japan, Taiwan, Phil. east to Int. Date Line (IDL)
SJ	Sea of Japan	incl Inland Sea, so coast of Korea
YS	Yellow Sea	incl E China Sea, Formosa Strait
OK	Sea of Okhotsk	
UW	Northeast Pacific	IDL east to N. America, south to Columbia
CS	South China Sea	including Gulf of Thailand, most of Malacca
SP	South Pacific	New Zealand, Solomon Is. east to IDL
SU	Sulu	Sulu, Celebes Sea, intra Philippine waters
JA	Java	Java, Banda, Arafurua, Timor seas,
WS	Southeast Pacific	east of IDL, north to Panama
TS	Tasman Sea	and Coral, Solomon Seas, Bass Strait
EI	East Indian Ocean	east of Sri Lanka to westend Malacca, Pt Phillip B.
WI	West Indian ocean	west of Sri lanka incl Gulf of Aden
PG	Persian Gulf	including Straits of Hormuz
RS	Red Sea	including most of Suez Canal (see below)
WE	Northeast Atlantic	including, Biscay, Channel, Irish, North Sea
UE	Northwest Atlantic	west of Greenland
BL	Baltic Sea	including Kattegat, Belts, Gulfs of Bothnia, Finland
CA	Caribbean Sea	including most of Panama Canal
GM	Gulf of Mexico	including FL Str to 25N, Old Bahama Channel
SL	Gulf of St Lawrence	including Cabot Channel
GL	Great Lakes	including Seaway
ES	Western South Atlantic	north to Trinidad, west of 20W
WA	Eastern South Atlantic	north to Gibraltar,east of 20W
EM	East Med	including east coast of Sicily s of 38N, Malta
WM	West Med	including Sicilian Channel. Strait of of Messina
BS	Black Sea	incl Bosporus, Marmora, (most of) Dardanelles
CP	Caspian Sea	
AR	Arctic	incl Bering Strait, Baffin Bay, Barents Sea

The regions are mutually exclusive and exhaustive. Every casualty for which we have any location data should be assigned a region code. Except as noted above rivers are assigned to the region they drain into.

A region need not lie entirely within a single empire.[4] The `WA` region

[4] This means the data structure is not a tree.

extends north to Gibraltar. The `ES` region extends north to Trinidad. The `WS` region extends north to Panama.

The region codes separate major bodies of water from the open oceans. In the CDB system, the Gulf of St Lawrence region is not part of the Northwest Atlantic region. Applications which prefer a broader definition of ocean will have to base it on the empire codes.

The Sub-region codes

The sub-region codes are

Table 2.3: Sub-Regions

ADN	Gulf of Aden	Socotra (55E) west
ADR	Adriatic Sea	
AEG	Aegean Sea	
ARS	Arabian Sea	including Gulf of Oman
BIS	Bay of Biscay	
BDM	Bosporus, Dardanelles	incl sea of Marmora, BS except S end
CKI	Cook Inlet	
COR	Columbia River	
DEL	Delaware Bay/River	including Big Stone anchorage
EAF	East Africa coast	south of Socotra
ECH	English Channel	including Straits of Dover
FLS	Straits of Florida	to 25 N, incl Old Bahama Channel
GAL	Galveston bay	including Houston Ship Channel
IRS	Irish Sea	incl St Georges Channel, all west coast of GB
KAT	Kattegat	including Orasund, Belts
KLC	Kiel Canal	Baltic except for west end
LMA	Lake Maracaibo	
MSR	Mississippi River	
SME	Straits of Messina	
NOS	North Sea	including Skagerrak, West coast of Norway
NYH	New York Hbr/Bay	including Hudson River
ORR	Orinoco River	
PAC	Panama Canal	CA except Balboa end (see below)
PUS	Puget Sound	incl Juan de Fuca, Rosario straits
PWS	Prince William Sound	
RPL	Rio Plata	
SZC	Suez Canal	RS except Port Said (see below)
SGB	Straits of Gibraltar	EM except west end
SHM	Straits of Hormuz	PG except south end
SFB	San Francisco Bay	
SMA	Malacca Straits	CS except for west end
SMN	Straits of Magellan	ES except for west end
SNR	Sabine/Neches River	
TKO	Tokyo bay	

The sub-regions are mutually exclusive but not exhaustive. The sub-region code may be three blanks (followed by a comma), which means either "don't know" or "none of the above". More sub-regions may be added later.

By default, straits, canals, etc connecting two regions are assigned to the smaller of the connected regions. Thus, the default region for the Suez Canal is the Red Sea. ***But judgement must be used.*** A casualty at Port Said at the north end of the Suez is both a Suez Canal casualty and an East Med casualty. A casualty at Balboa at the Pacific end of the Panama Canal

is both a Panama Canal and a Northeast Pacific casualty.[5] If a casualty occurs at the entrance or the approaches to a sub-region, then it should be assigned to that sub-region.

[5] This means the data structure is not a tree.

The Place codes

For the CDB the place codes are normally a port. The place codes are based on the UN LOCODES. This is a five character code assigned by the UNECE. They can be found at http://www.unece.org/cefact/locode/service/location.htm. The first two characters are the ISO 3166 country code of the port, the last three characters are unique for each country. The place code may be five blanks. However, a broad definition of port should be used. if a casualty occurs in the entrance or approaches to a port, then the casualty should be assigned to that port.

It is usually better to code the overall port rather than an individual terminal. Use the code for Singapore, rather than the more specific codes. And interpret the extent of the port broadly. Use Los Angeles for Long Beach, El Segundo, etc.

In a few cases, there appears to be no LOCODE for a place we need, in which case the CTX has made up a (hopefully temporary) code. These are distinguished from the real codes by being lower case. The LOCODES for some important tanker and bulk carrier ports including the CTX additions are shown in Table 2.4.

If the port code you need is not in Table 2.4, then you will need to add it to the `labels/area.xml` file. See Chapter B. Otherwise searches on this port code will fail in applications that use the `area` labels.

Table 2.4: UNECE Location Codes

AEFJR	Fujairah, Khor Fakkan
AUBWE	Brisbane River, Moreton bay
BMBDA	Bermuda
BRRIO	Rio de Janiero
BRSSO	Sao Sebastiao
BSFPO	Freeport, Bah. incl S Riding Point
CNTAO	Qingdao
DZAZR	Arzew
DZSKI	Skikda
EETLL	Tallinn
ESLCG	La Corunna
GBFLH	Flotta
GBIMM	Immingham
GBMLF	Milford Haven
GBSUL	Sullom Voe
GBTTL	Tetney SBM
GRAGT	Agioi Theodoroi
IEBTH	Bantry Bay
INBOM	Bombay
IRKHK	Kharg Island
ITCAX	Genoa????
KRYUS	Yosu
LTBOT	Butinge SBM
LTKLJ	Klapeida
LVVNT	Ventspils
NLRTM	Rotterdam
PKKHI	Karachi
PRGUY	Guayanilla, PR
PTLEI	Oporto, Leixos
ROCND	Constantza
RUnov	Novorossiysk
SARTA	Ras Tanura inc Juaymah
SGSIN	Singapore
THmat	Mataphut
USBOS	Boston Harbor
USCRP	Corpus Christi
USgla	Galveston Lightering Area
USLAX	Los Angeles incl Long Beach, Huntington
USloo	LOOP
USNAX	Barbers Point
USVDZ	Valdez
VILIB	St. Croix (HOVIC)
ZACPT	Cape Town
ZADUR	Durban
ZASDB	Saldanha Bay

2.3 The Casualty Sub-Elements

As Figure 2.1 shows, in each <casualty> element, the casualty common fields are followed by

1. A series of <event> sub-elements which can describe the casualty in some detail. [6]
2. One or more <ship> sub-elements which identify each ship involved in the casualty. <ship> in this context is a bit of a misnomer, for a <ship> can be a terminal, an offshore platform, or any other man-made object that is involved in an allision or terminal casualty. It is not necessary for a casualty to involve a collision or allision to have multiple ships. In December 2007, the ISI Olive had a steering gear failure at the south end of the Suez Canal. The ISI Olive went aground as did the next ship in the convoy, the Overseas Meridian, attempting to avoid her. The two groundings are part of a single casualty with the same cause, and are treated as such in the CDB. Another example is a fire on one ship spreading to another.

The key to linking <event>s to <ship>s is the `sid` field.

- Each <event> element must contain an `sid` field which must match the `sid` field in the corresponding ship's <ship> element.

A fundamental restriction on an <event> element is that it must have one ***and only one*** `sid` field. This means that events such as collisions which involve more than one ship must be represented by two or more <event> elements. These elements are linked by a `cid` field.

The casualty shown in Figure 2.1 involves two ships and has been described with nine events. The first three events record what little we know about what happened on each bridge before the collision. The fourth and fifth event record the collision itself. Notice how the `sid` field allows both a human and a computer to keep track of which ship we are talking about. The `cid` field informs us (and the computer) that the fourth and fifth events are really the same event viewed from each ship's perspective.

We will see how this system works in detail below. For now, all you need to keep in mind is that the `sid` field is the link between events and ships.

The final thing to notice about Figure 2.1 is that you don't need to read this manual to have a pretty good idea what happened to the Sea Star and the Horta Barbosa. In fact anyone with a background in the industry can figure out what almost all these fields are without referring to the manual.

[6] Some of the fields in the <event> sub-elements can over-ride the same named fields in the casualty's common block. In Figure 2.1, we use this capability to record the date and time of the Sea Star's sinking.

The data is nearly transparent to and easily auditable by a human, once you learn to zone out the <, >, and quotes clutter. [7] At the same time, as we shall see, any XML aware computer can extract and aggregate the non-`note` data as required to answer statistical queries.

[7] Figure 2.1 is the raw data. If you display the CDB core data file in just about any browser or general purpose editor, and search for "sea star", you will see a close replica of Figure 2.1. Unlike a relational database, no special software is required to examine the data. And all the data on a particular casualty is localized in one place.

2.4 The Major Event Sequence

2.4.1 The Event Element

The major event sequence is the very heart of the CTX Casualty Database. It consists of one or more <event> elements. Here is an example event sequence:

```
<event ec="Vu" sid="1" detect_mi="Y"
       note="Fogo, Trentbank both on 288, Trentbank faster">
   <alter tod="    " course="288" spd="11W" helm="M" engine="HA"/>
</event>
<event ec="VO" sid="2" Cause="N" Sure="5" detect_mi="Y" rule="G"
       note="Trentbank overtaking Fogo too close on Fogo port side">
   <alter tod="    " course="288" spd="14.5W" helm="M" engine="FA"/>
   <alter tod="1555" course="288" spd="14.5W" helm="M" engine="FA"
          note="CO relieves 2M, sees Fogo 2.5 to 3 miles away"/>
</event>
<event ec="MR" sid="2" Cause="P" Sure="5"
       note="Trentbank steering failed just after passing Fogo">
    <component er="M" sfi="403" name="steering gear"
       note="'several recent failures of electric SG'">
    </component>
</event>
<event ec="DS" sid="2" lop_hrs="m" TS="M"
       note="Single screw, no redundancy, lost steerage, turned to stbd"/>
<event ec="Vu" sid="1" detect_mi="Y" rule=" "
       note="Fogo tried to go stbd at end, but no time to alter course">
   <alter tod="    " course="288" spd="11W" helm="s" engine="HA"/>
</event>
<event ec="CN" sid="1" cid="1" nuc="N" impact="B"  spd="11W"
       enc="V" talk="N" VTS="N" course="010"
       note="Fogo hit Trentbank stbd side midships"/>
<event ec="CN" sid="2" cid="1" nuc="Y" impact="MS" spd=""
       note=""/>
<event ec="HL" sid="2" dead="1" DS="N" DB="N"
      note="Trentbank Engine room flooded. Also water in No 4 hold.">
     <tank code="ER" flood="Y"/>
     <tank code="H4" flood="Y"/>
</event>
<event ec="SK" sid="2" date="19640924" vol="-1" mat="B"
      note="Trentbank towed to Alexandria, anchored, sank"/>
```

Figure 2.2: Fogo/Trentbank Collision Event Sequence

Every casualty must have at least one <event> element. The events must be in temporal sequence (if known). A casualty may have as many

major events as needed. But the operative word here is *major*. Often a major event can be broken down into a series of sub-events. For example, a machinery failure usually results from a sequence of machinery component problems. Such sequences are best recorded in sub-elements. See Figure 2.4 for an example. Similarly, an <event> may contain a series of course or speed changes in <alter> sub-elements as in Figure 2.2 above.

2.4.2 Required Event Fields

The Event Code

Every <event> must have exactly one event code,ec. The event code is a two character field which specifies the type of event. Table 2.5 lists the current event codes. Many events can either be a cause or a consequence. However, some event codes by fiat cannot be causal. ***In order to be causal, an event must be an error, a defect, or a failure.*** The specific event codes are described in detail in the following sections. In this section we will describe the fields which are common to all, or at least many, event codes.

Table 2.5: Event Codes as of 2010-04-17T11:42:05

Code	Label	Can be Causal?
AP	Piracy	Y
AW	War Attack	Y
Ca	Allision	N
Cm	Near-miss allision	N
CA	Hit while stationary	N
CG	Tug contact, other tug screw up	N
CI	Interaction/Wake	N
CM	Near-miss Collision	N
CN	Collision	N
DA	Disabled, anchored	N
DC	Internal cargo transfer	N
DE	Escort tug used	N
DF	Counterflooded/ballasted	N
DL	Lightered/lightened	N
Dl	Listed, no self-recovery	N
DP	Total loss of power	Y
Dp	Power restored.	N
DR	Refloated	N
Dr	Massive roll, recovered	N
DS	Loss of steerage	Y
Ds	Steerage restored	N
DT	taken under tow.	N
Dt	Tow lost/slipped.	N
EB	Navaid out of position, inoperative	Y
EC	Charts incorrect, not ship/owner fault	Y

Continued on next page

Code	Label	Can be Causal?
ED	Channel depth not as charted	Y
ER	Coastal state refused refuge	Y
EA	Coastal state provided refuge	N
Eb	Coastal/port state blacklisted	N
Ed	Coastal/port state detained	N
ES	Hit unmarked submerged object	Y
ET	External Tampering	Y
E_	Other/unknown external error	Y
FE	Ignition upon Entry	Y
FH	Ignition by Hotwork	Y
FL	Lightning strike	Y
F_	Other/unknown fire ignition event	Y
GA	Anchor dragged	N
GB	Hit berth	Y
GC	Conning error	Y
Gc	Classification society change	N
GD	Ship too deep for depth, swell, channel	Y
GE	Engine department error	Y
GR	Bad Routing by service	Y
GS	Bad seamanship, deck	Y
Gs	Classification society survey	N
G_	Other/unknown guidance error	Y
HC	Minor hull failure, crack	Y
HD	Deck Vents/Equipment Failed	Y
HF	Hull failure	Y
HL	Holed	N
HP	Cargo pipe failure/leak	Y
IJ	Jettisoned cargo/bunkers	Y
IN	Intentional Discharge	Y
IS	Intentionally Sunk	Y
Lb	Lifevessel burned	Y
Lc	Lifevessel capsized	Y
Le	Lifevessel engine failure	Y
LF	Lifevessel found	N
LL	Lifevessel successfully launched	N
Ll	Lifevessel could not be launched	Y
Ls	Lifevessel swamped/sunk	Y
L_	Lifevessel other/unknown	Y

Continued on next page

Code	Label	Can be Causal?
MA	Unable to slow down	Y
MB	Blackout	Y
MD	Deck machinery failure	Y
ME	Main Engine failure	Y
MF	Engine room flooding	N
Ml	Lifeboat failure/problem	Y
MO	Fuel/lube/hyd pipe leak	Y
MP	Propeller failure/damage	Y
MR	Steering gear/rudder failure	Y
MS	Shaft/sterntube failure	Y
MT	Stern tube leak	Y
MW	Sea water line leak	Y
MX	Boiler failure/fire	Y
M_	Other/unknown machinery	Y
NA	Navigation error	Y
NC	Bad/missing charts on-board	Y
NS	Hit marked submerged object	Y
RB	Lifeboat rescue attempt	N
RH	Helicopter rescue attempt	N
RS	Ship rescue attempt	N
St	Resumed trading	N
Sr	Repaired	N
SC	Contructive Total Loss	N
SD	Scuttled	N
SK	Foundered/sank	N
SP	Capsized	N
ST	Scrapped	N
TU	Unmoored by weather	Y
TD	Deballasting screw up	Y
TH	Hose break/leak	Y
TL	Incorrect loading	Y
TP	Tank over/under-pressure	Y
TO	Tank overflow	Y
TS	Cargo shifted	Y
TW	Unmoored by wake	N
T_	Other/unknown transfer screw up	Y
U_	Primary Cause Unknown	Y
VA	Anchored	N

Continued on next page

Code	Label	Can be Causal?
Va	Anchored	N
VB	Burdened vessel failed to manuever	Y
VD	Dance of death, one went port, other stbd	Y
VL	Failed to detect other vessel	Y
VO	Overtaking too close	Y
VR	Rogue vessel in wrong lane	Y
VU	Uncoordinated manuever	Y
Vu	Non-causal manuever	N
V_	Other/unknown manuever	Y
WS	Grounding	N
Ws	Near Miss Grounding	N
XA	Accommodations Fire	N
XE	Engine Room Fire/Explosion	N
XP	Pump Room Fire/Explosion	N
XT	Cargo Tank Fire/Explosion	N
X_	Fire, explosion, cant say where	N

The Ship ID

Each <event> element must have an `sid` field. The `sid` is a small positive number. The value of the `sid` field must match the value of the `sid` field in the casualty's corresponding <ship> element. The `sid` field need by unique only within the casualty. Even if a casualty only involves one ship, all the events must have an `sid` field. By convention, the value of this field in single ship casualties is simply `1`.

The Collision ID

If an <event> is part of a multi-ship event, such as a collision, then all the <event>'s that make up the multi-ship element must have a `cid` whose value is the same. The `cid` is a small positive number. The value of the `cid` field need by unique only within the casualty. `cid` fields were developed for collision, but they can be used to join other sorts of events as well. For example, you might link two or more attack events to record the fact that these ships were damaged in the same overall attack.

The Cause Flag

If an event is actually causal, then its <element> must have a `Cause` flag. The cause flag is coded as follows:

P	Primary Cause
N	Necessary Cause
S	Secondary Cause
	Non-causal

Each casualty must have ***exactly one*** *primary* cause. Applications which analyse casualties on the basis of a single cause will use this event. Picking the primary cause is often the most difficult task facing the coder. If you don't have enough information to identify a primary cause, then the casualty must have an event whose event code is `U_` (Primary Cause Unknown) and this event must have a `cause` field whose value is `P`. Do NOT use `U_` if you cannot decide which of several necessary causes to call primary. Pick one. Intelligent analyses will treat both Primary and Necessary causes as pretty much the same.

should we do away with primary cause?

A *necessary* cause is an ***error*** or ***defect*** or ***failure*** which, if it had not occurred, then with high probability the casualty would not have occurred. A casualty may have zero or more necessary causes. CTX recommends that applications treat primary and necessary causes on the same footing. The primary cause is simply the first among equals necessary cause.

A *secondary* cause is an ***error*** or ***defect*** or ***failure*** which, if it had not occurred, then with high probability the effects of the casualty would have been mitigated, but not prevented. A common secondary cause is poor or non-existent inerting. A casualty may have zero or more secondary causes.

If an <event> has no or blank `Cause`, then that event will be treated as a pure consequence.

The primary cause will often be the first event in the sequence, but this is not necessarily the case. Consider the Trentbank/Fogo collision in Figure 2.2. The Trentbank was overtaking the Fogo with a separation that was much smaller than necessary or prudent. Just after passing, the Trentbank's steering gear failed, she turned into the path of the Fogo, was hit midships, and eventually sank. In this case, CTX called the steering gear failure the primary cause and the preceeding seamanship error a necessary cause.

In fact, there is no requirement that the first event in the event sequence be causal although this will often be the case.

Since the CTX CDB supports multiple causality, one must be careful in generating statistics to avoid double counting. Suppose your interest is in casualties in which either structural failure or machinery failure was causal. You can't ask for all structural failures and then ask for all machinery failures, and add the resulting numbers. The query has to check each casualty for either type of cause and then count the casualty once. In fact, since a casualty can have multiple causal events of the same cause category, you

need to be careful to avoid double counting even if you are dealing with a single cause category. See Appendix D for an example.

Notice that the `Cause` field is capitalized. The CDB divides all its machine readable fields into three categories:

Facts Data which in theory requires no subjective judgement. These fields all have all lower case names.

Causal Assignment of cause and causal factors. These fields necessarily require some subjective judgement. Such fields have names which start with a capital letter.

RCO Fields which make a judgement about how effective a particular RCO would have been/was in preventing or ameliorating this casualty. These fields require still more subjective judgement. Such fields have all upper case names

The `Sure` field.

If an event is a primary or secondary cause, then it must have a `Sure` field. The `Sure` field rates the quality of the data for this cause. The `Sure` rating is coded as follows:

0 Nil cause info. Use this with the cause Unknown code.

1 Situational data only, no scenario clearly dominant; e.g. all we know is ship suddenly turned into another. In this case, a steering gear failure is a decent bet, but it would be a stretch to call this cause highly likely. A common use of `Sure=1` is when we have an Engine Room fire or Engine Room flooding but no other information. In this case, CTX will normally use the `M_` causal event code, Unknown Machinery Failure, and set `Sure` to `1`.

2 Situational data only, but one scenario clearly dominant. e.g. ship sank in heavy weather, no survivors, no distress signal. In this case, structural failure is highly likely a cause even though we have no direct evidence.

3 Circumstantial evidence only, e.g. tank blew up, repair worker killed. Use this also for situations where we have conflicting or uncertain evidence; but we are willing to make a judgement based on the preponderance of the evidence.

4 Solid evidence or creditable eye witness testimony.

5 Beyond reasonable doubt, usually as a result of a court case or good port state investigation.

`Sure="1"` allows us to go out on the limb a little and provisionally assign a cause to casualties we would otherwise have to call Unknown. Conservative analyses might want to treat `1` and `0` the same. A super-conservative analysis might be based only on a `Sure` of 4 or more. But that would mean throwing away a tremendous amount of valuable information.

The `Sure` field refers to the quality of the evidence for a particular cause. The `Sure` field does NOT rate the Primary versus Necessary cause judgement. In the Trentbank/Fogo, we have excellent evidence for both causes, but someone who feels that operational errors are always Primary might flip the Primary/Necessary coding.

This example illustrates another role of the `Cause` field. The assignment of cause is necessarily a subjective judgement, especially the assignment of the primary cause. The `Cause` field separates these judgements from the other data which is hopefully less subjective. As a result, the CDB database can be used by people who don't necessarily agree with the coder's cause assignments.

2.4.3 Common Event Fields

In addition to the required fields, all <event> elements may have the following fields.

date The date of the event in *yyyymmdd* form. If an event has no `date` field, processors may assume the last preceding date in the event sequence; or, if there is none, the `date` in the <casualty> element.

tod The local time of day of the event coded as for `tod` in the <casualty> element. If an event has no `tod` field, processors may assume the last preceding time of day in the event sequence; or, if there is none, the `tod` in the <casualty> element.

lat The latitude of the event coded as for `lat` in the <casualty> element. If there is no `lat` field, processors may use the `lat` in the <casualty> element.

long The longitude of the event coded as for `long` in the <casualty> element. If there is no `long` field, processors may use the `long` in the <casualty> element.

dead The CTX CDB divides deaths and injuries into four categories: crew, passengers, attackers, and by-standers. Crew includes riding crews and other workers, pilots, surveyors, superintendents, inspectors, and crew family. Passengers are limited to people who paid for a passage. By-standers are deaths/injuries to people who were not on-board any of the casualty's ships, usually on-land. If you don't know what category a death/injury falls into, call it crew.

`dead` is crew killed in this event on this ship including missing. If the number is positive, it means we know at least this many crew were killed. If the number is -1, it means we know there were crew deaths, but we have no estimate on how many. If the number is -2, it means we suspect there were deaths, but we aren't sure. If the number is -3, it means we know there were no deaths. Use this only if the fact that there were no crew deaths is surprising. In the absence of any `dead` fields in a casualty, applications should assume there were no crew fatalities. If the number is -4, it means the people killed on this ship was counted in the dead on the first ship. Sometimes we only know the total number killed in a casualty. This code handles this case.

Zero or blanks are not allowed. This coding simplifies sorts and queries; but applications should convert negative numbers to meaningful strings for display.

Fatalities should be assigned to the event when the deaths occurred. If the fatalities occurred when the ship sank, put the dead field in the corresponding event. ***This does NOT imply any causality.*** Causality is determined by the `cause` flag(s).

Often we don't know when the deaths occurred. Don't worry about it. Just make your best guess. The timing of the deaths is much less important than the cause.

dead_pax Passenger fatalities on this ship including missing. This field is coded in the standard manner: a non-negative number.

dead_nearby By-stander fatalities caused by this ship. This includes deaths of on-land terminal personnel. Coded in the same manner as `dead_pax`

dead_perp Attacker fatalities on this ship. Coded in the same manner as `dead_pax` `dead_perp` is normally not included in overall death totals.

hurt Crew seriously hurt but not killed in this event on this ship. Coded in the same manner as `dead`.

hurt_pax Passenger serious injuries on this ship. Coded in the same manner as `dead_pax`

hurt_nearby By-stander serious injuries caused by this ship. Coded in the same manner as `hurt_pax`

hurt_perp Attacker serious injuries on this ship. Coded in the same manner as `hurt_pax`

vol This field must be an integer. If it is positive, it is CTX's best estimate of the amount spilled in this event in liters. This is all oil released into the environment regardless of whether or not is recovered, spilled on land, or burned, or captured in the sunken hull. However, oil that is contained onboard ship (say on deck) or oil that is contained within a terminal in save-alls and the like is not counted.

A -1 means we know there was a spill but we have no estimate of volume. A -2 means we suspect there was a spill but we are not sure. If the number is -4, it means the spillage from this ship was counted

in the spillage on the first ship. Sometimes we only know the total spillage in a casualty. This code handles this case.

Spillage should be assigned to the event when the spill occurred. This is usually a hull damage event. ***This does NOT imply any causality.*** Causality is determined by the `cause` flag(s). Sometimes we don't know when the spill occurred. Don't worry about it. Just make your best guess.

Even a single ship casualty may have multiple spills. The Sea Empress had at least four separate spills. The fact that spill volume is a property of an <event> also allows us to disinguish between cargo and bunker spills. A tanker might have a cargo spill as a result of being holed in a collision and then a bunker spill when she sank. But there may be only one spill `vol` per <event>.

mat Material spilled in this event's spill. The CTX data base covers only liquid petroleum, ethanol, LPG and LNG spills. Currently, other vegetable oils, non-petroleum chemicals, are not included. Material spilled is not necessarily the cargo. See `cgo` in Section 2.5.

	?
A	alcohol/MTBE
B	Ships own bunkers
C	Crude Oil
D	Distillate
F	Heavy fuel oil as cargo
G	Gasoline/naptha
K	Carbon Black
L	Lube oil
T	Tar/asphalt
l	Liquified Natural Gas
p	Liquified Petroleum Gas

If both cargo and bunkers were lost, as is often the case when a tanker founders, either use the predominate code; or split the volume into two events. It is illegal to have more than one `vol` field in the same event.

ref Free form text identifying the source of the information.

note This field is free form text describing the event. Do NOT repeat the event code definition or *label* in this description. Assume applications displaying this field will also display the event code label. The `note`

field should describe only the unique features of this specific event. This field is not required, but highly recommended. Many applications will display the event sequence when a user selects a particular casualty. Typically, the application will display the event code label and this `note` resulting in a concise, informative summary of the casualty if and only if the event `note` is well-coded. See Figure 2.2 above. The event `note` field should be as long as required, but no longer. Applications are expected to be able to handle multi-line strings in this field.

wind_bf Wind speed coded in the same manner as `weather`.

need to change name of weather

wind_dir True wind direction (from). Can use *nnn* with leading zeros or compass points up to three characters, e.g. `SSE`. Degrees true or compass points but only down to three letters, eg WSW.

vis Visibility coded in the same manner as `vis` in the casualty common block.

curr_spd Current speed in knots. If you only know that there was little or no current, you may code `N` for nil.

curr_dir Current direction (to). Degrees true (padded) or compass points but only down to three letters, eg WSW.

wave_ht Wave height in meters, peak to trough. This should be some sort of significant wave height, i.e swell. Usually we are only interested in waves long enough to generate sigificant ship motion. If you only know that there was little or no waves, you may code `N` for nil.

still called swell, must change

wave_dir Wave direction (to). Degrees true (padded) or compass points but only down to three letters, eg WSW.

sw_temp Sea water temperature in Celsius. Important if brittle fracture suspected.

air_temp Ambient air temperature in Celsius. Important if brittle fracture suspected.

spd Ship speed in knots at the time of this event. If this is speed over the ground, append `G`. If this is speed through the water, append `W`. If don't know and are unwilling to guess, append a `?`. Use `A` if anchored, `M` if moored. Can use `D` if drifting. You cannot enter both speed over

ground and speed though water. Use `curr_spd` and `cur_dir` fields if you have both.

squat Squat in meters. Should be maximum increased in draft due to vessel motion.

course Ship heading at the time of this event. Degrees true (padded) or compass points but only down to three letters, eg WSW. Normally used for heading at impact for collisions and grounding.

nuc `Y` implies Not Under Control at time of event due to loss of power or steering. Usually used in collisions or groundings. `N` means the ship had power and steering when she went aground/collided. Use `Y` if loss of power/steering causal, even if the ship had restored power/steering prior to the actual impact. This field has nothing to do with fault. The ship need not be at fault if `nuc` is `N` (eg charted depth incorrect). The ship could be at fault if `nuc` is `Y`. See BRAER for an example. Use `B` is the ship was purposely beached.

dop Maximum transverse depth of penetration in meters. Your can prefix a plus sign to indicate "at least". If hull is indented but not holed, use `I`. If you don't know the depth of penetration, but you are sure it is less than 2 meters, use `S`. If you don't know the depth of penetration, but you are sure it is more than 3 meters, use `L`. If you have more details on the damage, use a hull damage event, Section 2.4.7, but fill this field out anyway. Some queries depend on it.

hop Highest point of penetration in meters above ship baseline due to this event. Coded like `dop`. Generally used in groundings.

lop Maximum length of damage longitudinally in meters due to this event. -1 means none. Use a hull damage event to describe the damage in more detail. See Section 2.4.7.

2.4.4 Causal Factors

An <event> element may have any number of *causal factor* fields. A causal factor attempts to describe the circumstances surrounding an event at a finer level of detail than can be done at the event level. Recording "Structural failure" immediately raises a number of questions: most importantly, why? Was it corrosion or fatigue or brittle fracture? Causal factors tend to be a condition or attitude rather than a specific event.

Causal factors should be assigned to the event to which they are most closely related. Here is how the first two causal events in the Exxon Valdez casualty are coded:

```
<event ec="GY" sid="1" date="19890323" tod="2353" Cause="N" Sure="5"
       Drugs="M"
       note="VLCC with 20 man crew, leaving Valdez after loading.
             Master leaves nav to tired mate to work around ice"/>
<event ec="NA" sid="1" date="19800324" tod="0001" Cause="P" Sure="5"
       Tired="Y"       Crew_small="Y"  GPS="P"    Owner_care="Y"
       note="Mate starts stbd turn too late"/>
```

Figure 2.3: Exxon Valdez Causal Factors

The causal factors fields in this snippet are `Drugs`, `Tired`, `Crew_small`, `Owner_care`. They are defined below, but you have already guessed what they mean.

Causal factors, like the `Cause` and `Sure` fields, almost always require judgement. To indicate this, all causal factor fields start out with a capital letter. Processing applications should treat/display them differently from the purely factual fields, perhaps by using a contrasting color or font. Of course, in many cases, the evidence is such that no reasonable person would argue with the judgement. But it is also true that often this is not the case.

The level of confidence is indicated by the field value. Except where explicitly indicated below, all causal factor fields are coded as follows:

y	this factor almost certainly an important factor
p	this factor likely an important factor
m	strongly suspect this factor an important factor
n	this factor not an important factor

If this field is missing or blank, applications must assume that this factor was not an important factor, even though we may simply have no information on this factor. `N` can be used to explicitly indicate that we have

information that this factor was not an important factor. Generally, the `N` code is only used in situations where this might be surprising.[8]

This data structure accomplished two main goals:

1. Not only may a casualty have multiple causal events, but each event may have multiple causal factors. Many structural failures involve a combination of corrosion and fatigue. Multiple causal factors are a commonplace in casualties. With this structure, this can be recorded. The coder is not forced to single out a particular causal factor, and identify it as "the" causal factor.
2. The quality of evidence and hence the degree of confidence in each causal factor is accessible to the user. The sad fact is that in most casualties our data is incomplete. This structure allows the user to see the level of uncertainty, and react accordingly.[9]

Normally causal factors must be read negatively, eg `Inerting="Y"` means we are essentially certain that *lack* of inerting was an important causal factor. However, sometimes a causal factor can have a positive influence, such as when we have evidence that proper inerting prevented a casualty from being worse than it was. You can record this with the following codes.

3	This factor almost certainly had an important positive effect
2	This factor likely had an important positive effect
1	Strongly suspect this factor had an important positive effect

Positive effect codes are allowed for only some of the causal factors as detailed below.

The causal factor fields are:

Bank This flag indicates whether bank suction is an important factor in a collision or allision.

Blackbox A fairly recent development is the dependence on software for controlling critical systems which the crew cannot only not fix, but the owner and other third parties cannot even scrutinize. In CTX experience, often the vendors themselves do not understand their own software. If opaque, incorrect software is implicated in this casualty, record this with a `Blackbox` field.

[8] Coding a causal factor field with a blank as opposed to not coding it at all is used to indicate to other coders that, while we don't have enough information to say this field was strongly suspected to be a factor, the coder thinks it might have been; and we should look for information to confirm or deny this possibility.

[9] The `Sure` field relates to the overall assignment of the <event> as a cause, not to any individual causal factor field.

Brittle Use this causal factor to indicate that brittle material failure was implicated in the casualty. Usually applied to structural problems, but could also apply to machinery failures.

Cgo_leak Indicates whether cargo leaking . into a space where it should not be is an important factor in this casualty, usually a fire. Coded as follows.

too much overlap with sbt

Y	Cargo leaking almost certainly an important factor
y	Cargo leak into SBT almost certainly an important factor
P	Cargo leaking likely an important factor
p	Cargo leak into SBT likely an important factor
M	Cargo leaking possibly an important factor
m	Cargo leak into SBT possibly an important factor

Use the SBT varient if the space being leaked into is a segregated ballast tank, double bottom, or the like.

Cgo_survey Sometimes the fact that a tanker was undergoing a cargo survey is implicated, usually because the ullage/dipping points were opened and/or the inerting compromised.

Charts Indicates whether needed charts were lacking or not updated correctly. Coded as follows.

Y	Charts not updated almost certainly an important factor
y	Missing charts almost certainly an important factor
P	Charts not updated likely an important factor
p	Missing charts likely an important factor

Class_care If Class has failed to ensure that the owner maintains his ship in a robust condition, record it in this field.[10] Notice some causal factor fields must be interpreted in a negative sense. `Class_care` here really means inadequate Class care.

Crazy See `Spite`. Sometimes crew behavior goes far beyond bad feelings to something approaching insanity. The `Crazy` attribute tries to capture

[10] A reasonable question is: why do we not have a `Flag_care` field? The answer is that the Flag state almost never does anything to ensure the design or condition of the ship. Nobody expects it to do any thing. Lack of flag state care is almost automatic. In the CTX database it is treated as a sadly uninteresting given.

such inexplicable actions. An example would be the Mate of the give-way Yaw Malaya not only plowing into the side of the New World, but responding to the VHF requests of the stand-on New World with racial curses.

Crew_small Indicates whether crew size was an important factor in the casualty. This field has a lot of overlap with `Tired` but it is quite possible to have a large crew and a badly fatigued captain or chief mate. And one could have fresh crew, but one too small to reliably handle STS mooring, maintain a proper lookout, or the like.

Design Indicates whether incorrect or inadequate design was a factor.

Drugs Indicates whether drugs including alcohol was an important factor in this casualty.

Ergon Indicates whether bad ergonomics was a factor. Use this for confusing interfaces, badly placed switches, etc.

Fatigue Use this causal factor to indicate that structural fatigue was implicated in the casualty. Usually applied to hull structural problems, but could also apply to machinery failures.

Fp_aft Use this field to indicate that the fact that the emergency fire pump was installed aft next to the engine room was a factor in the casualty. Positive codes allowed.

Gen_size Many ships are built with under-sized generators. In theory, ship are designed with considerable generator redundancy. But Class rules allow the yards to use highly unrealistic assumptions in estimating both the electrical loads and the generator's actual in service capacity. As a result, ships need two or three generators on-line when they should only need one or two, and the loss of one generator blacks out a ship. If under-sized generators was an important factor in this casualty, record this with a `Gen_size` field

Hotwork This field indicates whether unsafe hotwork an important causal factor in a fire/explosion or personnel injury.

Inerting Indicates whether poor or non-existent inerting is an important factor in this casualty. Coded as follow:

Y	No IGS almost certainly an important factor
y	Poor inerting almost certainly an important factor
P	No IGS likely an important factor
p	Poor inerting likely an important factor
M	No IGS possibly an important factor
m	Poor inerting SBT possibly an important factor
3	Proper inerting almost certainly had an important positive effect
2	Proper inerting likely had an important positive effect
1	Strongly suspect proper inerting had an important positive effect

Use the upper case varient if the ship was not fitted with an inert gas system. Use the lower case varient if the ship was fitted with an inert gas system, but it was ineffective; or, if all you know is poor or no inerting was a factor.

If proper inerting would have prevented the original ignition, put the `Inerting` field in the ignition event. If proper inerting would only have reduced the damage associated with a fire or explosion, put the `inerting` field in the fire/explosion event.

Language Indicates whether lack of a common language was an important factor in this casualty.

License Indicates whether illegal, forged, missing license was an important factor in this casualty. Use also if crew man was doing a job for which he was not licensed.

Lightning Indicates whether lightning is an important factor in this fire.

Loading Indicates whether poor or illegal cargo loading pattern is an important factor in this causualty.

Machinery Indicates whether machinery failure is an important factor in a fire/explosion. This could be anything from over-heated bearings to a crankcase explosion. In this case the casualty should have a machinery event describing the details of the failure. See Section 2.4.8. It is possible to have a machinery failure such as a crank case explosion without having a fire event provided that the fire is immediately contained. bad name

Maint Indicates whether bad maintenance was a factor. If owner was aware or fostered this condition, code both `Maint` and `Owner_care`.

Maneuv Indicates whether poor or limited vessel maneuverabilty was a factor.

Nav_gear Indicates whether navigational equipment was lacking or not operating correctly.

Do you want to identify specific equipment?

- Y Improperly operating nav gear almost certainly an important factor
- y Uninstalled nav gear almost certainly an important factor
- P Improperly operating nav gear likely an important factor
- p Uninstalled nav gear likely an important factor

Owner_care If an owner has knowingly allowed the ship to deteriorate or failed to respond to reasonable requests from the crew or others for repairs, spares, etc, record it in this field. Use `M` if the ship has an owner with a bad record and the ship is clearly in bad condition.

Paperwork Indicates whether unnecessary or unreasonable paperwork was an important factor in this casualty. Do NOT use this for failure to maintain paperwork. The CDB has no field to record this common bureaucratic scapegoat.

Pilot Indicates whether the process of picking up or dropping off the pilot was a factor. Coded as follows.

- Y (Dis)embarking pilot almost certainly an important factor
- P (Dis)embarking pilot likely an important factor
- M (Dis)embarking pilot possibly an important factor

Piping Indicates whether fuel oil or lube oil piping failure is an important factor in this casualty. Usually applied to fires.

Pitting Indicates whether pitting failure implicated in the casualty.

Purging Indicates whether bad purging or tank cleaning is an important factor in this casualty.

Rust Indicates whether corrosion implicated in the casualty.

Safety Use this for clear violations of reasonable safety regulations.

Sbt Cargo leaking into a segregated ballast tank is often a very important factor in tanker and combo casualties. For the purpose of this fields, a permanent ballast tank on a pre-Marpol tanker is a segregated ballast tank, as is the forepeak tank.

Sbt_rust Corrosion is segregated ballast tanks is often a very important factor in tanker and bulk carrier casualties. There is a big overlap between this field and `Sbt`. But it is possible to have cargo leaking into a ballast tank with little or no corrosion (eg a fatigue crack), and it is possible that ballast tank corrosion could be a major factor with a cargo leak. If this field is coded, it is not necessary to code `Rust` unless non-ballast tank wastage was also important.

Spite `Spite` is short-hand for bad relationships among the crew affecting decisionmaking. Cahill points out that animosity between crew members can be a surprisingly important factors in ship casualties. Bad blood between the Captain and Chief Officer of the Torrey Canyon probably influenced the Master's decision to countermand the Mate's intelligent course change, and then stick with a reckless and unnecessary alternative. Conversely, the Fina Novege's Mate's antipathy towards his captain probably influenced him to hold a course that he knew was wrong until the Captain came to the bridge and changed it himself. The `Spite` attributes tries to capture such behavior.

Stability Indicates whether loss of stability was an important factor in this casualty.

Static Indicates whether static electricity from tank cleaning an important causal factor in this fire.

Stern_swing Indicates whether stern swing was an important factor in this casualty.

Switch Indicates whether fuel switching was an important factor in this casualty. Currently, fuel switching refers to the switching imposed by requiring very low sulfur fuel in certain areas.

Tired Indicates whether crew fatigue was an important factor in the casualty. See `Crew_small`.

Training Indicates whether lack of training an important factor in this casualty.

Wiring Indicates whether electric arc-ing due to bad wiring or a failure in the electic distribution system an important factor in this casualty, usually a fire.

Do not be afraid to have multiple causal factor attributes. Often bad ownership and a negligent Classification Society go together. It is perfectly legal to code `Owner_care="Y" Class_care="Y"`.

2.4.5 The RCO Fields

Each <event> element may contain zero or more Risk Control Option (RCO) fields. Each RCO field attempts to answer the question, would this particular RCO have helped avoid or mitigate this casualty. In making this judgement, assume that the RCO option is properly used. However, if the operation or management of the ship is clearly abyssmal, you may downgrade the RCO a notch. If the ship was fitted with or employed this Risk Control Option, the question becomes: did this particular RCO help in this casualty?

Pair the risk control option with the events they could have affected/did affect. In the Sea Star/Horta Barbossa collision, Figure 2.1, proper use of VHF would almost certainly have prevented the collision. So it is placed in the primary cause event, the Dance of Death maneuver. Proper inerting would have done nothing to prevent the collision, but it might have mitigated the effects. So it is placed in the collision event.

The RCO fields are not facts, They involve even more judgement than the causal factor fields. Therefore, the names of RCO fields, and only RCO field names are completely capitalized. Processing applications should treat/display them differently from the factual fields. Applications which wish to deal only in facts can choose to ignore the RCO fields. Except as explicitly noted below, all the RCO fields are coded as follows.

Y	Very likely to certain, this RCO would have helped
y	Very likely to certain, this RCO did help
P	This RCO would probably have helped.
p	This RCO probably did help.
M	This RCO might have helped
m	This RCO may have helped.
N	This RCO would probably have made no difference
n	This RCO probably made no difference
W	This RCO would probably have made things worse
w	This RCO probably did make things worse
	Cant say

Use the lower case values for ships for which the RCO (eg IGS) had been implemented. If you don't know whether the RCO was implemented, use the upper case codes.

If a particular RCO field is not present, then processing software will assume this RCO would have made (did make) no difference. However, you can use `N` or `n` to explicitly indicate that there is excellent reason to believe

that a particular RCO would not helped or did not help. This is often used with `db` and `ds` in conjunction with groundings and collisions.

The possible RCO fields are:

TS This field attempts to answer the question: would twin screw have helped (or did help) in this casualty? Twin screw should be defined in the strict sense: full redundancy, engine rooms separated by an A60 bulkhead, equivalent to ABS R2-S+ notation. Applies mainly to total loss of power/steerage events, but don't forget to credit twin screw for the improved low speed manueverability, and lack of stern swing if appropriate.

VHF This field attempts to answer the question: would/did proper communications with the other ship have helped/help in this casualty. Applies mainly to collisions.

VTC This field attempts to answer the question: would/did mandatory vessel traffic control have helped/help in this casualty. Applies mainly to collisions.

GPS This field attempts to answer the question: would a modern GPS based ECDIS system have helped in this casualty? Use `Y` for navigation error casualties where we know plotting errors. or delays were instrumental. Use `P` for navigation casualties where there was apparently little effort on the part of the crew to know where they were. In other words, the crew was so bad they could easily have screwed up with GPS/ECDIS. You can also use `P`, for cnning casualties where GPS/ECDIS would have alerted master/pilot to the problem earlier.

DS In considering hull type, it is extremely important to distinguish between double sides and double bottoms. They are two entirely different animals with respect to their impact on spillage and safety. See Tankship Tromedy, Appendix C.

The `DS` field attempts to answer the question: would double sides have helped or did help in this casualty? This RCO apples only to casualties in which the hull shell was penetrated, so it should only show up in a hull damage event. If damaged ship is not double sided, assume an IMO double side width in coding this field. Use `Y` for casualties for which location of damage is known and for which we have supporting calculations. Remember to give double sides credit for oil capture in the top of wings when appropriate. See Tromedy Section C.8. Use

`P` for casualties for which we don't have calculations but location, penetration favorable for double side. Do not use `M` for collisions unless we have good reason to believe that there was penetration and it was shallow. This field should not show up or be `N` for most collisions, high impact groundings, complete losses, or we know the double side was/would be penetrated.

This RCO can also have a value of `C`. Use `C` for internal cargo tank cracks which in single sided hull would have resulted (or did result) in spill but in double sided hull would have leaked/did leak into ballast tanks. If single hull leak is probably due to external corrosion, use M or P. The reasoning behind the special treatment of cracks is that, while a double side would/did prevent a spill associated with a shell crack, a crack which allowed a cargo leak into double hull ballast space is far more dangerous. If a ship is double sided and has a low impact damage to a single skin BFO tank, use `M` or `P`. That is, define double sides to mean BFO tanks also double sided.

DB This field attempts to answer the question: would double bottom have helped or did help in this casualty? This RCO apples only to casualties in which the hull shell was penetrated, so it should only show up in a hull damage event. If ship is not double bottom, assume an IMO double bottom depth. Use `Y` for casualties for which location of damage in known. and for which we have supporting calculations. Use `P` for casualties for which we have no calculations but location, penetration favorable for double bottom. Use `M` for medium or low impact groundings with spillage, for which we have no penetration, location data. This field should be missing or `N` for high impact groundings, complete losses, or we know the double bottom was/would be penetrated. Give little weight to double bottom oil capture. See Tromedy, Section C.8.

This RCO field can also have a value of `C`. Use `C` for internal cargo tank pits or cracks which in single bottomm hull would have resulted/did result in spill but in double bottom hull would presumably have leaked into ballast tanks. If single hull leak is probably due to external corrosion, use `M` or `P`. The reasoning behind the treatment of pits is that while a double bottom would/did prevent a spill associated with a cargo tank pit, a pit which allowed a cargo leak into double bottom ballast space is far more dangerous.

AIS This field attempts to answer the question: would AIS (Automatic

Identification System) have helped in this casualty? Applies mainly to collision events. Assume the AIS is properly used and, if the other ship is over 5,000 GRT, it has it. Use `P` for collisions for which we know the ships were aware of each other and collided anyway. Use `M` for collisions for which we have no cause info.

IGS This field attempts to answer the question: would proper (CTX) inerting have helped in this casualty? Applies mainly to fire/explosion events. Define proper inerting to mean ballast tanks as well as cargo tanks combined with a proper tank purging system. Use `Y` for casualties in which we know inerting/purging was non-existent or bad and a tank fire/explosion resulted. Use `P` for casualties where we can infer inerting/purging was bad and a tank fire/explosion resulted. Use `M` for `XT` casualties with no clear cause. Do not use if structural failure probably occurred first.

BTM This field attempts to answer the question: would proper protection of ballast tank steel have helped in this casualty? Define `BTM` to mean prevention of essentially all ballast tank wastage by a proper combination of coating, anodes, double scrubbed inerting, inspection and upkeep. That is, ballast tank maintenance to CTX standards. Use `Y` for structural casualties in which we know ballast tank wastage was a major factor. Use `P` for structural casualties in which we have evidence that tank wastage was a factor. Such evidence may include uncoated ballast tanks. Use `M` for structural casualties which are not clearly fatigue or massive over-stressing but we have no direct evidence of ballast tank corrosion.

BC This field attempts to answer the question: would a bigger crew have helped in this casualty? If crew fatigue other than Master or Chief Engineer is implicated, then you should should give an affirmative answer.

Analysts need to keep in mind a fundamental problem with these RCO fields. If an RCO has prevented a casualty, this success will never be recognized in the database. Use of VHF is an example. It is a good bet that communication between ships has prevented a sizable number of collisions; but the CDB has no way of recording this. There have been casualties in which the ships communicated and then collided anyway. In a few cases, the ships might possibly have been better off not to communicate. Any analysis of the VHF RCO field will pick up the failures but miss most of the successes.

2.4.6 Event Code Categories

The event codes fall into the following categories,

Structural Failures/Damage
Machinery Failures/Damage
Bridge Events
Navigation Errors
Guidance, Seamanship Errors
Inerting/Hotwork Screw Ups
Cargo Handling/Transfer Errors
Lifeboat/liferaft Events
Intentional Spills/Sinking
War/piracy attacks
Other External (non-Ship) causes

Fire/Explosion
Collision/Allision
Grounding/Stranding
Rescue
Disable/Response
Fate

The first eleven categories may be causal. The remaining categories may not. Each of these event code categories is described in the following sub-sections.

We also describe the "category specific" fields which can be included in each event. However, any field or sub-element may be included in any <event> element, where it is needed. If tidal information was important in a collision, by all means include it.

2.4.7 Structural Damage

Introduction

The hull failure/damage event codes are:

Structural Failure Codes

CODE	LABEL	CAUSAL?
HC	Hull crack, minor hull failure	Y
HD	Deck Vents/Equipment failure	Y
HF	Major Structural failure	Y
HL	Holed	N
HP	Cargo Pipe failure/leak	Y

Structural damage can be either a cause or a consequence.

Here is an example of a causal hull failure event from the CASTOR causalty. In such cases, the primary job of the event element is to record why the structure failed. Note the use of causal factors.

```
<event ec="HF" sid="1" Causal="P" Sure="5" deck="Y"
       Design="N" Rust="Y" Fatigue="N" Brittle="N" Pitting="N"
       Owner_care="Y" Class_care="Y"
       ref="abs20011017"
       note="26 m transverse crack on deck just aft of fr 72"
    <tank code="4P"
          note="deck plate and longys wasted in excess of 65 pct iwo fr 72 and 73.
                some longys detached from deck">
    </tank>
    <tank code="4C"
          note="deck plate and longys wasted in excess of 65 pct iwo fr 72 and 73
                apparently longys not detached">
    </tank>
    <tank code="4S"
          note="wastage not as bad as 4P and 4C
                but in some areas 16 mm deck down to 5 mm">
    </tank>
</event>
```

See also the <condition> sub-element of the <ship> element, Section 2.5.4 which allows us to record the structural condition of each compartment.

Whether or not the structural damage is a cause or consequence, in order to do any sort of analysis of the flooding, spillage and/or the ship's residual strength we need a much better description of the location and extent of the damage than simple depth or height of penetration. The EXXON VALDEZ is a casualty in which the structural damage is a consequence not a cause. Here is her hull damage event.

```
<event ec="HL" sid="1" ref="ntsb91" steel_wt="3500"
       note="ntsb height numbers look way optimistic.">
  <tank code="1C      " size="" perim=""
        note="ntsb way low in this tank, damage actually extended past cl">
        <hi xs="236.1" ys="-12.1" zs=" 1.0"/>
        <lo xs="274.5" ys="  0.0" zs=" 0.0"/>
  </tank>
  <tank code="2C      " size="" perim="">
        <hi xs="186.5" ys="-12.1" zs=" 1.0"/>
        <lo xs="236.1" ys="  0.0" zs=" 0.0"/>
  </tank>
  <tank code="3C      " size="" perim="">
        <hi xs="132.4" ys="-12.1" zs=" 3.0"/>
        <lo xs="186.5" ys="  0.0" zs=" 0.0"/>
  </tank>
  <tank code="1S      " size="" perim="">
        <hi xs="236.1" ys="-24.5" zs=" 3.3"/>
        <lo xs="274.5" ys="-12.1" zs=" 0.0"/>
  </tank>

 .... about ten more compartments ......

  <tank code="FP      " size="" perim=""
        note="jack guess, ntsb said nothing abt FP, but it was a mess">
        <hi xs="274.6" ys="-15.0" zs=" 3.0"/>
        <lo xs="292.0" ys="  0.0" zs=" 0.0"/>
  </tank>
</event>
```

The attributes specific to hull damage events are:

steel_wt Amount of steel replaced in metric tons.

hbl Certain casualties are interesting examples of hydrostatic balance. If this is the case, indicate it by including `hbl="Y"` in the <damage> element.

structural part You may indicate the function of the structure that initially failed by one or more of the following fields.

deck	Main Deck (not hatch cover)
hcvr	Hatch Cover
side	Side shell
bott	Bottom shell
inbt	Inner bottom
insi	Inner Sides
tbhd	Transverse Bulkhead
lbhd	Longitudinal bulkhead
tweb	Transverse web
hstr	Horizontal stringer
duct	Duct keel

For all the structural part fields, the coding is the same

Y	This structural part definitely part of the initial failure
P	This structural part probably part of the initial failure
M	We have information this part may have been part of the initial failure

LNG specific parts: inner/outer barrier, etc

Don't be afraid to have multiple structural part fields. If it is a 50/50 chance that the initial failure was a hatch cover and 50/50 chance it was the deck itself, you might use code `deck="M" hcvr="M"`.

If a structural part field is missing, it means only we don't have specific information that this portion of the structure was part of the the initial failure. This could be because we know the part was not involved or we have no information that the part was involved. We accept this ambiguity to avoid having to enter reams of negative fields.

ice This field indicates if ice was involved in the damage. Coding is similar to a causal factor, although the CDB regards ice more as an ambient condition than a causal factor.

Y	Ice almost certainly involved
P	Ice likely involved
M	Strongly suspect ice involved
N	Ice not involved

Damage Location Fields

The CTX CDB breaks damage down by compartment. All compartments which are known to be damaged should be represented by a <tank> sub-element. For the purposes of damage, any compartment (eg cargo hold,

engine room) is considered to be a <tank>. If a compartment does not show up in the list of <tank> elements, code using the CTX database will assume that that compartment was not damaged.[11] If all you know is that a compartment was damaged, and nothing more, the <tank> element should contain only the `code` attribute, that is the name by which that compartment is known in the ship's capacity plan. However, CTX follows the convention that the code for all cargo holds starts with an `H` and the holds are numbered from bow (1) to stern. Use ER for engine room, PR for pump room.[12] For example

```
<tank code="ER      ">
</tank>
```

The CTX model of damage within an individual compartment is a box. The damage box is represented by any two opposite corners. One of these corners <hi> should be at the level of the highest point of damage in the compartment. The other corner <lo> should be at the lowest level of damage in the compartment. (The vertical extent of damage is critical to spillage calculations.) The idea is that programs assessing residual strength will eliminate all structure in the compartment that is within the box. Programs assessing spillage will use the intersection of the damage box and the compartment's exterior in calculating outflows and inflows. The <hi> and <lo> points of the box are given in Joint Tanker Project (JTP) coordinates.

xs Distance forward of the Aft Perpendicular in meters.

ys Distance port of the ship's centerline in meters. Starboard is negative.

zs Distance above the ship's baseline in meters

Of course, real damage will almost never take the shape of a box. But normally you won't go far wrong in either structural of spillage calculations by fitting a box around the damage in each compartment as tightly as possible.

The damage box is not required, but without it little flooding or strength analysis of the casualty will be possible. Work very hard to make a guess at the box, even if you know it is not accurate. If you know the height of damage in a tank but not its longitudinal position, it is perfectly OK to center the damage longitudinally in the tank. You may record any assumptions you have made or any reservations you have about the data in the `note` attribute.

The other <tank> damage location attributes are:

need to implement tank type down the line

[11] The exception is, if there are no <tank> elements at all, then processing software should assume, we do not know which compartments were damaged.

[12] ER_P and ER_S for port/starboard engine rooms.

flood If all you know about a compartment is that it was flooded, use `flood="Y"` to record this.

size The total area of the external hole(s) in the compartment in square meters. Spillage programs may use this number to attempt to estimate spill rate. ***This should be used in conjunction with, and not as a replacement for the damage box.*** The location of the hole is much more important than the size. If the vertical extent of the hole is known, spillage programs can calculate how equilibrium spillage and flooding, without knowing the size of the hole. If the vertical location of the hole is not known, spillage programs can do nothing with the area.

perimiter The perimiter of the external hole in the compartment in meters. This should be used in conjunction with, and not as a replacement for the damage box. Spillage programs may use this number in conjunction with area in attempting to estimate spill rate. For cracks, this fields should be twice the crack length.

2.4.8 Machinery Failure

A casualty may involve one or more machinery failures. Machinery in the CTX vernacular includes deck machinery such as anchor windlessess.

The machinery failure event codes are.

Machinery Failure Codes

CODE	LABEL	CAUSAL?
MA	Unable to slow down	Y
MB	Blackout/electrical problem	Y
MD	Deck Machinery Failure	Y
MF	Engine Room Flooded	N
ME	Main Engine Failure/Problem	Y
Ml	Lifeboat failure/problem	Y
MO	Fuel/lube Pipe leak	Y
MP	Propeller failure/damage	Y
MR	Steering gear/rudder failure	Y
MS	Shaft/Sterntube Failure	Y
MT	Sterntube Leak	Y
MW	Sea Water/Condenser Line leak	Y
MX	Boiler failure/fire	Y
MY	Other or Unknown Machinery failure	Y

Figure 2.4 is an example of a machinery failure event from the Bright Field casualty. Notice the distinction between the machinery failure (main engine low lube oil pressure, the `ME` event) and the resulting loss of all propulsion power, the `DP` event, and the surprising loss of steerage, `DS` event, despite the fact that the ship's steering system was still functioning.[13] See Section 2.4.22. Notice also that the twin screw RCO field, `TS`, is applied to the total loss of power/steerage events, not to the machinery failure.

The purpose of the `MA` code is to record situations in which the ship was not able to slow down further without losing steerage, or did not turn when it could reasonably be expected to. The causal factor `Maneuv` will almost always be part of an `MA` event; but it is more general allowing you to indicate that poor low speed maneuverability was a factor in a wide range of situations.

[13] The Bright Field, like many bulk carriers, had such poor maneuvering characteristics that she was uncontrollable without propulsion power, even at close to 10 kts through the water.

```
<event ec="ME" sid="1" tod="1410" cause="P" sure="5"
       Maint="Y" Owner=care="Y"
       note="Main engine trips on low LO pressure">
    <component er="M"  sfi="713004" name="No 1 lube oil pump"
          maker=""     model=""
          power="   "  hours="    " ttr="  " survey="       "
          failure="low pressure tripped mn engine"
          note="lo pump in bad condition">
    </component>
    <component er="M"  sfi="713006" name="No 1 LO pump filter"
          maker=""     model=""
          power="   "  hours="    " ttr="  " survey="       "
          failure="filter partially clogged"
          note="lube oil contaminated with fuel">
    </component>
    <component er="M"  sfi="797010" name="LO pump automation"
          maker=""     model=""
          power="   "  hours="    " ttr="  " survey="       "
          failure="standby pump auto-start failed"
          note="relay had high resistance">
    </component>
</event>
<event ec="DP" sid="1" tod="1410" Cause="N" Sure="5"
       lop_hrs="0.03" TS="Y"
       note="Single screw, no redundancy"/>
<event ec="DS" sid="1" tod="1410" Cause="N" Sure="5"
       lop_hrs="0.03" Maneuv="Y" TS="Y"
       note="at about 10 kts, even tho steering operational"/>
```

Figure 2.4: Sample machinery failure event

CTX uses a broad definition of casualty. A discovery of a serious problem which requires diverting or delaying the ship is a casualty, even if it does not result in death, damage or a spill. For example, Stena's taking the Stena Vision out of service for several months in May of 2006 to repair major engine problems discovered during her first Special Survey is included in the database.

Similarly, you should use a broad definition of machinery. An anchor cable parting is a perfectly bad machinery failure.

The attributes specific to machinery failure events are:

main engine sub-system For `ME` events, you may indicate the sub-system(s) of the main engine involved by one or more of the following fields.

Probably will go away in favor of component

air_sys	Control air problem
bedplate	Bedplate/bearing girder
crankcase	Crankcase Explosion
crankshaft	Crankshaft/lower end
cyl_head	Cylinder head/exhaust valve
liner	Cylinder Liner
piston	Piston/cross-head
turbocharger	Turbocharger

For all the engine sub-system fields, the coding is the same

- Y This sub-system definitely part of the initial failure
- P This sub-system probably part of the initial failure
- M We have information this sub-system may have been part of the initial failure

deck machinery sub-system For `MD` events, you may indicate the deck sub-system(s) involved by one or more of the following fields.

Probably will go away in favor of component

windless	Anchor Windless
winch	Mooring winch

Coded in the same manner as main engine sub-systems.

Machinery failures often involve a sequence of *component* problems. A machinery failure event should contain a <component> sub-element for each component involved in this event. The component attributes are:

er Engine room flag. This field is required if and only if the ship is twin screw and it is an engine room casualty. For the purposes of Machinery Failure, the steering gear flat is considered to be part of the engine room. Use `M` for the engine room of a single engine room ship, `P` for port, `S` for starboard for twin screw ships.

name Free form description of component involved. This should be as specific as possible, but put what you know. If all you know is the casualty involved the main engine, put "main engine". Every <component> element must have a non-blank `name`.

sfi The SFI Group code as described in SFI Group System, The Ship Research Institute of Norway. This is a six digit code, with the first digit

SFI is proprietary. Need an alternative. Maybe Martingale sys/subsys/label.

specifying the main ship system, (eg, 3 for cargo system), the second, the group (eg, 37 for tank inerting/purging) the third, the sub-group (eg 375 for pressure release systems), and the final three digits (the "detail" code) the specific component (eg 375014 Flame arrestor). If only the main system is known, then code just that single digit. If group is known, but not sub-group, code the two digit group code. If sub-group is known but not the component, this field should be three digits. Otherwise, use the entire six-digit code.

maker Maker. Currently this is free form, but try to be consistent. Sulzer in all its various name changes is Sulzer. MAN/B&W, etc is MAN.

model The maker's model number.

power The output power of the machine at the time of failure as a fraction of MCR.

hours Running hours on the component at time of failure.

since overhaul, reconditioning?

survey_date Last survey date.

last overhaul date?

ttr Time to repair this component in hours. This need not be the time to restore power.

failure Free form description of problem.

note Freeform description of any other info on component failure.

Only if you have no idea which ship system was involved, should there be no <component> sub-elements. <component> elements are not limited to machinery failure events. For example, they can be used in lifeboat hook failures to document the particulars of the hook that failed.

A casualty may have as many machinery failure events as necessary to fully describe the casualty. Sometimes a component is thought to be repaired, and then fails again a short time later. This should be represented by multiple events rather than multiple <component> sub-elements in one event.

A properly responding trip is not a causal event. If something fails, which causes the generator(s) to trip, resulting in a black out. The generators did not fail, even if the time to restore power is longer than the time to repair the failed component.

2.4.9 Bridge Events

The Bridge events attempt to record what happened on each bridge prior to and during a casualty, usually (but not always) a collision. Here are two sample Bridge events.

```
<event ec="VD" sid="1" Cause="P" Sure="5" detect_mi="6" rule="H"
       note="Patchy fog off Ushant. Both detect. No talk.
             Northbound Arietta Livanos, did nothing, at 1.5 miles went hard port">
    <alter tod="1536" course="050" spd="15.5W" helm=" " engine="FA"/>
    <alter tod="1541" course="050" spd="15.5W" helm="p" engine="FA"/>
</event>
<event ec="VD" sid="2" Cause="N" Sure="5" detect_mi="8" rule="H"
       note="Southbound Anneliese altered slightly to stbd, late went stbd">
    <alter tod="1531" course="237" spd="15W" helm=" " engine="FA"/>
    <alter tod="1536" course="235" spd="15W" helm="S" engine="FA"/>
    <alter tod="1539" course="224" spd="15W" helm="S" engine="FA"/>
    <alter tod="1542" course="207" spd="15W" helm="S" engine="ST"/>
</event>
```

Figure 2.5: Sample Bridge events

The Bridge event codes are:

Bridge Event Codes

VA	Anchored
Va	Raised anchor
VB	Giveway vessel failed to manuever
VD	one port2port,other stbd2strb
VL	failed to detect other vsl
VO	Overtaking too close
VR	rogue vessel in wrong lane
VU	Uncoordinated manuever
Vu	Non-causal Manuever
V_	Other/unknown manuever

The fields specific to this category are

detect_mi Range in miles at which this ship first detected other ship. If you don't know the distance, but you know this ship detected other ship in plenty of time to avoid collision given proper response, then enter `Y`. If you don't know the distance, but you know this ship did not detect other ship in time to avoid collision given proper response, then use `N`.

what does this mean in 3+ ship collisions

ais Y if this ship identified other ship via AIS.

rule Rules of the Road flag coded as follows.

G	this ship is give-way vessel
g	this ship probably give-way vessel
S	this ship is stand-on vessel
g	this ship probably stand-on vessel
H	this ship required to alter to starboard
A	rules are ambiguous
	Dont know

Use `A` only when it is clear the rules are ambiguous.

The alter Sub-element

The CDB allows us to record the sequence of course and speed changes prior to the casualty. This can be done via <alter> sub-elements within the <event> elements. See the sample at the beginning of this section. The <alter> sub-elements must be in chronological order. The <alter> fields are:

tod Local time of day of this <alter>. Required but may be blank. Either *hhmm* or *hhmmss* or blank. Blank means sometime after preceding <alter> time and sometime before following <alter> time. If seconds are ommitted, processing software is allowed to assume `00`. The date is based on the `date` of the parent <event> element. If the times roll into the next day, processing software is required to adjust the date according: for example an <alter> at `125645` followed by an <alter> whose `tod` is `000230`.

course Course (degrees true) at this `tod`. If only have estimate, can also code as compass points; but only to three characters, eg NNE. If missing, processing software will assume no change. If blank, processing software will assume, don't know.

Is this what we want?

spd Speed over the ground at this `tod`. Code as in <event> element. Note use of `G` or `W` suffixes to indicate speed over the ground or speed thru the water.

helm Helm order at this `tod`. The codes for this field are.

*nn*S	*nn* degrees starboard
*nn*P	*nn* degrees port
s	starboard
S	hard starboard
p	port
P	hard port
M	midships
	Dont Know

If missing, processing software will assume no change. If blank, processing software will assume, don't know. Helm order is not necessarily ship response. The helm order could be hard starboard but ships goes port because of say bank effect. If twin screw and the rudder orders are different, code as a comma separated pair, port first.

rudder Rudder angle in degrees. Plus means ship will turn to port. If twin screw and the rudder angles are different, code as a comma separated pair, port first. not CSR???

engine Engine order at this `tod`. Coded as follows.

FA	Engine(s) full ahead
HA	Engine(s) half ahead
SA	Engine(s) slow ahead
DA	Engine(s) dead slow ahead
ST	Engine(s) stopped
AS	Engine(s) astern
SS	Engine(s) slow astern
HS	Engine(s) half astern
FS	Engine(s) full astern
	Dont know

If missing, processing software will assume no change. If blank, processing software will assume, don't know. If ship is twin screw, and engine orders are different, code *pp*,*ss* where *pp* is the port engine order and *ss* is the starboard engine order.

rpm Main engine RPM. Minus means going in reverse. If twin screw and the RPM's are different, code as a comma separated pair, port first.

lat The ship's latitude at `tod`. . Coded as in <casualty>. where is acc

long The ship's longitude at `tod`. Coded as in <casualty>.

detect Use a `detect` field in an <alter> element to record the time that the ship detected the other ship. Coded as follows.

R	Radar detection
V	Visual detection
L	Lost detection
I	Intermittant

contact Use a `contact` field in an <alter> element to record attempts to agree on a collision avoidance maneuver. Coded as follows.

P	Requests Port to Port
p	Agrees Port to Port
q	Refuses Port to Port
S	Requests Starboard to Starboard
s	Agrees Starboard to Starboard
t	Refuses Starboard to Starboard
B	Requests burdened to give way
b	Burden agrees to give way
c	Burden refuses to give way
f	Fails to respond

note Free form text.

If the <alter> elements are complete enough, processing software should able to plot the ship's course.

Today, all big ships are required by law to carry Voyage Data Recorders. Yet this information seems to simply disappear. The goal of the <alter> element is to change that. However, the <alter> sub-element does not record environmental conditions. This is done in the <event> element iteslf.. If you need to record a change in environmental conditions, for example, a change in visibility, then you will need to start a new event.

One could argue logically that every <alter> is an event. But since the data set for an <alter> is much smaller than that for an <event>, it is far more efficient to group this data as a sub-element. There is nothing to stop you from starting a new <event> for every <alter> although in most cases this would be silly.

There is a problem in assigning causality to casualties in which the underlying cause was lack of coordination of both ships' maneuvers, usually through lack of communication. This can be handled two ways.

1. If it is clear that the preponderance of the blame can be laid on one ship, then that ship's Bridge event should be deemed causal.

2. Otherwise, arbitrarily (if necessary) assign a cause code of `P` to one ship's bridge event and a cause code of `N` to the other, The event codes will be `VD`, `Vd`, or `V_` in which case processing software is assumed to be smart enough to treat both event codes the same. kluge??

2.4.10 Navigation Errors

The CTX CDB attempts to distinguish between navigation errors and guidance or conning errors. In a navigation error, the ship is not where it thinks it is.[14] In a guidance error, the ship is where it thinks it is, but gets into trouble anyway because someone misjudged the current or made some other kind of conning error. If GPS properly used would have almost certainly avoided the casualty, then it is probably a navigation error. If there's still a substantial chance of the casualty happening, even if the ship had GPS, then it is probably a guidance error.

In the CTX CDB, navigation errors refer to problems internal to the ship and its ownership. Problems external to the ship such as navigation aids not properly functioning or erroneous charts are treated as *external* events. See Section 2.4.16.

The navigation error codes are:

	Ship Navigation Error
NA	Navigation error
NC	Bad charts on-board, ship fault
NS	Hit marked submerged object

`NA` and `N_` are really just a special kind of Bridge event, and as such these <event> elements may contain a sequence of <alter> sub-elements.

[14] The exception is `NC`, the ship has been supplied by the owner with out of date or inadequate charts, or has not maintained the charts correctly.

2.4.11 Guidance/Seamanship Errors

The Guidance/Seamanship category encompasses on-board errors or problems that don't fit into any other catgory, including conning errors.

We need to separate the crew related events from non-crew. Put them is a different category, maybe O for owner.

The guidance/seamanship event codes are:

Guidance, Seamanship Errors

CODE	LABEL	CAUSAL?
GA	Anchored dragged	N
GB	Hit Berth, mooring/unmooring	Y
GC	Conning error, misjudged current, turn	Y
Gc	Classification Society change	N
GD	Ship too deep for water depth, swell, channel	Y
GE	Engine dept error	Y
GR	Bad Routing by Service	Y
GS	Bad seamanship, deck	Y
Gs	Survey or Inspection	N
G_	Other Guidance or Seamanship error	Y

The fields specific to these codes are to these event codes.

class_old For the `Gc` code, `class_old` is the Classification Society prior to the switch. Coded in the same manner as `class` in the <ship> element. If all you know is the year that the Class change was made, you may use *yyyy*`0000` in the `date` field.

class_new For the `Gc` code, `class_old` is the Classification Society prior to the switch. Coded in the same manner as `class` in the <ship> element. If Class is withdrawn, but there is no record of any replacement, use `??`. `??` is quite different from blanks in this field, which means we have no reason to beliveve the ship was not classed, but we don't know which Class it wass.

survey_type For the `Gs` code, `survey_type` is the type of survey coded as follows

A Annual Survey
I Intermediate Survey
S Special Survey

class For the `Gs` code, `class` is the Classification Society which performed the survey. Coded in the same manner as `class` in the <ship> element.

The guidance errors that are bridge errors may contain a sequence of <alter> elements to describe the ship's maneuvers.

2.4.12 Ignition Events

In the CDB, a fire or explosion is always a consequence. It is never a cause. However, the ignition event can be causal. The ignition event codes are:

	Ignition Events
FE	Ignition upon Entry
FH	Ignition by Hotwork
FL	Lightning strike
F_	Other Ignition Event

If the conflagration resulted from a structural failure or machinery fault or collision or grounding or the like, then there is no need to code a separate ignition event.

FE covers cases where the crew entered the tank and it then ignited. Usually, we don't know why: maybe a dropped tool, or, since any combustible atmosphere will quickly render a human unconscious, maybe a falling body.

FL is an exceptional event. As a rule, the CDB does not regard "heavy weather" or "rogue wave" or other "acts of God" as causes. They are not an error, defect or failure. Rather they are recorded as ambient conditions. A purist would take the same view of lightning. After all a lightning strike on a properly inerted and designed ship should not do any damage. But for now we are allowing this to be part of the cause chain.

F_ is a catch-all for ignition events for which we don't have a more specific event code. It should be used when we have some idea of the event, but it doesn't match any of the other ignition codes. Otherwise use U_.

The key here is to use the causal factor fields, Section 2.4.4, to record the circumstances surrounding each fire in as much detail as possible. Many of the causal factor felds were designed for just this purpose.

2.4.13 Cargo Handling/Transfer Errors

The Transfer event category is aimed at cargo handling errors, usually at a terminal. The cargo handling/transfer event codes are:

Cargo and Transfer Errors

Code	Label	Cause?
TD	Deballasting screw up	Y
TH	Hose break/leak during transfer	Y
TF	Cargo on fire/overheated	Y
TL	Excessive Loading	Y
TO	Tank Overflow	Y
TP	Tank over/under pressurization	Y
TS	Cargo Shifting	Y
TU	Came unmoored not by wake	Y
TW	Unmoored by wake	Y
T_	Other/unknown transfer screw up	Y

Hose vs Chicksan?

Currently, there are no fields specific to these event codes. TH can be causal, for example, a defective hose bursts; and it can be non-causal, for example, hose parts as a result of the ship coming unmoored.

The fields specific to transfer events are

terminal Indicates whether this was a terminal or other shore-side screw up.

Y	Almost certainly terminal error/failure
P	Probably terminal error/failure
M	Suspect terminal error/failure

If this field is missing or blank, applications should assume, it was a ship-side error.

2.4.14 Intentional Casualties

Occaisionally, a crew or owner will generate a casualty on purpose. The Intentional event category encompasses both fraud and jettisioning cargo in an attempt to save the ship. The Intentional event codes are:

	Intentional
IJ	Jettisoned cargo/bunkers to lighten/save ship
IN	Other intentional Discharge
IS	Intentionally Sunk

Currently, there are no fields specific to these event codes.

2.4.15 War and Piracy Events

Historically by far the most important cause of tanker casualties is war. Sadly both Jihadist attacks and piracy are bring this category back to prominence. The goal of the attack event fields is to accumulate data which might be useful in developing counter-measures, although the basic problem here is attitude and will.

Here's a sample attack event. Notice that the `dead` and `hurt` fields do ***not*** include casualties among the attackers. separate fields?

```
<event ec="AW" sid="1" tod="0800"   Cause="P"     Sure="5"
       war="JH"     attacker="jh"   impact="MS"   dead="1"   hurt="-2"
       weapon="IED" platform="BOAT" sensor="NONE" range="AL" hit="H"
       note="appoaching Yemen SBM, jihadists detonated launch alongside"/>
</event>
```

The Attack event codes are:

External (non-ship) Codes		
AP	Piracy	Y
AW	War attack	Y

Currently, the fields specific to these event codes are.

war For attacks taking place during a war, the `war` field identifies the war coded as follows.

CW	Cold War
DS	Desert Storm
II	Iran-Iraq War
JH	Jihadist Conflict
W1	World War I
W2	World War II
YK	Yom Kippur War
	None or Dont know

attacker For attacks, this is either an ISO-3166 country code or `jh` for Jihadist, or `pr` for Pirate.

weapon The type of weapon used in an attack coded as follows. why not weapon itself?

ARMS	Small Arms/RPG
BOMB	Bomb
GUN	Artillery
IED	Improvised explosive
MINE	Mine
MISS	Missile
TORP	Torpedo
	Dont Know

platform For attacks, the platform via which the weapon was delivered, coded as follows.

AIRC	Fixed wing air-craft
BOAT	Small boat/launch
FROG	Swimmer
HELO	Helicopter
LAND	Land
SHIP	Surface Ship
SUBM	Submarine
	Dont Know

sensor For attacks, the sensor or guidance system the weapon used coded as follows.

do we want a weapon name field?

ASON	Active Sonar
HEAT	Heat-seeking
LASR	Laser guided
MAGN	Magnetic
NONE	None
PRES	Pressure
PSON	Passive Sonar
RADR	Radar
WIRE	Wire guided
	Dont Know

based For attacks, the ISO-3166 country code of the country from which the attack was launched.

impact For attacks, the center of impact of the weapon coded in the same manner as `impact` in <collision>. See Section 2.4.19.

range For attacks, the range at which the weapon was launched coded as follows:

VL	More than 50 miles
LR	10 to 50 miles
MD	1 to 10 miles
AL	alongside, 0 to 1 miles
BD	Boarded
	Dont Know

hit For attacks, The success of the attack coded as follows:

H	Hit, exploded
F	Hit, failed to explode
M	Missed
	Dont Know

Use damage event elements, Section 2.4.7, to record location and extent of damage.

As always, an <event> element can refer only to a single ship. If two or more ships were hit in the same attack, you will need multiple <event> elements to describe the attack. To link these <event> elements together to indicate that they are really part of the same attack, put a same valued cid <cid> field in each such element. The value used need only be unique among <cid> fields in this casualty.

2.4.16 Other External Events

This Other External event category is a catch all for casualties caused by entities external to the ship other than war or piracy attacks. For CDB purposes, the owner, the ship's classification society, and the ship's flag state are ***not*** external entities. Often the external cause involves a port or coastal state screw-up, such as a navaid out of position or a channel that has not been maintained as charted.

Here are a couple of external events.

```
<event ec="ER" sid="1" coastal="ES" cause="S"
       note="counter-flooded, increased stress but spillage stopped."/>
<event ec="ER" sid="1" coastal="PT"
       note="Spain, Portugal forced offshore anyway."/>
```

The External event codes are:

	Other External (non-ship) Codes	
EA	Coastal state provided refuge	N
EB	Navaid out of position, inoperative	Y
EC	Charts incorrect	Y
ED	Bad channel depth	Y
EG	Tug problem/screw up	Y
ER	Coastal state refused refuge	Y
ES	Hit unmarked submerged object	Y
ET	External Tampering/Robbery	Y
E_	Other external problem	Y
Ea	Coastal state attack	N
Eb	Coastal/port state blacklisted	N
Ed	Coastal/port state detained	N

Currently, the fields specific to these event codes are.

coastal For the coastal state codes, you must identify the coastal state involved using the 2 character ISO 8166 code. You may have only one coastal state per event. If the ship was refused refuge by more than one country, then you must have an `ER` event for each such refusal. You can also use this field to identify which country was responsible for problems with buoys and other navaids.

lod Length of detention in days. Applies to the `Ed` event.

nod Number of deficiencies. Applies to the `Ed` event.

The `Ea` event refers to a deliberate coastal state attack for the purposes of reducing environmental damage. For this event, all the attack fields can be coded.

2.4.17 Fire or Explosion

The explosion/fire event codes are

XA	Accommodations Fire
XE	Engine Room Fire/Explosion
XP	Pump Room Fire/Explosion
XT	Cargo Tank Fire/Explosion
X_	Fire/Explosion. Cant say where

Explosions and fire are consequences. These events cannot be causal. The CDB does not attempt to make a distinction between an explosion and fire. This manual uses the term *fire* to cover both. The meaningful distinction is where did the conflagration start: cargo area, pump room, or engine room?

Here is an example fire event together with the preceeding, causal hull failure.

```
<event ec="HR" sid="1" Cause="P" Sure="3" Sbt="P" Rust="Y" BTM="Y"
       Sbt_rust="Y"    Owner_care="Y"     Class_care="Y"
       note="Appalling rust in leaking uncoated segregated ballast tanks.
             1977 SS, BV passed steel 50 pct wasted, anodes not replaced.
             Total knew condition, did nothing since ship was to be sold.">
   <tank code="3P" coat_extent="N"
         note="badly corroded, no coating, anodes down to bare wire."/>
   <tank code="3S" coat_extent="N"
         note="badly corroded, no coating, anodes down to bare wire."/>
</event>
<event ec="XT" sid="1" tod="0100" dead="50" hurt="-2" IGS="P"
       Cgo_leak="p" Inerting="Y" Rust="Y">
       note="Discharging Bantry Bay. No Loadicator.
       note="10 ys after Mactra et al, Total ship not inerted.
             All 42 crew, 1 wife, and 7 terminal employees murdered">
   <tank code="3P"/>
   <tank code="3S"/>
</event>
```

The fields specific to the fire events are:

response Free form description of the response. We may try to make this machine readable in the future, so if you know the type of fire fighting system that was used, put it here.

An explosion/fire event may have one or more <tank> elements, which should be the compartment(s) where the fire is believed to have started. You may use `DECK`, `ACCOM` and `MANIFOLD` as appropriate. These elements may

have the same compartment condition fields as the <tank> sub-elements in the <damage> element. See Section 2.4.7.

An event sequence may have as many fire events as needed. If a fire started in the pump room and then spread to the cargo area, this should be recorded as a `XP` event followed by an `XT` event. Do not use `X_`.

2.4.18 Groundings

The grounding event codes are

WS	Grounding
Ws	Near-miss grounding

A grounding is a consequence; it can never be causal. The Sea Empress casualty involved a series of groundings. Here is her event sequence.

WP powered, WN NUC, WB beached?

```
<event ec="GC" sid="1"  Cause="P" Sure="5" GPS="P"
       note="pilot cuts corner, misjudges tidal set, worried abt stern swing"/>
<event ec="WS"   sid="1"   date="19960215" tod="2007" nuc="N"      swell=" "
       str="N"   spd="10G" squat="0.8"     acc="A"    lat="51.670" long="-5.167"
       hop="3.7" lop="200" depth="17"      chart="17" cur_spd="0.8" cur_dir="ESE"
       note="hits stbd side, 10 knots, massive venting, nearly low tide">
    <tide datum=""  next_hi_tod="0342" next_hi="6"
          at_hit="" next_lo_tod="2139" next_lo="0"/>
</event>
<event ec="HL" sid="1" vol="3000000" mat="C" hbl="Y" DS="Y" DB="P"
       note="8 min to stop, damage all along stbd side, 2500t spill" />
    <tank code="FP"
          note="generally holed but no data, maybe not initial grounding"/>
    <tank code="1C" note="deformed, small hole on bottom fwd">
        <hi xs="" ys="" zs="0.0"/>
        <lo xs="" ys="" zs="0.0"/>
    </tank>
    <tank code="2C" note="deformed, small hole on bottom fwd, stbd corner">
        <hi xs="" ys="" zs="0.0"/>
        <lo xs="" ys="" zs="0.0"/>
    </tank>
    <tank code="4C" note="deformed but no hole HOW TO CODE"/>
    <tank code="5C" note="holed all along stbd bhd, probably very low"/>
    <tank code="6C" note="holed along stbd bhd, low, also port side later"/>
    <tank code="1S" note="holed entire length of tank, ht may be low">
        <hi xs="" ys="" zs="3.1"/>
        <lo xs="" ys="" zs="0.0"/>
    </tank>
    <tank code="2S" note="holed entire length of tank, 3.1 m fwd, 1.7 m aft">
        <hi xs="" ys="" zs="3.1"/>
        <lo xs="" ys="" zs="0.0"/>
    </tank>
    <tank code="3S" note="holed entire length of tank, 1.7 m fwd, 3.6 m aft">
        <hi xs="" ys="" zs="3.6"/>
        <lo xs="" ys="" zs="0.0"/>
    </tank>
    <tank code="4S" note="holed entire length of tank, 3.7 m fwd, 2.9 m aft">
        <hi xs="" ys="" zs="3.7"/>
        <lo xs="" ys="" zs="0.0"/>
```

```
    </tank>
    <tank code="5S" note="holed at fwd end">
        <hi xs="103.0" ys="-15.0" zs="1.0"/>
        <lo xs="132.4" ys="-12.1" zs="0.0"/>
    </tank>
    <tank code="FOS" note="leaking slowly, but no data"/>
</event>
<event ec="DR" sid="1"  date="19960216" tod="0127"
       note="max draft 23m, counterflooded, anchored in Pool" />
<event ec="ER" sid="1"  coastal="GB" Cause="S" Sure="5"
       note="Harbor master rules prevent bringing ship in.
             Turned 5000 t spill into 72,000 t spill"/>
<event ec="WS" sid="1"  date="19960217" tod="1805" nuc="Y"
       str="Y" days="1" spd="" acc="A"  lat="51.677" long="-5.175"
       hop=""  lop=""   depth="15" chart="15"  cur_spd="" cur_dir=""
       note="regrounded after ship turned, ballasted down" />
     <tide datum=""  next_hi_tod="0530" next_hi="6"
          at_hit="" next_lo_tod="2334" next_lo="0"/>
</event>
<event ec="DR" sid="1"  date="19960218" tod="0800"
       note="refloated herself" />
<event ec="WS" sid="1"  date="19960218" tod="0840" nuc="Y"/>
       str="Y" days="1" spd="" acc="A"  lat="51.672" long="-5.163"
       hop=""  lop=""   depth="18" chart="18" cur_spd="" cur_dir=""
       note="">
       <tide datum=""  next_hi_tod="1755" next_hi="6"
             at_hit="" next_lo_tod="1158" next_lo="0"/>
</event>
<event ec="DR" sid="1"  date="19960218" tod="1500"
       note="refloated herself" />
<event ec="WS" sid="1"  date="19960218" tod="1715"/>
<event ec="DR" sid="1"  date="19960219" tod="0810"
       note="refloated herself" />
<event ec="WS" sid="1"  date="19960219" tod="0830"/>
<event ec="DR" sid="1"  date="19960219" tod="1715"
       note="refloated herself" />
<event ec="WS" sid="1"  date="19960219" tod="1815" nuc="Y"
       str="Y" days="1" spd="" acc="A"  lat="51.672" long="5.163"
       hop=""  lop=""   depth="11" chart="11" cur_spd=""   cur_dir=""
       note="11m depth at low water, most oil spilled on this ebb tide"/>
       <tide datum=""  next_hi_tod="1839" next_hi="6"
             at_hit="" next_lo_tod="0103" next_lo="0"/>
    </grounding>
<event ec="HL" sid="1" vol="81400000" mat="C" hbl="Y" DB="N" DS="P"
       note="We put remaining volume in this grounding"/>
     <tank code="3C" note="large hole, bottom cl, not on 1st grounding">
        <hi xs="" ys="" zs="0.0"/>
        <lo xs="" ys="" zs="0.0"/>
      </tank>
```

```
      <tank code="1P" note="holed fwd, not initial grounding, no data"/>
      <tank code="4P" note="holed aft, not initial grounding, no data"/>
      <tank code="5P" note="holed fwd, not initial grounding, no data"/>
      <tank code="SLOP_P" note="badly holded, not initial grounding, no data"/>
      <tank code="PR" note="5m flooding????, holed port side, not initial/">
</event>
<event ec="DR" sid="1"  date="19960221" tod="1800"
       note="But some came from blowing out to refloat"/>
<event ec="DT" sid="1"
       note="towed into berth on a draft of 11.95m"/>
```

Notice how the hull damage and the spillage is split as best we can between the initial grounding and the subsequent groundings. It was the fourth grounding which took place at nearly high tide in 11 meters of water in which almost all the oil was spilled. This was after the harbor master had effectively refused refuge.

The fields specific to a grounding event are:

go to y if ship never refloated

str Y if the ship is stranded for more than a tidal cycle, N if not, or blank if unknown.

days Length of stranding in days if known. Use 99 to mean longer than 99 days, or the ship was never refloated.

get rid of, redundant, see DR event.

chart The charted depth where the ship grounded at datum. This field can be either a single number (assume flat bottom), two comma separated numbers (depth forward and depth aft), or four comma separated numbers (depth forward port, depth forward starboard, depth aft port, depth aft starboard.) `depth` is measured parallel to the direction of gravity and thus can be different from draft if there is trim or heel. For channels and canals, this normally should be the controlling depth.

depth The actual depth(s) where the ship grounded at datum, coded in the same manner as `chart`. If this element is missing, processing software will use the charted depth(s).

tide If the ship is stranded for any length of time, it is essential that the key tidal data be recorded. Tide will often have a critical effect on the amount of spillage. Not to mention the ability to refloat the ship. The <tide> sub-element consists of the following fields.

datum A string explaining the datum for the following numbers. This must match the depth datum.

at_impact The tide level relative to the datum at the time the ship went aground.

next_lo_tod The local time of the next low tide in 24 hour format. Next means next tide after the ship went aground.

next_lo The tide level relative to the datum at the next low tide.

next_hi_tod The local time of the next high tide in 24 hour format.

next_hi The tide level relative to the datum at the next high tide.

In XML, the attributes of an element can be in any order. Processing software cannot assume that just because the `next_hi` stuff comes before the `next_lo`, that the ship grounded on a rising tide. The code must work it out from the time of grounding. But it makes it easier for humans to read, if these attributes are in temporal order.

The damage described in a grounding event should only be the damage resulting from that grounding.

2.4.19 Collisions

The collision event codes are

Ca	Allision, vessel hit non-vessel
Cm	Near-miss allision
CA	One vessel moored, anchored or mothership
CG	Tug contact
CI	Inter-ship interaction, no contact
CN	Collision, both vessels moving
CM	Near-miss collision

Collisions and allisions are consequences. These events cannot be causal.

iceberg, anch dragging over pipeline? prob shud be in bridge, guidance

In the CTX casualty database, a physical collision (or allision) must involve exactly two ships (or one ship and one fixed object). This allows us to compute a relative speed for each impact, etc, and eliminates a number of possible ambiguities. Collisions involving three or more ships must be recorded as a series of two ship collisions.[15]

Figure 2.6 is an event sequence in which one ship was involved in two closely related collisions.

The fields specific to the collision event codes are:

cid Since collisions (impacts) are two ship events, each collision must be represented by two <event> elements.[16] Each such <event> element must have a `cid` field whose value is unique (within this casualty) to that collision. In Figure 2.6 the first collision between the Vysotk and the Diego Silang has been given a `cid` of `1`. The second collision between the Diego Silang and Brazilian Faith has been given a `cid` of `2`.

vts This field describes Vessel Traffic System involvement with if any. It is coded as follows.

O	VTS only observed casualty
Y	VTS talked to this ship
u	VTS unsuccessfully tried to talk to this ship
N	VTS not involved
	Dont know

[15] However, it is possible to have more than two ships with the same `cid` but the 3rd, etc ships must have an `impact` of `NO`. This allows us to record "third" ships who influenced or perhaps caused the collision, but we not actually part of the physical impact.

[16] Hitting a well-marked submerged object is not a collision. It is a single ship, navigation error. See Section 2.4.10. A collision may have more than two <event> elements but only if the extra elements have `impact="NO"`.

```
<event ec="VR" sid="2" Cause="P" Sure="5" rule="H" detect_mi="Y" Crazy="Y"
       note="Diego Silang loaded eastbound Malacca is overtaking
             Brazilian Faith under tow on Faith port side.
             Vysotsk westbound inexplicably turns to port. No talk.">
   <alter tod="2331" course="310" spd="   " helm="P" engine="  "/>
   <alter tod="2333" course="304" spd="   " helm="P" engine="  "/>
   <alter tod="2334" course="290" spd="   " helm=" " engine="  "/>
</event>
<event ec="Vu" sid="1" detect_mi="10" rule="H"
       note="Diego hesitates then goes astern, bow swings to stbd">
   <alter tod="2331" course="310" spd="13.5W" helm="M" engine="  "/>
   <alter tod="2337" course="310" spd="13.5W" helm="M" engine="ST"/>
   <alter tod="2338" course="   " spd="     " helm="M" engine="HA"/>
   <alter tod="2339" course="   " spd="     " helm="M" engine="FA"/>
</event>
<event ec="CN" sid="1" cid="1" nuc="N" impact="BP"
       tod=2342" enc="H" talk="N" vts="N" VHF="M"
       note="Diego hits Vysotk on stbd aft side."
</event>
<event ec="CN" sid="2" cid="1" nuc="N" impact="AS"
       note="No fire!"
</event>
<event ec="Vu" sid="1"
       note="On impact, Diego stops engines, then goes astern.">
   <alter tod="2342" course="137" spd="    " helm="M" engine="ST"/>
   <alter tod="2343" course="   " spd="    " helm="M" engine="HA"/>
   <alter tod="2346" course="   " spd="    " helm="M" engine="FA"/>
</event>
<event ec="Vu" sid="3" detect_mi="  " rule="S"
       note="Brazilian Peace under tow, maintains course and speed">
   <alter tod="    " course="125" spd="   5W" helm="M" engine="  "/>
</event>
<event ec="CN" sid="1" cid="2" nuc="N" impact="S" angle="90"
       tod="2350" enc="V" talk="N" VTS="N" VHF="N" TS="M"
       note="Diego bow goes in front of Faith."
</event>
<event ec="CN" sid="3" cid="2" nuc="N" impact="B"
       note="Somehow must have crossed tow line.">
</event>
<event ec="HL" sid="1" vol="6200000" mat="C" DB="N" DS="M"
       note="Diego hit on stbd side by Faith, 6000t spill" />
```

Figure 2.6: Sample Multiple Collision Casualty

enc The type of encounter coded as follows.

A	This ship anchored/moored
a	Other ship anchored/moored
B	Crossing, this ship giveway
b	Crossing, this ship probably giveway
c	Probable crossing
H	Confirmed Head on or near Head on
h	Probable head on or near head on
L	Allision
P	Crossing, this ship stand-on
b	Crossing, this ship probably stand-on
S	STS Mooring/unmooring
V	This ship overtaking
v	This ship overtaken
	Cant say encounter type

Notice the encounter type field can be used to separate out allisions.

talk This field describes the level of communication between the ships prior to the collision.

A	Ship attemped unsuccessfully to communicate
N	Ship made no attempt to communicate
Y	Ship did communicate
F	Ship failed to respond to communcation
	Cant say if ship communcated

impact Center of impact coded as follows:

B	center of impact on bow
BP	center of impact on port bow
BS	center of impact on starboard bow
FP	center of impact forward, port side
FS	center of impact forward, starboard side
MP	center of impact midships, port side
MS	center of impact midships, starboard side
AP	center of impact aft, port side
AS	center of impact aft, starboard side
T	center of impact on stern
P	All we know is impact on port side
S	All we know is impact on starboard side
NI	Interaction effect but no impact.
NM	Near Miss.
NO	No impact.
	dont know

Use `NO` for third ships which may have caused or influenced the collision, but were not involved in the actual impact.

angle If this ship is the hittee, the angle at which she was penetrated (0 to 180), 45 degrees means 45 degrees from her bow, 90 meaning she was struct perpendicular to the centerline, 135 degrees means 45 degrees from the stern. If you don't know the angle accurately enough to assign a number, you may code as follows.

L	Less than 30 degrees
M	30 to 60 degrees
H	60 to 120 degrees
m	120 to 150 degrees
l	Greater than 150 degrees

The CDB regards an allision as simply a collision with a non-ship object or maybe a vessel which we choose not to regard as a ship. Is contact with a moored barge a collision or an allision? It depends on how much emphasis we want to give to the barge. If it is a large, loaded petroleum barge which blows up and a lot of people are killed, we probably want to treat it as a ship. If it's an empty derelict that has been sitting on a river bank for years, we probably want to treat it as a non-ship.

The CDB can go either way. Either way the object collided with should be represented by a (poorly named) <ship> element with its own `sid`. If the object is treated as a non-ship, its <ship> element must have a ship type field of `NV` (Non-vessel), and the other ship specific fields can be missing or blank. See Section 2.5.

2.4.20 Lifevessel Events

The purpose of the lifevessel events is to document the role and performance of lifeboats/liferafts in the casualty. To document rescue attempts from outside the ship, use a Rescue event. See Section 2.4.21. If such an attempt involves the use of the rescue ship's lifeboats, you can use these events to document the performance of those boats; but only with the rescue ship's `sid`.

Lifevessel Events

CODE	LABEL	CAUSAL?
Lb	Lifevessel burned	Y
Lc	Lifevessel capsized	Y
Le	Lifevessel engine failure	Y
LF	Lifevessel found	N
LL	Lifevessel successfully launched	N
Ll	Lifevessel could not be launched	Y
Ls	Lifevessel swamped/sunk	Y
L_	Lifevessel other/unknown	Y

The fields specialized to lifevessel events are:

vessel Type of lifevessel, coded as follows

BE	Enclosed lifeboat, davits
BO	Open lifeboat, davits
BF	Freefall lifeboat
BP	Pod, single falls
BR	Rescue boat
B_	Lifeboat unknown/other
RI	Raft inflatable
RR	Raft rigid
R_	Raft other/unknown

davits Lifeboat/raft's location on board. Coded in the same manner as `impact`.

drill Indicates whether event took place during a life boat/raft drill. Coded as follows.

Y	Event took place during a drill
P	Probably took place during a drill
M	Suspect took place during a drill
N	Event did not take place duirng a drill

If this field is missing, applications will assume the event did not take place during a drill.

saved Number of lives saved by using lifeboat/raft. Do not count if the people would have survived without using lifeboat/raft. Do count if the lifeboat/raft was an important link in the survival, even though they would not have survived without additional help.

days Number of days between launching and rescue.

found This field applies to the `LF/lF` events. Coded as follows.

N	Lifeboat/raft never found
E	Lifeboat/raft found empty

Use N to indicate we have good reason to believe that lifeboat/raft was successfully launched, but was never located.

lifeboat There is a conceptual problem associated with lifevessel events. It is difficult to regard "inability to launch lifeboats due to fire" as causal in the same sense as the cause of the fire. On the other hand, if the boat could have been launched, lives would have been saved. Currently,the CDB temporizes on this issue. A lifevessel event is regarded as causal only if it is in the casualty ***initiating*** chain, e.g. hook releases during drill. But to allow exploration of all casualties involving lifevessel events via the web interface, all lifevessel events, should include a `lifeboat` field. The code is the same as the second letter in the event code. You can regard this as sort of strange causal factor.

We must do a better job of this. Maybe a new cause category. But we also want to see casualties where the lifevessel worked.

The following *reason* fields apply to the `Ll/ll` event They attempt to pin down the reason why the lifeboat/raft was not or could not be successfully launched.

fire Either the lifeboat/raft was involved in the fire or fire prevented the crew from reaching the lifeboat/raft.
Y/P/M/N. Yes/Probably/Suspect/No.

list List prevented the lifeboat/raft from being launched, or the ship capsized or sank, before the lifeboats/rafts could be launched.
Y/P/M/N. Yes/Probably/Suspect/No.

away The lifeboat/raft has been carried away or damaged before it could be launched.
Y/P/M/N. Yes/Probably/Suspect/No.

hook Hook failure. This field takes two forms. Lifeboat hook failed to release.
y/p/m/n. Yes/Probably/Suspect/No.
Lifeboat hook released prematurely or unexpectedly.
Y/P/M/N. Yes/Probably/Suspect/No.

inflate Inflation failure. This field takes two forms. Liferaft failed to inflate.
y/p/m/n. Yes/Probably/Suspect/No.
Liferaft inflated prematurely.
Y/P/M/N. Yes/Probably/Suspect/No.

sea Heavy seas prevented the lifeboat from being successfully launched.
Y/P/M/N. Yes/Probably/Suspect/No.

These reason fields tend to be more factual than a Causal Factor, so they are not capitalized. Obviously, you can assign any of the ordinary Causal Factor fields to lifeboat/raft events, as well as any reason field.

Some lifevessel events are essentially machinery failures, e.g. hook releasing prematurely. Such events may have have one or more <component> elements describing the component that failed. See Section 2.4.8 for details.

Ideally, you should have an event for each lifeboat/liferaft successfully or unsuccessfully employed. But in some cases, you will just have to lump the numbers together.

Figure 2.7 is an example of a lifeboat drill casualty.

```
<casualty
  id="20041006_001" date="20041006" Edu="6"  site="anch off Port Hedland "
  locale="R"        tod="1520"      acc="B"  lat="-20.118" long=" 118.565"
  coastal="AU"      area="in,EI,   ,     "   weather="CM"         vis="GD"
  note="lifeboat drill, wasted aft keel stays fail, 2 killed, 3 hurt     ">
  <event ec="Ll" sid="1" Cause="P" Sure="1" drill="Y" dead="2" hurt="3"
         vessel="BE" davits="AP" hook="Y"
         Maint="Y" Owner_care="Y" Class_care="Y" Design="Y"
         note="lifeboat drill, badly wasted aft keel stays fail, 2 killed">
      <component er=" " sfi="501   " name="aft hook keel stays"
            maker="Blue Sea" model="24LE"
            power="" hours="" survey="20040407" ttr=""
            failure="badly wasted 15 mm mild steel keel stays failed"
            note="stays, keel block should have been stainless">
      </component>
      <component er=" " sfi="501   " name="fwd on-load release hook"
            maker="Umoe Schat-Harding" model="Titan"
            power="" hours="" survey="20040407" ttr=""
            failure="boat rotated, fwd hook released, boat fell 16 m"
            note="hook design not robust, yard used wrong ring">
      </component>
  </event>
  <ship sid="1"  imo="8911499" class="BV"  name="lowlands grace"
        st="BC"  dwt="149518"  yob="1991"  flag="HK" tnks=" 9" pob=" "
        ht="  "  grt="      "  status="B"  cgo="  "  sts=" "
        ns="1"   crew="25"     ig=" ">
     <load draft_fp="7.10" draft_ap="7.84"/>
  </ship>
</casualty>
```

Figure 2.7: Sample lifeboat drill casualty

2.4.21 Rescue Events

The purpose of the rescue events is to document rescue attempts from outside the ship. Do not use these events for the ship's own efforts to save herself or her crew.

Rescue Events

CODE	LABEL	CAUSAL?
RH	Helicopter Rescue	N
RS	Ship Rescue	N
RB	Boat Rescue	N

It is good practice to create an `sid` for each significant rescuer; but if the identity of the rescuer is not significant or not known, you can cheat by using the ship's own `sid`. This practice is strongly discouraged.

We should make this illegal, but we will need to expand the ship element to handle helicopters, etc.

The fields specialized to rescue events are:

saved Number of lives saved by this rescue attempt. Do not count rescuers that have to be rescued themselves. Do count a life even if you have already included this life in the lifeboat/raft `saved`. In other words, a seaman can be "saved" more than once in the same casualty. Applications must not equate the sum of all the `saved` fields with the total number of survivors. The objective here is to measure the effectiveness of each step in the rescue process.

If the rescue attempt involves the rescue ship's lifeboats, you can use the lifeboat events (see 2.4.20 to document the success or failure of these attempts. But in this case these lifeboat events must have the `sid` of the rescue ship, not the ship being rescued.

Rescue attempts can result in deaths/injuries to rescuers. If this is the sad case, put these deaths in the most appropriate event.

2.4.22 Disable Event Codes

The *disable* events are for the most part responses to a casualty, but some of them can be causal. The Disable event codes are: blowout?

Disable Event Codes

DA	Disabled, anchored	N
DC	Cargo Transferred Internally	N
DE	Escort Tug Used	N
DF	Ballasted down/Counterflooded	N
DL	Lightered/lightened	N
Dl	Listed, no self-recovery	N
DP	Loss of Propulsion Power	Y
Dp	Propulsion Power Restored	N
DR	Refloated	N
Dr	Massive Roll, recovered	N
DS	Loss of Steerage	Y
Ds	Steerage Restored	N
DT	Taken under Tow	N
Dt	Tow lost/slipped	N

It is important to distinguish between `MR` and `DS`. On a single screw ship, a steering gear or rudder failure almost always results in loss of steerage. But on a twin screw ship, we can have a steering system failure without loss of steerage. And loss of steerage can occur without a steering gear failure. Most current single screw tankers and bulk carriers lose steerage whenever their speed drops below 4 or 5 knots for whatever reason.

In general, current single screw tankers and bulk carriers have very poor low speed manueverability. Sometimes this poor maneuverability can be causal. An example is the Bright Field which lost steerage at close to 10 knots through the water when she lost her main engine, ***even though the steering system was still functioning***. The Bright Field ended up ramming the New Orleans waterfront and injuring over 60 people. If a ship's maneuvering characteristics are so bad that it could not do what it would reasonably be expected to do, call this a causal loss of steerage. You probably want to include a `design` attribute in the event as well.

Sometimes a ship cannot go as slow as it would like without losing steerage, and then this ship's speed bceomes a causal factor. On the Neches/Sabine waterway ships being unmoored or damaged by "excessive" wake is an almost regular occurrance. Use the `MA` event code to record this sort of event. See Section 2.4.8.

Loss of power, `DP`, means total loss of propulsion power to the ship as a whole. Loss of one main engine on a twin screw ship is not a total Loss of Power event.

Many of the machinery failure fields are applicable to the `DP` and `DS` codes. In addition, the fields specific to these event codes are.

list Max roll/list angle in degrees. Use `Dr` is ship recovers by herself, use `Dl` if ship does not (eg lolling). You may append a `P` or `S` to indicate the roll is to port or starboard. Append nothing if you don't know,

lop_hrs Length of loss of power or steerage in hours if known. You can also use

T	had to be towed in
M	Major (LOP/steering for a hour or more)
m	Minor (LOP/steering for only a few minutes.)
N	Power/Steerage never restored
	dont know

lof Use `lof="Y"` if ship accepted a Lloyds Open Form or equivalent.

pan Made panpan call.

herself Use `herself="Y"` if ship refloated herself in a `DR` event.

section If a ship breaks in two, then you may need to identify which part is involved in a `DT` event. You can use the `section` field coded as follows.

B	Bow or forward section.
S	Stern or aft section.
M	Mid-section.

Use `M` only if the ship breaks into three sections.

vol_m3 Volume transferred, lightered, or blown out in cubic meters.

2.4.23 Fate Event Codes

The last category of events describes a ship's fate. The fate codes are:

Fate Codes		
Sr	Repaired	N
St	Resumed trading	N
SC	Contructive Total Loss	N
SD	Scuttled	N
SP	Capsized	N
SK	Foundered/sank	N
ST	Scrapped	N

Currently, the fields specific to these event codes are.

section If a ship breaks in two, then you may need to identify which part is involved in a `S` event. You can use the `section` field coded as follows.

B	Bow or forward section.
S	Stern or aft section.
M	Mid-section.

Use `M` only if the ship breaks into three sections.

distress If a ship sinks with no survivors and no distress signal was received, put `distress="N"` in the `SK` element. If this field is missing, then processing applications should assume some sort of distress signal.

sink_hrs Time between damage and sinking in hours.

If a ship is declared CTL and then scrapped, you need show only the scrapped `ST` code. You can use the `Sr` code to show that a Constructive Total Loss was repaired. If a ship's event sequence does not end with an `S` code, processing software will assume the ship survived. Use `St` if the ship did not have to divert for repairs. In most cases, the `Sr` and `St` codes are not really needed, but can alert other coders that we know the ship survived in circumstances where this might be surprising.

2.5 The Ship Element

2.5.1 Introduction

The CTX Casualty Database contains very little ship data. Ideally it would contain none, other than a field which would allow us to link to the ship's information in data bases which do focus on describing the ship rather than its casualties. However, the current sitation in ship data bases is, to put it politely, unsatisfactory. Much of the data is proprietary, or incomplete, or difficult to access. Therefore, the CDB compromises by including (where known) a few basic ship particulars for each ship in a casualty. This data is contained in the <casualty>'s <ship> sub-elements. This sub-element also contains some information on the state of the ship just prior to the casualty. Here's a sample <ship> element:

```
<ship sid="1"  imo="8414520" class="AB"  name="exxon valdez"
      st="TC"  dwt="214853"  flag="US"   yob="1986" tnks="11"
      ht="SM"  grt=" 94999"  status="L"  cgo="PC"   sts="N"
      ns="1"   ig="Y"        pob="N">
   <load draft_fp="17.07"  draft_ap="17.07" heel="0.0"
         cargo_wt="174391" ballast_wt="0"   fuel_wt="1336"
         note="djw1 guess to give 56 ft draft, even keel, min fuel to LA">
       <tank code="1C"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="2C"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="3C"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="4C"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="5C"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="1P"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="1S"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="3P"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="3S"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="5P"     pct="83.24" sg="0.897" temp="35"/>
       <tank code="SLOP_P" pct="64.97" sg="0.897" temp="35"/>
       <tank code="SLOP_S" pct="64.97" sg="0.897" temp="35"/>
       <tank code="FO_P_A" pct="15.00" sg="0.980"/>
       <tank code="FO_P_F" pct="15.00" sg="0.980"/>
       <tank code="FO_P_A" pct="15.00" sg="0.980"/>
       <tank code="FO_P_F" pct="15.00" sg="0.980"/>
   </load>
</ship>
</casualty>
```

There must be one <ship> element for each ship in the casualty. [17] In the case of an allision or shore-side damage, each fixed facility/object that is involved is also described by a (poorly named) <ship> element. In most casualties, there will be exactly one <ship> element; but the CDB allows as many ship elements as you need.

The order of the <ship> elements has no meaning. Unlike some vessel casualty data bases, the CDB is even handed toward ships in multiple ship casualties. However, the first <ship> element does have a slightly priviledged position.

1. Some applications generate a one line listing of each casualty. Often there is room for only one ship name is this listing. In such situations, the application will often use the name from the first ship element. In general, applications that focus on one ship will use the first ship in the casualty element. This is more a limit of the application than the database.
2. Some simplistic analyses are based on ship age or size or type. In multi-ship casualties, it is not obvious which ship's particulars should be used. Often applications that attempt to break down casualties in this manner will use the first ship listed. Once again this is an application limit, not a database fault; but you should be aware of it in ordering the <ship> elements.

[17] It is not necessary to have a collision to have multiple ships. For example, a fire — however caused — can spread from one ship to another. See also ISI Olive, Overseas Meridian.

2.5.2 The Ship Element Fields

Each <ship> element begins with a number of fields dentifying the ship, and describing its status at the time of the casualty. Unless the description says otherwise, none of these fields are required.

The <ship> fields are:

sid The `sid` field is required and its value must match this ship's `sid` code in the <event> elements. The `sid` field need be unique only within this casualty. Even if a casualty involves only one ship, the <ship> element must have an `sid`. By convention, the value of this field in single ship casualties is `1`.

imo The CTX CDB itself contains very little permanent ship data, such as LOA, beam, etc. The seven digit `imo` field is the key for joining ship data in data bases external to the CDB to the casualty. Since ship names both change and are re-used, they cannot be used for this purpose.

For post-1980 ships, `imo` is the seven digit number assigned by IMO. For 1960 to 1990 era ships, this usually is the seven digit Lloyds Register number upon which the IMO numbering system is based. For still older ships, CTX uses the Miramar Ship Index ID. The Miramar system converts older American, British, Japanese, etc ID numbers to a seven digit index. For ships for which we have ship data but neither an IMO number nor a Miramar ID, CTX uses a seven digit number which starts with three zeros. Hopefully, this is a temporary assignment.

A <ship> may have a blank or missing `imo`. This is not uncommon when the casualty involves small coastal vessels and the like. But if this is the case, it will not be possible to link to more detailed data for this ship outside the CDB.

Despite appearances, this seven character field is not an integer. Leading zeros must be preserved.

class The ship's Classification Society at time of casualty. according to the following coding.

??	No record of any Class
AB	American Bureau of Shipping
ALL	ALL
BG	Bulgarski Koraben Registrar
BI	Biro Klasifikasi Indonesia
BV	Bureau Veritas
Bm	Belize Maritime Bureau
Br	International Registry of Belize
Bs	Bureau Securitas
CR	Croatian Register of Shipping
CS	China Classification Society
Cc	China Corporation Register
Fn	Fidenavis S.A.
GL	Germanisher Lloyd
Gm	Global Marine Bureau
HE	Hellenic Registry of Shipping
IR	Indian Register of Shipping
Ib	Isthmus Bureau of Shipping
Il	Inclamar
Im	Intermaritime Certification
In	International Register
Is	Isthmus Maritime Classification
Iv	International Naval Surveys
KR	Korean Register of Shipping
Kj	Korean Classification Society
Ks	Korean Ship Safety Authority
LR	Lloyds Register
Mo	Mongolian Ship Registry
Mt	Maritime Technical Systems
NK	Nippon Kokkan
NV	Det Norske Veritas
Om	Overseas Maritime Certification
PR	Polski Rejestr Stakow
Pc	Panama Shipping Certificate Inc
Pd	Panama Maritime Documentation
Ph	Phoenix Register of Shipping
Pr	Panama Shipping Registrar Inc
Ps	Panama Maritime Surveyors Bureau
RI	Registro Italiano Navale
RM	Registrol Naval Roman
RP	Rinave Portugesa
RS	Russian Maritime Register
Rs	Register of Shipping (Albania)
TL	Turkish Lloyd
UK	Shipping Register of Ukraine
Ub	Universal Shipping Bureau
Um	Universal Martime Bureau
VR	Vietnam Register of Shipping

`??` does ***not*** mean Class unknown. `??` means we have reason to believe that the ship was trading without being in any Classification Society. Usually, this happens when we know Class was withdrawn, but we have no record of a replacement Class. All blanks as usual in the `class` field means we have no reason to believe the ship was trading without Class, but we don't know which Class it was.

name Normalized ship name at time of casualty. Normalize names by deleting apostrophes and trailing periods, translating any non-Roman, non-ASCII character to ASCII dropping any diacritics, substitute blank for underscore or period or slash or any other non-alphanumeric character. Collapse multiple blanks to one. Truncate to 22 characters and then lower case. This normalization is designed to make searches by ship name as reliable as possible.

st Ship type. The possible values of `st` are

TC	Crude Tanker
TP	Products Tanker
Tc	Chemical Tanker (also chem and oil)
T_	Other/unknown tanker
BC	Dry bulk carrier
CB	Ore Bulk Oil carrier
CO	Ore Oiler
C_	Other/unknown combo
LN	LNG Carrier
LP	LPG Carrier
FV	Fishing Vessel
GC	General cargo
GN	Container ship
NV	Non-vessel (use for allisions)
PC	Passenger/cruise
RR	Roll-on, Roll-off, ferry
SV	Offshore Supply Boat
UG	Tug mooring
UB	Tug barge
VC	car carrier
WS	Warship
U_	Other or No info

`st="NV"` is required if the object the <ship> element is describing is a non-ship.

ht Hull type. The possible values of `ht` are

SP	Pre-Marpol Single Hull
SM	Marpol Single Hull
DB	Double Bottom
DS	Double Sides
DH	Double Hull
MD	Mid-Depth
CE	Coulombi Egg
Me	Membrane
Sp	Spherical
Pr	Prismatic

`Me`, `Sp`, `Pr` applies only to LNG carriers. "Single-skin" bulk carriers are actually double bottom ships.

dwt Ship's summer deadweight in metric tons.

yob Year delivered, four digits.

flag Country of Registry at time casualty if known. Otherwise blank. Use the ISO-3166 two character code, all caps. Use parent country for local FOC's. GB for Isle of Man. NO for Norwegian International Registry. PT for Madeira.

tnks For pure tankers, this should be the total number of cargo tanks exclusive of slop tanks. For bulkers, OBO's and ore/oilers, it should be the number of cargo holds. What we are looking for here is a measure of sub-division.

pob The field describes the ship's pilotage status at the time of the casualty.

Y	pilot/mooring master on board
N	pilot not on board
P	ship was in process of picking up pilot
D	ship was in process of disembarking pilot
	dont know pilot status

ns Number of main propulsion shafts.

Must improve, shaft gen, fully indep, etc

grt Ship's gross registered tonnage. Deadweight is a better measure of ship size, so if you know the deadweight be sure and enter it. But sometimes we have GRT but not deadweight. In this case use this field, most applications will estimate the ship's deadweight from this number if `dwt` is not given.

status Activity of this ship at time of casualty. This field may be

A	Anchored, apparently doing nothing. Use only if none of the following
L	Loaded, including part-loaded, unmooring/mooring at load/discharge port
B	Ballast, including mooring/unmooring at load/discharge port
T	Tank Cleaning, (use instead of B if appropiate)
l	Loading, including deballasting
d	Discharging
R	Repairing
S	FPSO/floating storage
b	Bunkering, but not at load/discharge port
	dont know

cgo Cargo at time of casualty. If ship is in ballast, put in last cargo if known.

is this a good idea?

	?
AL	Alcohol/MTBE
CL	Coal/Coke
CM	Cement
FH	Phosphate
FP	Potash
F_	Other/unknown fertilizer
GW	Wheat
G_	Other/unknown grain/beans
LN	LNG
LP	LPG
OI	Iron Ore
ON	Nickel Ore
O_	Other/unknown ore
PA	Asphalt/Tar
PB	Carbon black
PC	Crude
PD	Distillate
PG	Gasoline/naptha
PH	Heavy fuel
PL	Lube oil
P_	Other/unknown petroleum
SA	Salt
SG	Sugar
SM	Sulfur molten
SP	Steel products/pig iron
SS	Steel/iron scrap
SU	Sulfur dry
VM	Molasses
VP	Palm Oil
V_	Other/unknown vegetable oil

sts Ship to ship transfer indicator/ Code this field as follows.

M	Ship was a mothership
L	Ship was a lighter
R	Ship was a reverse lighter
P	Ship probably was a lighter
N	Ship not a lighter or mothership
	Dont know lighter status

A ship is regarded to be lightering for the entire shuttle trip.

ig Inert gas flag.

Y	IMO IG system installed, operating properly	
P	IMO IG system installed, probably operating properly	
p	IMO IG system probably installed, operating properly	
B	IMO IG system installed, not operating properly	
N	IMO IG system not installed	ll
n	IMO IG system probably not installed	
X	CTX IG system installed (ballast tnks inerted, dbl scrubbed)	
x	CTX IG system installed, not operating properly	
	Dont know IGS	

crew The number of crew on-board at time of casualty. Crew is define broadly, including everyone on board that did not pay for their passage. See Section 2.4.3.

pax The number of passengers on-board at time of casualty. Passenger is defined narrowly, only those on-board that paid for their passage.

note Free-form comment on the ship or its condition, anything special about the ship.

In addition, a <ship> may contain one or zero of each of the following sub-elements.

load	Use to describe pre-damage cargo load pattern
condition	Use to describe ship's condition at time of casualty
insurance	Use to describe ship's insurance coverage

Each of these sub-elements is described in the following sections. The order of these sub-elements is not significant.

2.5.3 Loading Pattern

In order to do any sort of analysis of a ship's flooding, spillage, and/or the ship's residual strength, we must know the ship's loading pattern just prior to damage. To record this vital data for a ship, use a <load> sub-element. Here's an example, once again from the EXXON VALDEZ.

```
<load date="" tob="" ref=""          alt=" "
    draft_fp="17.07"  draft_ap="17.07"    heel=" 0.0"    sag="   "
    cargo_wt="174391" ballast_wt="     0" fuel_wt="1336"
    note="jack guess to give 56 ft draft, even keel, min fuel to LA">
<!--<tank code="1C" ull=" " inn=" " vol=" " pct=" " sg=" " temp=" "/>   -->
    <tank code="1C      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="2C      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="3C      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="4C      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="5C      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="1P      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="1S      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="3P      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="3S      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="5P      " pct="83.24" sg="0.897" temp="35"/>
    <tank code="SLOP_P  " pct="64.97" sg="0.897" temp="35"/>
    <tank code="SLOP_S  " pct="64.97" sg="0.897" temp="35"/>
    <tank code="FO_P_A  " pct="15.00" sg="0.980"/>
    <tank code="FO_P_F  " pct="15.00" sg="0.980"/>
    <tank code="FO_P_A  " pct="15.00" sg="0.980"/>
    <tank code="FO_P_F  " pct="15.00" sg="0.980"/>
    <spike name="FIXED   " wt="300.0" xs=" 45.000" ys=" 0.000" zs=" 20.0"/>
</load>
```

The <load_pattern> attributes are:

date It is possible to have more than one loading pattern. If you do have more than one, than you must specify the `date` of all but one of the loading patterns. If `date` is missing processing software will assume it is the pre-damage loading pattern.

for damage loading patterns, how to record both oil and seawater volumes?

tod If you need to distinguish loading patterns by time of day, use `tod`.

ref Freeform description of the source for the data.

alt Alternate hold loading flag. This field applies only to dry bulk cargos.

Y	Ship was alternate hold loaded
y	Ship was nearly alternate hold loaded If you dont know, this
N	Ship was not alternate hold loaded
	attribute should be missing.

draft_fp Pre-damage draft at the Forward Perpendicular in meters.

draft_ap Pre-damage draft at the Aft Perpendicular in meters.

heel Pre-damage heel in degrees. Heel to starboard is positive.

sag Pre-damage sag in m. Negative is hog.

backwards from JTP?

cargo_wt Cargo deadweight at time of casualty in metric tons.

ballast_wt Ballast at time of casualty in tons.

fuel_wt Fuel on board at time of casualty in tons.

note Any commentary on the data.

If you only know one draft, put it in `draft_fp` and leave `draft_ap` blank. Processing software will assume this is the maximum draft. Draft is measured in ship coordinates, that is, perpendicular to the flat bottom. Draft may be different from depth, which is measured in earth coordinates, that is, in the same direction as gravity.

The <load> element should contain at least one <tank> sub-element for each non-empty compartment. Analysis programs will assume that any tank or compartment not listed was empty just prior to the casualty.[18] The attributes of this element are:

code The name by which the tank is known in the ship's capacity plan. Must match the name of this compartment in the <damage> element.

pct The volume of the liquid in the tank as a percentage of the 100% full volume. Alternatively, you may specify ullage (ull="xx.xxx") in meters, or innage in meters (inn="xx.xxx") or absolute volume in cubic meters (vol="xxxxx.x"), or weight in metric tons(wt="xxxxx.x"). You must specify exactly one of these attribute for each non-empty tank.

sg The specific gravity of the liquid in the tank. If there is no `temp` attribute, this must be the density at the tank temperature. If there is a `temp` attribute, this must be density at standard temp (15C), and analysis programs will use temp to correct the density. Alternatively, you may specify API (api="xx.xx") in which case you must specify temperature. Either `sg` or `api` is required.

[18] Except if there are no <tank> elements at all. In this case, analysis programs should assume, we don't know anything about the ship's loading pattern, other than the drafts and heel.

temp Tank temperature in degrees Celsius. If this attribute is given, processing software will attempt to convert the standard density/API to ambient density using this temperature.

Do we require layers in order bottom to top?

If a tank is damaged, it may contain both cargo and seawater. In this case, use multiple <tank> elements with the same tank code. Processing software will assume the tank is layered with the densest liquid on the bottom. The volumes/percentages apply only to the volume of this particular liquid in the tank.

The <load> element may also contain one or more <spike> sub-elements. <spike>'s are used to represent loads that count against deadweight, but are regard to be fixed in space. This may include stores, consumables, crew, and sometimes very small tanks.

The attributes of this element are:

name Any reasonable name for the load, eg "STORES".

wt The weight of the load in metric tons.

xs The longitudinal center of gravity of the load, forward of the AP.

ys The transverse center of gravity of the load, off the centerline. Port is positive; starboard negative.

zs The vertical center of gravity of the load, above the ship's baseline.

These <spike>s must not include any portion of the ship's lightweight.

The <load> element might seem like a lot of data. But if the ship survives, it is usually easy to collect. All tankers and bulk carriers carefully document their loading patterns. Collection of this data simply requires a copy of the most recent loading pattern.

2.5.4 Ship Condition

The purpose of the <condition> element is to record whatever information we have on the ship's condition. The <condition> element may contain any of the following fields:

survey_date Date of last Class survey in YYYYMMDD form.

TODO rewrite, need survey type, probably go to a survey element so we can have survey history

wastage Sometimes all we have is overall percent wastage by structural type. This sort of data can be recorded in a <wastage> sub-element. The possible fields are `deck_plate`, `side_plate`, `bott_plate`, `lbhd_plate`, `tbhd_plate`, `deck_stiff`, `side_stiff`, `bott_stiff`, `lbhd_stiff`, `tbhd_stiff`. In each case, the value is the average percent wastage.

note Free form description of the steel condition.

In addition, the <condition> element may have a <tank> sub-element for each compartment for which we have compartment specific condtion inforamtion. The fields for this element ate

code The compartment name coded in the same manner as for the <tank> sub-element in hull damage events.

coat_extent The extent of any coating codes as follow:

N	None
F	Full
B	Bottom only
U	Underdeck Only
b	Bottom and underdeck only

coat_type The type of coating coded as follows:

E	Epoxy
Z	Zinc Silicate

coat_breakdown Percentage area breakdown codes as follows..

E	No visible breakdown
G	< 1%
F	1% to 3% (equivalent to IACS 87 Good)
P	3% to 20% (equivalent to IACS 87 Fair)
H	> 20% (equivalent to IACS Poor)

IACS 87 talks about 3 metrics, pct area of breakdown, pct area of hard scale ????, NIL/¡10/¿10 and pct of edge/weld breadown, ¡20/20-50/¿50 ?????

anodes Condition of anodes if any.

N	no anodes
Y	tank has anodes, no info on condition
E	Lots of calcareous deposits everywhere
G	Evidence of calcareous deposits everywhere
F	Portions of tank under-protected
P	Only scatted evidence of calcareous deposits
H	Gone. Nil evidence of calcareous deposits

The condition fields are still very much under development. Among the machine readable fields which CTX is considering using are, max wastage, hull moment of inertia as percent of as built, deck/bottom section modulus as percent of as built. If you have information on these or other numbers, make sure you get those into the freeform descriptions.

2.5.5 Insurance

As Samuel Plimsoll recognized, insurance has the perverse effect of allowing owners to avoid the results of imprudent purchase and operation decisions, or even benefit from them. In many casualties, the insured value of a ship is well above the market value. In any event, you may use the <insurance> sub-element to record a ship's insured and market values. Here's an example from the BRAER grounding.

```
<insurance hull_value="12.7" loh_value="6.3" mkt_value="4">
</insurance>
```

The <insurance> attributes are:

hull_value Hull and machinery value in millions of US dollars. This should include any "increased value" insurance which is just a tranche of H&M insurance at a lower premium.

loh_value Loss of Hire value in millions of US dollars. Applications will assume zero if not present.

mkt_value CTX's estimate of the ship's market value in millions of US dollars

note Any free form text relating to this element.

2.6 Sample

Just to make things concrete, here is the complete <casualty> element for the EXXON VALDEZ spill.

```
<casualty
  id="19890324_001" date="19890324" Edu="8"  site="Prince William Sound  "
  locale="R"        tod="0009"      acc="B"  lat=" 60.850" long="-146.867"
  coastal="US"      area="np,UW,PWS,     "   weather="B3"         vis="A3"
  note="master not on bridge, nav error, stranded leaving Valdez         ">
  <event ec="Vu" sid="1" date="19890323" tod="2324"
         note="VLCC, 20 man crew, lving Valdez after loading.  Pilot off.">
     <alter tod="2325" course="   " spd="   " helm=" " engine="FA"/>
     <alter tod="2330" course="   " spd="   " helm="P" engine="HA"
            note="turning away from ice, reduced speed"/>
     <alter tod="2339" course="200" spd="   " helm="P" engine="HA"/>
     <alter tod="2352" course="180" spd="12G" helm=" " engine="FA"
            note="Load program up, 55 to 79 RPM over 43 mins"/>
  </event>
  <event ec="GY" sid="1" date="19890323" tod="2353" Cause="N" Sure="5"
         Drugs="M"
         note="Master leaves bridge to tired mate to work around ice">
     <alter tod="2355" course="180" spd="   " helm=" " engine="  "
            lat="-60.123" long="-146.123"
            note="Busby Is. Lt abeam, "/>
     <alter tod="2356" course="180" spd="   " helm=" " engine="  "
            note="Mate said he ordered turn, but no course change"/>
  </event>
  <event ec="NA" sid="1" date="19890324" tod="000130" Cause="P" Sure="4"
         Tired="Y"       Crew_small="Y"  GPS="P"    Owner_care="Y"
         note="Starboard turn begins at 0001.5, too late">
     <alter tod="000130" course="180" spd="   " helm="S" engine="  "
         note="Course recorder shows first course change, 0001.5"/>
     <alter tod="0002" course="   " spd="   " helm="s" engine="  "
         note="Almost certain Mate ordered turn too late"/>
     <alter tod="0007" course="247" spd="   " helm="s" engine="  "
         note="But possible helmsman failed to execute order"/>
  </event>
  <event ec="WS" sid="1" tod="0009"  nuc="N"
         lat=" 60.855" long="-146.873" spd="12G"   course="289"
         wind_bf="B3"  wind_dir="   "  vis="A3"
         ait_temp="1"  sw_temp="  "
         cur_spd="N "  cur_dir="   "   wave_ht="N"   wave_dir="  "
         depth="15.0"  chart=""        squat=""
         hop="4.6"     lop="160.0"     str="Y" days="12"
         note="Hits Bligh reef 2 hrs before high tide at 11-12 kts">
       <tide datum="MLLW"   next_hi_tod="0155" next_hi="+3.81"
             at_hit="+2.99" next_lo_tod="0811" next_lo="+0.0"/>
  </event>
```

```
<event ec="HL" sid="1" vol="41000000" mat="C" hbl="Y" DS="Y" DB="M"
       steel_wt="3500" ref="ntsb91, fig 8, height numbers look optimistic"
       note="160m long damage, 8 of 11 cargo tanks holed, 41000m3 spill">
    <tank code="1C" size="" perim=""
          note="ntsb way low in this tank, damage extended past cl">
        <hi xs="236.1" ys="-12.1" zs="1.0"/>
        <lo xs="274.5" ys="0.0"   zs="0.0"/>
    </tank>
    <tank code="2C" size="" perim="">
        <hi xs="186.5" ys="-12.1" zs="1.0"/>
        <lo xs="236.1" ys="0.0"   zs="0.0"/>
    </tank>
    <tank code="3C" size="" perim="">
        <hi xs="132.4" ys="-12.1" zs="3.0"/>
        <lo xs="186.5" ys="0.0"   zs="0.0"/>
    </tank>
    <tank code="1S" size="" perim="">
        <hi xs="236.1" ys="-24.5" zs="3.3"/>
        <lo xs="274.5" ys="-12.1" zs="0.0"/>
    </tank>
    <tank code="2S" size="" perim="">
        <hi xs="186.5" ys="-25.3" zs="4.6"/>
        <lo xs="236.1" ys="-12.1" zs="0.0"/>
    </tank>
    <tank code="3S" size="" perim="">
        <hi xs="132.4" ys="-25.3" zs="3.0"/>
        <lo xs="186.5" ys="-12.1" zs="0.0"/>
    </tank>
    <tank code="4C" size="" perim="">
        <hi xs="103.0" ys="-12.1" zs="1.0"/>
        <lo xs="132.4" ys="0.0"   zs="0.0"/>
    </tank>
    <tank code="4S" size="" perim="">
        <hi xs="103.0" ys="-15.0" zs="1.0"/>
        <lo xs="132.4" ys="-12.1" zs="0.0"/>
    </tank>
    <tank code="5C" size="" perim="">
        <hi xs="69.2"  ys="-12.1" zs="0.2"/>
        <lo xs="103.0" ys="0.0"   zs="0.0"/>
    </tank>
    <tank code="5S" size="" perim="">
        <hi xs="69.2"  ys="-25.3" zs="1.2"/>
        <lo xs="103.0" ys="-12.1" zs="0.0"/>
    </tank>
    <tank code="FP" size="" perim=""
          note="jack guess, nstb said nothing about FP, but it was a mess">
        <hi xs="274.6" ys="-15.0" zs="3.0"/>
        <lo xs="292.0" ys="0.0"   zs="0.0"/>
    </tank>
```

```
  </event>
  <event ec="DL" sid="1"  date="19890325" tod="0736"
         note="lightering to Exxon Baton Rouge begins"/>
  <event ec="DL" sid="1"  date="19890330" tod="1518"
         note="lightering to Exxon San Franciso begins"/>
  <event ec="DL" sid="1"  date="19890402" tod="2200"
         note="lightering to Exxon Baytown begins"/>
  <event ec="DR" sid="1"  date="19890405" tod="    "
         note=""/>
  <event ec="Sr" sid="1"
         note="3500 tons of steel, renamed S/R Mediterranean"/>
  <ship sid="1"  imo="8414520"  class="AB"  name="exxon valdez"
        st="TC"  dwt="214853"   yob="1986"  flag="US" tnks="11"
        ht="SM"  grt=" 94999"   status="L"  cgo="PC"  sTS="N"
        ns="1"   crew="20"      ig="Y"      pob="N">
     <load ref="djw1"
           draft_fp="17.07"  draft_ap="17.07" heel="0.0"
           cargo_wt="174391" ballast_wt="0"   fuel_wt="1336"
           note="djw1 guess to give 56 ft draft, even keel, min fuel to LA">
        <tank code="1C"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="2C"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="3C"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="4C"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="5C"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="1P"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="1S"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="3P"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="3S"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="5P"     pct="83.24" sg="0.897" temp="35"/>
        <tank code="SLOP_P" pct="64.97" sg="0.897" temp="35"/>
        <tank code="SLOP_S" pct="64.97" sg="0.897" temp="35"/>
        <tank code="FO_P_A" pct="15.00" sg="0.980"/>
        <tank code="FO_P_F" pct="15.00" sg="0.980"/>
        <tank code="FO_P_A" pct="15.00" sg="0.980"/>
        <tank code="FO_P_F" pct="15.00" sg="0.980"/>
     </load>
     <condition
        note="steel in excellent condition, like new.
              but stiffener to bottom welds useless">
     </condition>
  </ship>
</casualty>
```

This is about as detailed a casualty description as you are likely to see. Most of the casualties in the database include far less data. But when we have reasonably complete casualty data such as this, combined with detailed data on the ship, all sorts of analyses are possible, including flooding, spillage, and residual strength.

Use cathay seatrade as an example

For our purposes, white space between elements and attributes is not significant. The order of the attributes within any element tag is insignificant. Therefore, using white space, we can arrange the elements as we wish as long as we maintain the structure of the document (the element nesting). The above arrangement including the indenting was chosen to facilitate human browsing and make this manual easy to print. In fact, with very few exceptions, such as the event sequence, even the order of the elements ***at the same nesting level*** has no significance. For example, the <tank> elements can be in any order you desire.[19]

It is interesting that even with this level of detail, we have no data on the area of the holes in the Exxon Valdez hull. And none is needed. In this case, as it is in most major strandings, the damage is so extensive that the tanks will reach equilibrium within one tidal cycle. Since nothing was done in that time, the area of the openings is irrelevent. Even if Hazelwood and his crew had jammed the P/V valves shut on the damaged tanks at high tide, which would have prevented up to two-thirds of the spillage by pulling a vacuum in top of the tanks (see Devanney, CTX Mate a Combined Loading Instrument and Spill Reduction Package), the area would still not have been important. When the damage is extensive and low, the only thing that is important from a spillage point of view is the highest point of damage in each tank.

The area becomes increasingly important the smaller the damage. But even then you still need the location of the damage in order to do any flooding/spillage calculations at all.

Finally, this casualty demonstrates some of the problems in assigning Primary cause. It is easy to argue that the Captain's error was more egregious than the Mate's mistake. And in fact CTX could easily have gone the other way. The basic point here is that intelligent applications should not make much of the difference between Primary and Necessary, but treat all such causes on pretty much the same basis.

[19] Even the order of the <casualty> elements is insignificant. CTX normally keeps the casualties sorted by casualty ID (basically date). But this is just a convenience to make it easy to avoid duplicating casualties.

2.7 ctx_coresort.xml

The ctx_core.xml file actually exists in two versions:

- The standard ctx_core.xml which we have been describing so far in this manual.
- and ctx_coresort.xml

ctx_coresort.xml is exactly the same as ctx_core.xml; but for the convenience of processing programs several fields have been programatically added to each casualty's common block. They are:

dead Normally the total number of known dead in the casualty, exclusive of attackers; that is, the sum of the known crew, passenger, and bystander dead in each event in the casualty. If it is possible to unambiguously maintain the -3, -2, -1 codes — often the case for single ship casualties – then the negative codes are retained, allowing a more complete sort.

hurt Normally the total number of known injured in the casualty, exclusive of attackers; that is, the sum of the known crew, passenger, and bystander dead in each event in the casualty. If it is possible to unambiguously maintain the -3, -2, -1 codes — often the case for single ship casualties – then the negative codes are retained, allowing a more complete sort.

vol Normally the total number of known spillage in the casualty, that is, the sum of the known spillage in each event in the casualty. If it is possible to unambiguously maintain the -3, -2, -1 codes — often the case for single ship casualties – then the negative codes are retained, allowing a more complete sort.

mat The predominant material spilled coded as in the event material field. In casualties involving spills of more than one material, it is the code of the material with the largest volume spilled.

The primary purpose of these redundant fields is to avoid re-calculating these numbers (the mis-named sort fields) over and over in the web interface. But other processing software can take advantage of these pre-calculated numbers. However, if you are modifying the data base, you should edit ctx_core.xml; and then apply the addsort.pl script to the modified file to create the new ctx_coresort.xml. This script also creates the augmented core file in zip format for the convenience of downloaders.

Chapter 3

The Precis Files

3.1 The <precis> Element

Each casualty in the Core table must have a Precis file containing text descriptions of the event and/or links to such descriptions elsewhere on the Internet. Each Precis file is an XML fragment whose name is the casualty ID, *yyyymmdd_nnn*. An example is shown at the end of this section. These files are all in the directory `precis`.[1] The connection between each record in the Core Table, and its Precis file is by this file name.

Unlike `ctx_core.xml` the Precis files may and usually do contain mixed content. The Precis files are at best semi-structured. They are not really designed for machine based search and extraction. That is the job of `ctx_core.xml`. However, by following the conventions below, it is possible to do some pretty interesting full text searches.

The outer element of each Precis file is <precis>. This element has three attributes:

id This attribute is the Precis file's name; for example, `19890324_001` for the Precis file for the Exxon Valdez spill. This field allows the Precis file to be included in other documents without losing its identity. Required.

ship A comma separated list of normalized ship names involved in the casualty. See Section 2.5 for the normalization rules. This field allows a group of Precis files to be searched by ship name.

do you want a last modified field, author?

imo A comma separated list of IMO numbers of the ships involved in the casualty. This field allows a group of Precis files to be searched by IMO number.

[1] The location of `precis` is up to each site. At CTX sites, it is `/ctx/job/cdb/`.

not_in_core Sometimes CTX keeps data on a casualty in a Precis file, but that casualty is not represented by a <casualty> element in `ctx_core.xml`. There are a number of possible reasons for this. Usually, these are interesting or important casualties which do not qualify for `ctx_core.xml` under the current database rules, but might in the future. The coding of this attribute is:

C	Non-petroleum, non-ethanol cargo (eg palm oil)
D	Being demolished or on way to demolition
N	Not a vessel spill (terminal, pipeline)
P	In process, still gathering data on casualty
R	In repair yard
S	Ship too small
T	Other type of ship (non-tanker, non-bulker)
W	War casualty.
Y	Any other reason not in core

This attribute is required if the casualty is not in `ctx_core.xml` If the casualty is in `ctx_core.xml`, it must ***not*** be present. This field allows programming which checks that every Precis file has a <casualty> element and vice versa to ignore those Precis files which have intentionally been left out.

3.2 The <entry> Element

Each <precis> element is made up of a number of <entry> elements, one <entry> for each source cited or abstracted. <entry> elements have two required attributes:

source The name of the source can either be given directly or by reference to the sources table, `/ctx/job/cdb/sources.xml`. If the value of the `source` attribute is all caps, eg `ACOPS2000`, then this a key to a <reference> in the sources file. See Chapter 5. it is a standalone source name. There are two special values of the `source` attribute.

CTX `CTX` means this <entry> contains CTX's own description of the casualty, and/or CTX opinion, speculation, and critique of other sources.

LINK `LINK` means refer to the URL in the `link` attribute for this <entry>.

type The type of <entry> is coded as follows.

C A direct copy from the source, nothing left out.
A An abridgement; CTX extracted from the source document, but did not change the wording
D Digest, CTX made its own summary of the source document
L Nothing here, must go to the source using `link`. See below.

In addition, <entry> elements may have the following optional attributes:

volume Spill volume according to the entry's source. May be Y meaning yes there was a spill but no volume given in source; N source explicitly says no spill, or a number followed by MMG, KG, G, T, KL, L, B, LT (million gallons, thousand gallons, gallons, metric tons, kiloliters (m3), liters, barrels, long tons). In the event, that a source has multiple volume estimates, use a comma separated list. Every volume number must be followed by a units indicator.

dead Number killed or missing per source. This also can be a comma separated list. Do not include attackers.

hurt Number seriously injured per source. This also can be a comma separated list.

mat Spill liquid. This is free form, using the source's description of the material.

site Location of spill. This is free form, using the source's description of the location.

tod Time of day when casualty occured, ccording to source Use hhmm local if source provides a numerical time of casualty. Otherwise freeform starting with an alphabetic character.

weather Source's description of the weather, if available.

vis Source's description of the visibility, if available.

pos Source's latitude and longitude of casualty if available. Format is nn.mmA,nn.mmB where nn is degrees, mm is minutes, A is either N or S, B is either E or W. If you have seconds, use nn.mm.ssA,nn.mm.ssB. Degrees can be 1, 2, or 3 digits, but minutes and seconds are always two digits, with a leading zero if needed.

link URL of the source document if one exists. In the web site interface to the database, this attribute is coded as a hyperlink. Clicking on this

field, should take you to the entry's source. The URL must include the protocol, that is start with `http://` or equivalent.

To allow some control over formatting, the text content of each <entry> element may contain XHTML tags which should be passed thru to the final HTML document. This also allows the text to contain links to other documents or images via <a> or <img> tags. See Chapter 4 for the handling of CTX's own collection of casualty photos and drawings. Use <cite> tags for documents which do not have URLs. Please note: the XML rules must be followed, as well as HTML. Elements must all be properly nested, and the reserved XML characters escaped.

Why not xlink? Autogenate part of src attribute?

In addition, to all the normal XHTML tags, an <entry> may contain one or more <ctx_comment> elements which is CTX commentary on the source text, which should be passed thru to the final displayed document. In general, try to avoid interpolating CTX commentary into a non-CTX <entry> element. It is almost always much better to use the `CTX` <entry> to critique a source. Use a <ctx_comment> only when the source is clearly misleading or needs explaining.

Finally an <entry> may contain one or more <ctx_image> elements referencing a photo or drawing in CTX's own collection of casualty pictures. This element has only one attribute `name` which is file name of the picture in the casualty's photo folder. See Chapter 4. The <ctx_image> `name` should contain only the file name, not the path to the file. This allows the location of these pictures to be site dependent. In fact, site software could direct this link to some other site, avoiding the need to store the picture files at every site.

```
<precis key="19721219_001" ship="Sea Star, Horta Barbossa">
<entry source="SIS83" type="D" volume="" dead="12" link="">
<p>
Sea Star (hitee) collided with Horta Barbosa (hitter).  Barbosa was in ballast,
good weather, fire, sank, lat 25.4N, long 58.12E, sank on 24th, 12 killed
</p>
</entry>
<entry source="CAHILL_G" type="D" volume=""  dead="11" link="">
<p>
On the night of December 19, 1972, the Sea Star and the Horta Barbossa were proceeding
on nearly complementary courses in the Gulf of Oman.  The visibility was excellent
with a light NE wind.  The Sea Star was loaded on a course of 142 at about 16 knots.
The Horta Barbossa was in ballast on a course of 322 also at about 16 knots.
</p><p>
The Sea Star detected the Barbossa at 14 miles and altered slightly to stbd to 145.
The Barbossa detected the Sea Star at 16 miles, determined that she would easily pass stbd to stbd.
Indeed their courses were sufficiently displaced so that, if both had maintained course and speed,
they would have passed starboard to starboard with a Closest Point of Approach of about one mile.
The Sea Star apparently regarded this separation as insufficient and,
at a range of about four miles, went starboard to effect a port to port passing.
The Horta Barbossa maintained course and struck the Sea Star almost
at right angles just forward of the bridge.  She exploded and sank, killing 11 of her crew.
</p>
</entry>
<entry source="HOOKE" type="A" volume="" dead="12" link="">
<p>
The 120,300 dwt South Korean motor tanker Sea Star was on a voyage
from Ras Tanura to Rio de Janeiro loaded with crude oil, when she was in a collision
with the Brazilian motor tanker Horta Barbossa in the Gulf of Oman
in approximately 25.18N, long 57.334E just before dawn on December 19, 1972.
A massive fire broke out on the Sea Star on which 12 crew members died.
</p><p>
... the Sea Star, still blazing furiously, drifted, listing heavily to port.
After several huge explosions, oil spillage was noted through the 40 ft collision hole
in her side, leaving the sea on fire and the tanker blazing end to end.
She continued to drift, partially submerged, until finally sinking after another massive explosion ...
</p>
</entry>
<entry source="CTX" type="D" volume="115000T" mat="C" dead="12" link="">
<p>
CEDRE lists this spill at 115,000 tons.
<p></p>
This is a classic case of the dance of death.
Two ships on complementary courses, displaced to starboard.
One ship opts for port to port, the other ship decides to pass starboard to starboard.
Neither ship talks to the other, which is the only solution in this situation.
</p>
</entry>
</precis>
```

Chapter 4

Photos and Drawings

4.1 The *pics* directory

The photos, plans, charts and drawings that CTX has collected relating to the CDB casualties are filed in a `pics` directory which must be in the same directory as the `precis` directory. Each casualty for which we have such pictures is represented by a sub-directory in `pics`. This sub-directory's name is the casualty ID, *yyyymmdd_nnn*. It is called the casualty's photo folder.

Each such folder will contain a number of image files and a `pics.xml` file. The image files can be named anyway you like but the extensions (`.jpg`, `.svg`, etc) must match the file's format.

The `pics.xml` file contains catalogue data on the images in this folder. Here's a sample

```
<pics id="20021113_001">
  <pic name="prestige_sinking.jpg"
       title="Helo view of Prestige sinking"
       ships="prestige" imos="7372141"
       date=""
       image_type="photo"
       ack="ABS 2003 report"
       source="Technical Analyses Related to the Prestige Casualty on 13 Nov 2002"
       link=""
       size=""
       note=""/>
  </pic>

.... more pic elements ....

</pics>
```

The outer <pics> has only one attribute, the casualty ID. <pics> must contain one <pic> element for each image file in the folder. The attributes of this element are:

name The name of the image file to which this <pic> refers.

title This field will be displayed in photo lists, etc. Should give the user a pretty good idea of what the image shows. Free form but anything more than 40 characters may be truncated.

ships Comma separated list of normalized ship names in this image. This field allows our images to be searched by ship name.

imos Comma separated list of IMO numbers of the ships in this image. This field allows our images to be searched by IMO number.

date Date photo was taken in *yyyymmdd* format. For drawings, use the last revision date if known.

image_type One of `photo`, `chart`, or `plan`. Use `plan` for all drawings other than geographical sketches.

ack Short string describing source. Anything more than 30 characters may be truncated. This will be displayed where ever an acknowledgement is appropriate. Use `ANON` only for sources whose confidentiality CTX has guaranteed.

source Description of source, ideally complete enough to allow user to locate the source.

link Link to `source` if available. Include protocol, eg `http:....`

size The size of the image file in kilobytes.

note Freeform commentary on the image. But this field should not attempt to draw lessons from the image or make inferences about cause. Keep this comment simply descriptive.

The <pic> element may contain text content which may be anything you want to say about this image. This text can contain background information on the image, the photographer, camera, film, etc; but should not describe the casualty. That is the job of the precis file.

The order of the <pic> elements should be the order in which the images should be displayed in a slide show, But there is no guarantee that a particular application will respect this order.

Chapter 5

Documenting sources

5.1 The `sources.xml` File

The Precis files are supported by `sources.xml` which is a bibliography for the database. The root element of `sources.xml` is <references>. This element has two required attributes:

update The GMT time the file was last modified in ISO8601 (short) format.

author The user ID of the person responsible for the last modification.

The <references> element contains a number of <reference> elements, one for each source catalogued. Here are two sample <reference> elements.

```
<reference
    key="DOLPHIN"
    author= "Dolphin Maritime"
    title=""
    date=""
    link ="dolphin-maritime.com/9.html">
<p>
Up to date and fairly comprehensive list of all vessel casualties,
which involve the vessel signing a Lloyds Open Form or equivalent.
Good place to find lots of engine failures.
This is the most comprehensive site I have found so far,
but tries to get report out very quickly,
rarely has an details, cause info.
nicely organized into one page per casualty.
</p>
</reference>
<reference
    key="ACOPS2003"
```

```
    author= "Advisory Committee on Protection of the Sea"
    title= "Annual Survey of Reported Discharges Attributed to
            Vessels and Offshore Oil and Gas Installations
             Operating in the United Kingdom Pollution Control Zone, 2003"
    date ="200403"
    link ="www.mcga.gov.uk/c4mca/acops_final_report_2003.pdf">
<p>
ACOPS annually produces a detailed report on discharges into UK waters,
It will typically have several hundred spills, mostly quite small.
In a normal year, only a handful of these spills will be from tankers.
There is often some info on initial cause.
The data is not in a format that is easily machine readable.
Nor does it include the ship's IMO number.
</p>
</reference>
```

Each <reference> element contains the following attributes.

key The code by which the source is known in the Precis files. This field is required and must be unique. It should be in all caps, and is usually an abbreviation of a author or entity, combined with a year if required.

author The entity responsible for generating the report. This need not be a person. Often it is an organization. Required.

title Source title. Not required.

date Publication date. May be a year (YYYY), a year and a month (YYYYMM), of a full date (YYYYMMDD). For annual casualty reports this will normally not be the year covered. Not required.

link URL of the source if available. Not required. Currently, only the `link` in the <entry> element is actually used. bad design

The content of each <reference> is a free form description of the source or document. To control formating, this description may contain XHTML tags.

Chapter 6

The Docs Files

The Precis files contains exerpts and links to various source documents. These brief entries are designed to be displayed whenever a user clicks on the Sources tabs for a casualty.

The Docs directory on the other hand is a library of material collected by the CTX filed by casualty. There should be one sub-directory in the `docs` directory for each casualty. The name of that directory is the casualty `id`. This material can and should be referenced by the Precis files with a link of the form `http://www.c4tx.org/ctx/job/cdb/docs/`*yyyymmdd_nnn*`/`*xxxxxx* where *yyyymmdd_nnn* is the casualty `id` and *xxxxxx* is the file name.

There is really no difference between such a link and an external link to a non-CTX URL, except that there is a much smaller chance that the link will be moved or removed. The intention is that the `docs` directory be a permanent repository for these documents.

Often documents in the `docs` directories have information that is relevent to more than one casualty. This is handled by hard links, that is: giving the same document more than one name. However, summary reports and studies covering a large number of casualties are not filed in the `docs` directories.

Unlike the Precis and Pics files, there are is no formatting requirements on the Docs files. While the CTX attempts to convert source documents to a format that is displayable by a browser, this need not be the case. Of course, when this is not the case, the document will be of limited use.

The `docs` directory is not really part of the CTX database proper. However, the CTX follows a policy of not filing any material that cannot be fully disclosed. So the documents in the `docs` directories may be downloaded on an individual basis.

Chapter 7

The Ship Files

7.1 Permanent Ship Data

`ctx_core.xml` and the Precis files record very little permanent ship data other than the IMO number. Ideally, `ctx_core.xml` would contain no permanent ship data, referring to a ship data base for all such data. Unfortunately, no reasonably complete, non-proprietary ship data base yet exists. Until such a data base is available, we will include a few overall ship parameters in the casualty data to allow simplistic analyses by ship type, size and age. The idea is that the IMO number will be used to access a vessel database containing the relevent ship data. The CTX casualty data can be used in conjunction with any ship data base that includes the ship's IMO numbers, which means practically all of them. In particular, major ship particulars, build yard, and original owner can almost always be obtained from the Miramar Ship Index, by entering the ship's IMO number in the Miramar ID field.

However, in order to do more detailed casualty analysis, we often need ship data that is not recorded in any of the standard data bases. In particular, we need non-standard information on hull form, compartmentation, machinery redundancy, and maneuverability.[1] Hopefully this will change. But until this happens CTX must maintain its own tanker and bulk carrier data base.

Each tanker and bulk carrier for which CTX has ship data has its own

[1] Responders should be aware of this and gather as much data on the ship as they can in investigating a casualty. At a minimum, we need the hull offsets, the tank capacity plan, and the lightweight curve. If possible, obtain a copy of the General Arrangement, the Midship Section, and the Trim and Stability booklet.

directory in the CTX tanker data base (TDB). The name of that directory is the ship's IMO number. This *ship folder* contains all the information that CTX has on this tanker in a set of XML files, known as the Ship Data Files or SDF. Fully filled out these files represent an immense amount of information on the ship. See The CTX Ship Data Files Manual for a description of these files. This information is designed to be the single source for any application that needs data on this ship. This means that the SDF could have far more data than normally needed in casualty analysis. For obvious example, the ship's speed-fuel curve is required for all sorts of purposes, including chartering calculations. But it is almost never needed in casualty analysis. Of course in many cases we have little or no info on a particular ship, in which case most if not all of the SDF files for that ship will be missing.

Unfortunately, at this point in time, CTX is not able to make all the SDF files downloadable since some of them still contain proprietary data. You can access individual ship data we have for any ship in the TDB via the web interface. And we may be able to send you the SDF files on a particular ship. Send an email to tdb@c4tx.org to find out.

7.2 Ship History

The CTX CDB does not record ownership nor management at the time of casualty. Ship owners are masters at concealing their identity Nor do we attempt to identify the ship management. Tracking down this data for all our casualties is a task far beyond the capability of the CTX. Fortunately, for recent casualties, — about 1999 on — this information is often available in the Equasis database, www.equasis.org. The Equasis database is keyed by IMO number. Once you have a ship's IMO number, we suggest you search Equasis on this number. In many cases, Equasis has a fairly complete history of the ship.

Unfortunately, currently there is no way of automating the joining of the CTX and Equasis data.

Appendix A

Download

A.1 Core file only

To download only the current version of the core file in zip format, simply point your browser at www.c4tx.org/ctx/job/cdb/prod/ctx_coresort.zip. After unzipping, you will have `ctx_coresort.xml`. This file is guaranteed to be the same file as used by the web-interface at the time of the download. It is updated every time `ctx_coresort.xml` is uploaded to the web-site.

A.2 Full Database

The core database can be downloaded as a compressed tar file from
www.c4tx.org/ctx/job/cdb/prod/ctx_cdb.tar.gz.

CTX uses a simple versioning system. The standard location for the Casualty Database is `/ctx/job/cdb`. In this directory, there is

1. a sub-directory for the precis files called `precis`,
2. a sub-directory for the image files called `pics`,
3. and a subdirectory for each version called $m.n$ where $m.n$ is the version number.

It is the version sub-directory that is included in the download file.[1] This directory contains both the core database for the version and CTX's code for accessing that version. Until CTX resolves possible legal problems, it is

[1] CTX follows the practice of always putting a symlink called `prod` in `ctx/job/cdb` so that the current production version of the database can always be referred to as `ctx/job/cdb/prod`. Simlarly, `ctx/job/cdb/search.html` redirects to `ctx/job/cdb/prod/search.html`. If there is a development version on the web site, it will be linked to `ctx/job/cdb/dev`.

not possible to download the precis files or the CTX ship database directly. These must be accessed via the web interface.

When you untar the download file, you will be presented with a number of sub-folders.[2] The casualty data is in the `xml` subdir. The core data file is `xml/ctx_core.xml`.[3] The `xml` subdir also contains a sub-directory called `labels`. The `labels` directory contains a number of files which provide explanatory descriptions or labels for each of the coded fields in the database. See Section D for how to incorporate these labels in your analyses.

The other version sub-dirs are:

scripts A number of scripts for analysing the data base. You may want to use these as templates for your own analyses. See Section D.

MAN The version Manual. The mark-up language is latex, but the directory should include the manual in PDF format.

[2] On Unix systems, this can be done by issuing `tar xvzf ctx_cdb.tar.gz`

[3] Be sure and check the `update` field at the top of the `ctx_core.xml` file to see how old this version is. The CTX does not guarantee that the tarball is as up-to-date as the web-site version. If you need a more up-to-date version, email cdb@c4tx.org.

Appendix B

The `labels` Files

Each CDB distribution includes a `labels` sub-folder. This folder contains a number of simple files which convert the short-hand codes in `ctx_core.xml` to a longer, more explanatory descriptions, called a *label*. There is one such file for each field that allows/requires such a translation. The name of each of these files is *xxxxx*`.xml` where *xxxxx* is the attribute name. For example, here is `labels/Sure.xml`.

```
<labels field="Sure" shortname="Evidence?" longname="Quality of Causal Evidence"
        regex="^[0-5 ]$>
    <label key="0" short="None"           long="No info on cause"
           note="Use this with primary cause Unknown code"/>
    <label key="1" short="Very weak"
           long="Only situational evidence for cause, no scenario dominant"/>
    <label key="2" short="Situational"
           long="Only situational evidence for cause, one scenario dominate"/>
    <label key="3" short="Circumstantial"
           long="Circumstantial or preponderant evidence for cause"/>
    <label key="4" short="Solid"
           long="Solid evidence for cause"
           note="Including creditable eye witness"/>
    <label key=" " short="Beyond doubt"
           long="Unimpeachable evidence for cause."/>
</labels>
```

All the `labels` files have the same simple structure. Theres is a single enclosing XML element <labels> containing a number of <label> elements, one for each legal value of the field. The <labels> element has three required attributes:

field the name of the field in the database,

shortname a name for the field which can be used as a label in displaying the contents of the field.

longname a longer name for the field which can also be used as a label in displaying the contents of the field, or as an explanation of the field's meaning.

The <labels> may also have a `regex` attribute which is a regular expression which values of the field must match. Input checking programs can use this regular expression to confirm that the coded field value is legal.

This attribute may become mandatory.

Each <label> element has three required attributes:

key the value coded

short a short description for that code.

long a longer description for that code.

The `short` description is designed to be used in situations where the field being displayed is separately identified with a name such as the `shortname` in the <labels> element. The `short` label is restricted in length. It is illegal to have a `short` label that is longer than the longest, already existing `short` label. Typically, the `short` label, together with a name, will be inserted into a table.

The `long` description is designed to be used by itself, so it must be self-describing. There is no restriction on length. Typically, the `long` label will be displayed on its own line in a list. It can also be used as a tool tip.

Each <label> element may also have a `note` field which may contain additional information or coding instructions. This field is aimed at the data entry coder. It is not intended to be normally displayed.

The <label> elements for the event codes in `labels/ec.xml`, must have an additional attribute called `causal`. This field must be either `Y` or `N`. If `causal` is `Y`, the event may be causal. Otherwise, it can only be a pure consequence. This field is used by CDB data checking programs, which make sure that the data coding follows the rules of this manual.

Using labels files such as that shown above, it is easy for a program to convert the cryptic `enc="h"` to the more informative:

```
Encounter type:  Probable head on or near head on
```

You do not need the `labels` files to access the core file. But by properly employing them, you can write analysis programs or scripts which don't have to be changed, even if the CTX changes the database coding. See Section

D for one way to do this. If CTX changes the database coding, it will also change the `labels` files appropriately.

It is possible to have a `labels` file without any <label> sub-elements. For example, here is `labels/imo.xml`

```
<labels field="imo" shortname="IMO Number" longname="Ship IMO Number"
        regex="^ {7}$|^\d{7}$" >
</labels>
```

Such files can be used both to identify the field and provide a regular expression for input checking. In this case, the regular expression says the field must be either exactly seven blanks or exactly seven digits.

I think there should be two different kinds of files:

1. Regular labels files for fields which can be enumerated.
2. Regex files for fields wich cannot.

There probably should be two directories.

Appendix C

Stylesheets (to be written)

There are two basic approaches to analysing XML databases.

stylesheets In this approach, special purpose tools designed just for XML are used. The two most important are XSLT, a language for transforming XML to some other form, and XQuery the XML counterpart to SQL.

scripting In this approach, a general purpose programming language is used. The XML database is converted to a data structure easily understood by the language, at which point the full power of the language is available. The desired analysis is carried out via normal programming techniques. See Section D for some examples.

In this section, we demonstrate how to use the first approach to convert portions of the CDB to HTML for display in a browser.

(To be written.)

Appendix D

Scripting the CDB

In this section, we demonstrate how to use the scripting approach to analyse the CTX CDB using the Perl language. All these scripts are actual working code pulled in from this version's `scripts/sample` folder. If you have downloaded the database and have Perl installed, you should be able to go to `scripts/sample` and run these scripts.

We start out with a very simple script, Figure D.1, which sorts the casualties by coastal state (`coastal`) printing out the casualty date, ships involved, and deaths by ship.

This job takes about 20 lines of code. The most important line is the one that starts with `$cdb = XMLin`. This line does all the work of parsing the core file and generating an array of casualties. It's a lot of work and `XMLin` is slow, so be prepared to wait for 5 or more seconds.

The second important line starts with `foreach $cas`. This line loops thru the array of casualties. Within this loop, any of the casualty's common (top level) fields can be referenced by `$cas->{`*xxxx*`}` where *xxxx* is the desired attribute. In this case, we throw in a sort on the `coastal` field, and the job is just about done.

We want to show the number killed on each ship. But in the CTX CDB dead are assigned to events, not ships. The first inner loop runs through the casualty's <event>s, and puts the event dead into a hash keyed by the event's ship ID.

The second inner loop runs thru the casualty's <ship>'s and prints out the desired information.[1] Within that loop `$ship->{`*yyyy*`}` references the

[1] This code only works if the casualty's <event> and <ship> elements are in an array, even if there is only one of them, the normal situation for ships. The `ForceArray` parameter in the call to `XMLin` assures that this is the case.

```
#!/usr/bin/perl
#sort_coastal.pl
#    print out casualties sorted by coastal state
#
use XML::Simple;

$cdb_file = "../../xml/ctx_core.xml";     # default location of core file
$cdb_file = shift @ARGV if @ARGV;         # allow user to set core file

# parse database putting casualties into big array

$cdb = XMLin($cdb_file,
             NormalizeSpace => 2,        # always trim whitespace
             ForceArray => [event, ship],# put all event and ship elements in arrays
             KeyAttr => [],              # never fold arrays into hash
             SuppressEmpty => "");       # ignore empty elements

# loop thru array of casualties sorting by coastal state

$total_known_dead = 0;
$number_cas = 0;
foreach $cas (sort {$a->{coastal} cmp $b->{coastal}} (@{$cdb->{casualty}})) {
    $number_cas++;

# for this casualty, put each event's known dead into hash keyed by ship id

    %ship_dead = {};
    foreach $event (@{$cas->{event}}) {
        $ship_dead{$event->{sid}} += $event->{dead} if ($event->{dead} > 0);
    }

# loop through this casualty's ships, printing one line per ship

    foreach $ship (@{$cas->{ship}}) {
        printf(STDOUT "coastal=%2s date=%8s", $cas->{coastal}, $cas->{date});
        printf(STDOUT "  ship=%-22s imo=%7s known_dead=%5.0f\n",
            substr($ship->{name}, 0, 22), $ship->{imo}, $ship_dead{$ship->{sid}});
        $total_known_dead += $ship_dead{$ship->{sid}};
    }
}
print(STDOUT "\nCasualties=$number_cas Total Known Dead=$total_known_dead\n");
```

Figure D.1: Sort by Coastal State

ship element's fields. We grab what we want and are done. Just treat the `$cdb` line and the last part of `foreach` lines as inscrutable boilerplate.

Notice how all the database names are already supplied to us. Notice also the database can change pretty drastically without affecting this code. As long as the basic structure — one or more <event> elements and one or more <ship> elements inside a bunch of <casualty> elements — doesn't change, and the names of the variables we are interested in doesn't change, changes elsewhere to the database (new elements, new fields, etc) can occur without affecting this script.

Basically, all Perl scripting analyses go the same route. Use XMLin to put the casualties into an array, and then loop through that array, doing whatever you need to do. ***Almost all higher level languages have similar tools for parsing the XML and putting the data into a structure, that is easily accessible by that language.***

Figure D.2 is a slightly more interesting script that sorts the casualties by known dead. This codes uses two passes through the casualty array.[2] In the first pass, we compute the number of known dead for each casualty, and attach that number to the casualty node in the variable `$cas->{dead}`. In the second pass, we can sort on this variable, just as it were part of the original database. As always we place no limitations on the number of events nor the number of ships.

Usually the work involves data in the <event> elements. Figure D.3 produces a list of causal hatch cover failures including ship name and number killed by ship.

The first loop converts dead by event to dead by ship for each casualty, and adds those numbers to the casualty's data structure. Then for each casualty, this script loops over the casualty's events picking out the events of interest. Often — as in this case — we need to figure out which ship is involved. This is done by finding the <ship> element whose `sid` matches the event's `sid`.

Notice how we exclude non-causal hatch cover failures. Sometimes a hold will flood rapidly due to say a side shell failure. The increase in pressure inside the hold will blow the hatch cover off. In this case, the hatch cover failure is a consequence, not a cause; and it would be incorrect and misleading to blame the dead, etc on the hatch cover. In the CTX CDB, there is no automatic assumption that an event is causal. In fact, an event

[2] This could be done in one pass by using a custom sort comparison function. But the code would be more complicated, and little would be gained. Almost all the time is spent in the initial read in and parsing of the database.

```
#!/usr/bin/perl
#sort_by_dead.pl
#    print out casualties involving deaths
#    sorted by number of known dead
#    this version uses two passes to keep the code simple
#
use XML::Simple;

$cdb_file = "../../xml/ctx_core.xml"; # default location of core file
$cdb_file = shift @ARGV if @ARGV;       # allow user to set core file
#
# read in and parse the database
#
$cdb = XMLin($cdb_file, NormalizeSpace => 2,
                        ForceArray => [event, ship],
                        KeyAttr => [], SuppressEmpty => "");
#
# figure out total known dead for each casualty
# and put that number in the casualty subtree
#
$known_dead = 0;
$n_cas = 0;
foreach $cas (@{$cdb->{casualty}}) {
    $n_cas++;
    $cas->{dead} = 0;
    foreach $event (@{$cas->{event}}) {
        $cas->{dead} += $event->{dead} if ($event->{dead} > 0);;
    }
    $known_dead += $cas->{dead};
}
#
# sort casualties by known dead and print
#
foreach $cas (sort {$b->{dead} <=> $a->{dead}} (@{$cdb->{casualty}})) {
   last if ($cas->{dead} <= 0);
   printf(STDOUT "date=%8s dead=%4d", $cas->{date}, $cas->{dead});
   foreach $ship (@{$cas->{ship}}) {
    printf(STDOUT "  ship=%-22s", $ship->{name});
}
   printf(STDOUT "\n");
}
print(STDOUT "\nTotal Casualties= $n_cas Total Known Dead=$known_dead\n");
```

Figure D.2: Sort by Known dead

```
#!/usr/bin/perl
# hatch_covers.pl
# list dead by ship associated with causal hatch cover failures
# example of obtaining all dead on ship, not just dead assigned to event
#
use XML::Simple;

$xml_file = "../../xml/ctx_core.xml";
$cdb = XMLin($xml_file, NormalizeSpace => 0,
             ForceArray =>[ship, event],
             KeyAttr => [], SuppressEmpty => "");
#
# calculate casualty dead for each ship and add to tree
# we lose diff between -3, -2 and -1
#
foreach $cas (@{$cdb->{casualty}}) {
    foreach $event (@{$cas->{event}}) {
        next if ($event->{dead} < 0);
        $cas->{dead}->{$event->{sid}} += $event->{dead};
    }
    $known_vol += $cas->{vol};
}

$known_dead = 0;
foreach $cas (@{$cdb->{casualty}}) {          # loop over all casualties
    foreach $event (@{$cas->{event}}) {       # loop over events, this casualty
        if ($event->{ec} =~ /H[hH]/ && $event->{Cause} =~ /[PNS]/) {
           foreach $ship (@{$cas->{ship}}) {  # find this event's ship
               if ($ship->{sid} eq $event->{sid}) {
                   printf(STDOUT "date=%8s ship=%-22s dead=%d\n",
                          $cas->{date}, $ship->{name}, $cas->{dead}->{$ship->{sid}});
                   $known_dead += $cas->{dead}->{$ship->{sid}};
                   $cas->{dead}->{$ship->{sid}} = 0; #avoid dbl count if 2 cvr failures
                }
            }
        }
    }
}
print "Total known dead = $known_dead\n";
```

Figure D.3: List Hatch Cover Casualties

is non-causal unless it has a non-blank `Cause` field.

Notice also the flexibility of this structure. This script does not care how many of a casualty's events qualify nor how many ships are involved. This is a requirement of any valid query of the CTX CDB.

Figure D.4 generates a little table showing the depths of penetration in collisions for which we have depth of penetration.

```
#!/usr/bin/perl
#list_dop.pl
#    list known depths of penetration in collisions
#
XML::Simple;

$xml_file = "../../xml/ctx_core.xml";
$cdb = XMLin($xml_file, NormalizeSpace => 2,
             ForceArray =>[event, ship],
             KeyAttr => [], SuppressEmpty => "");

printf("  DATE  |      SHIP NAME       | DOP |ANGLE |IMPACT|\n");
foreach $cas (@{$cdb->{casualty}}) {
    foreach $event (@{$cas->{event}}) {
        next unless $event->{ec} =~ /^C/; # filter out non-collisions
        if ($event->{dop}) { # print only if non-blank
            foreach $ship (@{$cas->{ship}}) {   # find event's ship
                $shipname = $ship->{name} if $ship->{sid} eq $event->{sid};
            }
            printf("%8s|%-22s|%5s|%6s|%6s|\n",
                $cas->{date}, $shipname, $event->{dop},
                $event->{angle}, $event->{impact});
        }
    }
}
```

Figure D.4: List Collision Depth of Penetration

The pattern is pretty obvious. Simply work your way down the element nesting until you get to the field you are interested in. The `dop` we are interested in is in a <event> element, which is in a <casualty> element. To get ship information for the ship associated with an event, we match the <event>'s `sid` to a <ship> `sid`.[3]

Once again a casualty can have any number of collisions involving any number of ships without affecting this code.

[3] If we wanted to compute relative speed, we could use the <event>'s `cid` to find the other (non `impact="NO"`) ship's speed (if known).

Possibly multiple sub-sub-elements such as <component> in machinery events, <tank> in the hull damage events, and <alter> in bridge events are handled in the same manner. Figure D.5 prints out a list of all the damaged compartments in the CDB.

```
#!/usr/bin/perl
#list_damaged_tanks.pl
#
use XML::Simple;
$xml_file = "../../xml/ctx_core.xml";
$cdb = XMLin($xml_file, NormalizeSpace => 2,
             ForceArray =>[event, tank, ship],
             KeyAttr => [], SuppressEmpty => "");

foreach $cas (@{$cdb->{casualty}}) {
    foreach $event (@{$cas->{event}}) {
        next if ($event->{ec} !~ /^H/);           # filter out non-damage events
        foreach $tank (@{$event->{tank}}) {
            if ($tank->{code}) {                  # dont print blank tanks
                foreach $ship (@{$cas->{ship}}) { # find this event's ship
                    $shipname = $ship->{name} if $ship->{sid} eq $event->{sid};
                }
                printf(STDOUT "date=%8s ship=%-22s tank=%s\n",
                              $cas->{date}, $shipname, $tank->{code});
            }
        }
    }
}
```

Figure D.5: List Damaged Tanks

`XMLin` puts the <tank> elements for each <event> element into an array. So we need an innermost loop to traverse this array and pick out the individual compartments. The tanks will print out in a weird order, so you may want to sort by tank code (see the first script) in the innermost loop.

That's all there is to it. With these few lines, you can access any piece of data in the CDB, then do with it as you wish.

Figure D.6 is a very important example for it demonstrates two basic techniques needed to properly query the CTX CDB. This script lists the killed, hurt, and spill volume associated with each necessary (including primary) cause machinery failure together with the name of the ship that had the failure.

As usual we put the casualties into an array and loop through that array. Within each casualty, we loop through the events until we find an event that

matches the query. When we encounter such an event, we need to find the ship involved which we do by finding the <ship> element whose `sid` matches the <event> element's `sid`.

Now we must figure out the deaths, injuries, and spill volume associated with the causal event. These consequences may be recorded in subsequent events. So we need to look ahead and go through all the events from the causal event on and accumulate each event's death's etc. But in so doing we stop at the next event that matches the query, if there is one. In the CTX CDB it is not double counting to assign a death, etc to more than one cause, as long as the causes are different. But it is double counting to assign the same death to the same cause more than once.[4]

Notice that if there is more than one necessary cause machinery failure in a casualty that casualty will show up more than once in the listing.

There is some arbitariness to this logic. If in a sinking the coder put all the deaths in the sinking event, then all the casualty's deaths will be assigned to the last qualifying necessary cause. But this is far less arbitary than a "logic" that says for example: if a casualty involves a sinking, blame all the deaths on the sinking.

[4] Actually, this is CTX policy, not a requirement of the database. An application which never wanted to double count could choose to stop at the next necessary cause. The CTX's view is that if a death could have been avoided if either A or B hadn't happened, then that death should be counted against both A and B.

```
#!/usr/bin/perl
#list_mach.pl.pl
#    list necessary cause machinery failures
#    with known killed, injured, and spill volume
#
use XML::Simple;

$cdb_file = "../../xml/ctx_core.xml";   # default location of core file
$cdb_file = shift @ARGV if @ARGV;       # allow user to set core file
$cdb = XMLin($cdb_file, NormalizeSpace => 2,
                        ForceArray => [event,ship],
                        KeyAttr => [], SuppressEmpty => "");

printf("  DATE  |      SHIP NAME      |DEAD|HURT|  VOLUME  |EC|\n");
#
# loop over all casualties and over all events in each casualty
#
foreach $cas (@{$cdb->{casualty}}) {

    $n_event =0;
    foreach $event (@{$cas->{event}}) {
        $n_event++;                     # needed to ignore earlier events below
#
# pick out events that qualify, in this case necessary cause, machinery failures
#
        if ($event->{ec} =~ /^M/ && $event->{Cause} =~ /[PN]/) {
#
# find ship which had machinery failure, save name, any other ship stuff desired
#
            foreach $ship (@{$cas->{ship}}) {
                if ($ship->{sid} eq $event->{sid}) {
                    $shipname = $ship->{name};
                    $ship_imo = $ship->{imo};
                }
            }
#
# look ahead to get known killed, injured, spill volume
# associated with this machinery failure
# we stop at next necessary cause machinery failure if there is one
#
            $dead = $hurt = $vol = 0;
            $n_ev_2 = 0;
            foreach $ev_2 (@{$cas->{event}}) {
                $n_ev_2++;
                next if ($n_ev_2 < $n_event); # ignore earlier events
                last if ($n_ev_2 > $n_event &&
                        ($ev_2->{ec} =~ /^M/ && $event->{Cause} =~ /[PN]/));
                $dead += $ev_2->{dead} if $ev_2->{dead} > 0;
                $hurt += $ev_2->{hurt} if $ev_2->{hurt} > 0;
                $vol  += $ev_2->{vol}  if $ev_2->{vol} > 0;
            }
            printf("%s|%-21s|%4d|%4d|%10.0f|%2s|\n",
                    $cas->{date}, $shipname, $dead, $hurt, $vol, $event->{ec});
        }
    }
}
```

Figure D.6: List Necessary Cause Machinery Failures

Figure D.7 is a varient of figure D.6; but instead of merely listing the qualifying casualties, it breaks the machinery failures down by machinery event code, generating a summary table. The first step is to run through the casualties as before, but this time we accumulate the deaths, spill volume in hashes keyed by the causal event code. Otherwise the logic is exactly the same as Figure D.6.

In the second step, we want to create a table with a one line summary for each event code. But the two character machinery event codes are hardly descriptive. What we need are the event code descriptions described in Appendix B. To do this we put the event code descriptions (called *labels*) into a hash keyed by the event code. Since the little file, `labels/ec.xml`, that contains the event code labels is also XML, XMLin can do this automatically for us. Later in the script, we can reference these descriptions via `$labels->{$ec}->{short}` where `$ec` is the event code.[5] Figure D.8 shows sample output of this script.

This script has no idea what the valid event codes are nor their explanatory labels. This means that, if the coding of the `ec` field changes or the descriptions change, this script does not have to be changed. Of course, the `labels` files must match the database coding which is why CTX includes a set of `labels` files in each distribution of the database.

[5] Obviously, this technique is not limited to event codes; but can be used for any labeled field in the database. Equally obviously, if you need the long (self-standing) label, simply replace `short` with `long`.

```
#!/usr/bin/perl
#sum_mach.pl  summarize necessary cause machinery failures by event code
#
use XML::Simple;
$cdb_file = "../../xml/ctx_core.xml";   # default location of core file
$cdb_file = shift @ARGV if @ARGV;       # allow user to set core file
$cdb = XMLin($cdb_file, NormalizeSpace => 2,
                        ForceArray => [event,ship],
                        KeyAttr => [], SuppressEmpty => "");
#
# loop over all casualties and then all events in each casualty
#
foreach $cas (@{$cdb->{casualty}}) {

    $n_event =0;
    foreach $event (@{$cas->{event}}) {
        $n_event++;                     # needed to ignore earlier events below
#
# pick out events that qualify
#
        if ($event->{ec} =~ /^M/ && $event->{Cause} =~ /[PN]/) {
            $ec = $event->{ec};
            $num_cas{$ec}++; # number of casualties this code
#
# look ahead to get known killed and spill volume
# associated with this machinery failure
# put statistics into hashes keyed by the event code
# we stop at next necessary cause machinery failure if there is one
#
            $n_ev_2 = 0;
            foreach $ev_2 (@{$cas->{event}}) {
                $n_ev_2++;
                next if ($n_ev_2 < $n_event); # ignore earlier events
                last if ($n_ev_2 > $n_event &&
                        ($ev_2->{ec} =~ /^M/ && $event->{Cause} =~ /[PN]/));
                $dead{$ec} += $ev_2->{dead} if $ev_2->{dead} > 0;
                $vol{$ec}  += $ev_2->{vol}  if $ev_2->{vol} > 0;
            }
        }
    }
}
# put event code labels into a hash keyed by the event code
#
$label_file = "../../xml/labels/ec.xml";
$label_tree = XMLin($label_file,
                NormalizeSpace => 2, ForceArray => 1,
                ContentKey => '-content', SuppressEmpty => "");
$labels = $label_tree->{label};  # reference to the labels hash,
#
# print out table including event code description
#
print "   Machinery Failure Event        |Number| Dead |Volume M3|\n";
foreach $ec (sort {$num_cas{$b} <=> $num_cas{$a}} (keys %num_cas)) {
    printf("%-34s|%6d|%6d|%9.0f|\n", $labels->{$ec}->{short},
            $num_cas{$ec}, $dead{$ec}, 1.0e-3*$vol{$ec});
    $total_cas  += $num_cas{$ec};
    $total_dead += $dead{$ec};
    $total_vol  += $vol{$ec};
}
printf("%-34s|%6d|%6d|%9.0f|\n",
      "TOTALS", $total_cas, $total_dead, 1.0e-3*$total_vol);
```

Figure D.7: Machinery Failure Summary

Machinery Failure Event	Number	Dead	Volume M3
Other/unknown machinery	173	6	88663
Probable machinery failure	61	84	198024
Loss of steering	58	182	330605
Blackout	25	42	129400
Crankshaft/piston failure	10	0	0
Lack of maneuverability	9	10	87530
Sea water line leak	6	0	0
Fuel/lube pipe leak	6	6	238
Propeller failure/damage	6	0	1
Shaft/sterntube failure	5	0	0
Boiler failure/fire	5	10	3000
Stern tube leak	4	0	0
Turbocharger failure	4	0	0
Deck machinery failure	3	27	0
Crankcase explosion	3	0	2100
Cylinder liner failure	3	6	1300
Cyl head/exh valve failure	1	0	0
TOTALS	382	373	840862

Figure D.8: Output of Machinery Summary Script