BEAM

Blacknest BDS Event System

Overview of System for Discussion

Project	Blacknest/bds
Date	2020-11-02
Reference	BdsEventSystem
Author	Dr Terry Barnaby

Table of Contents

1. References

1. The BDS System: https://portal.beam.ltd.uk/support/blacknest

2. Introduction

This is a document to cover the addition of an Event/Project handing system to the BDS. There is an immediate requirement to store and provide access to sensor data and metadata from some experiments involving the generation of test seismic signals. Some information on the test signal source along with the sensor data captured is needed to be stored.

As this is similar to a more generic Event /Project system that was originally proposed for the BDS the idea is to produce an Events system for the BDS that can handle the particular experiments requirement as well as more general Events/Projects. In essence the Event/Project system provides a high level name and set of information to define an event at a particular time with optionally a linked set of sensor data files.

3. Design

The BDS can store seismic and other sample data and the associated Metadata and provide access to this in any of the formats the BDS supports.

The BDS will be extended to store information on the Event in an Event object that will be stored in the BDS MySql database. The BDS API will provide access to this Event information, both read and write. The core information on an Event will be a title, the time of the event and the location. It will also be possible to associate particular channels of seismic data with the event for ease of access to the associated data (if there are particular channels of data associated).

4. Event Information

The core information for all events will be:

Field	***Type***	***Description***
id	Integer	Unique id defining this event within the BDS system
title	String	Some text describing the event
type	String	The event type (a hierarchy of types)
startTime	TimeStamp	The startTime of the event to the nearest microsecond. For data access.
endTime	TimeStamp	The endTime of the event to the nearest microsecond. For data access
eventTime	TimeStamp	The actual time of the event to the nearest microsecond
longitude	Float	Longitude location WGS84
latitude	Float	Latitude location WGS84
elevation	Float	Elevation or depth (negative) to WGS84
waterDepth	Float	Water depth of the event if in water
magnitude	Float	Magnitude of the event
magnitudeUnits	String	Do we need this ?
description	String	General description of the Event/Project
notes	String	General notes on the Event/project
extra	Dict<String>	An array of name value pairs for extra metadata specific to particular events.
dataChannels	List<ChannelName>	List of BDS Channels of associated sensor data files if any.

Apart from the title, all fields are optional. The id field is generated with the BDS.

The Event will then be able to have extra fields added in an adhock basis to store specialised information. So there can be an array of additional fields like “chargeWeight”. All of these would store a string value.

4.1. Notes

- All locations and elevations stored to WGS84 datum.
- The type field could be one of “event/*”, “project/*” etc. We will define some standard heiarcal naming convention for these (what types are needed ?).

5. Sensor Data

For any event an associated list of sensor-data can be imported in any of the formats the BDS supports. This currently includes: GCF, CD-1.0, CD-1.1, SEED, BDRS and variants, AD_22_YKA and CSS. We can add other formats if needed.

6. Importing Experiment Data

To import event data the following actions would be carried out:

1. Add the event information (at least the title). This will provide an EventId. Either the bdsAdminUserGui, a newly created bdsEvent command line program or the BDS API (C++, Python, Php) can be used for this.
2. Add the sensor data metadata. This will include at least the calibrationFactor, calibrationUnit for each Network:Station:Channel:Source data channel. The bdsAdminGui, bdsMetadata command line program or the BDS API can be used for this. We are thinking of creating a bdsMetadata program that will be able to import the Metadata from StationXML format files and perhaps other formats in the future. Perhaps the bdsEvent program could perform this as well.
3. Add the sensor data. This will import the raw seismic data files that would be in a format the BDS supports for each Network:Station:Channel:Source data channel. The bdsImportData command line program, bdsImportGui program or the BDS API can be used for this. The imports will use the “Event id” to link seismic data with a particular event.

A simple Shell or Python script can be used to automate the above stages.

7. Exporting Experiment Data

The BDS will support the export of seismic data and metadata in any of the formats it supports by all of its existing methods (bdsAdminGui, bdsUserGui, bdsDataAccess, bdsAutoDrm, bdsWeb and BDS API.

Data can be selected by Network:Stations:Channel:Source as now or by the events id or event title when the Event/Project system is added. The system will also allow the Event Metadata to be exported in simple ASCII and perhaps another format if desired.

The command line bdsEvent program will also be able to export the data for an event along with the Events metadata and channel metadata.

8. Questions on Current Experiment Requirements

8.1. Questions from Beam

1. Are the items listed in the Events object sufficient or are there particular fields of information you wish to store with the Event ?
2. What format will the Sensor data samples be in and will there be a particular file system layout/file name format used (GCF or miniSEED) ?
3. The BDS requires Metadata for each channel containing things like the calibrationFactor, responses, instruments used etc. How will this information be available ?
4. The BdsNotes system could be extended to allow PDF files to be attached to events. Would this be useful ?

8.2. Questions from Sheila

1. Are the data tied intimately to active sources i.e. data records intentionally of specific shots, explosions, earthquakes or other such one-off events ?
2. Are data start times and sampling rates available ?
3. Are station and instrument metadata available ?
4. We need an idea of the total volume of data ?

5. What format or formats are used for storing the data?
6. Are metadata in the data headers sufficient and accurate, particularly timestamps and channel allocations, or will the import process need to include adjustments ?
7. How are the data arranged in the storage medium ? Will the import process need to use the names of files or hierarchical directories to attach metadata (e.g. station, date, time) to the data, and if so, what is the format of these names ?
8. In what format are station and instrument metadata stored?
9. Do they include:
    1. Station position information: lat/long or x/y relative to a fixed point, elevation, depth of emplacement, datum used (e.g. WGS84), and whether the station is part of an array;
    2. Array information if necessary - which stations belong to an array;
    3. Data channel names, and in particular any channels with names not compliant with FDSN standards;
    4. Instrument gain or calibration factor and period/frequency at which this was measured (this is a property of the instrument and digitiser combined);
    5. Instrument frequency response specified as poles-and-zeros or a FAP table; frequency response of digitiser and amplifiers if these impinge on the instrument bandwidth, specified as poles-and-zeros, FAP or FIR filter response coefficients;
    6. Unique identifications of instruments and digitisers (usually serial number) so they can be tied to equipment inventory or manufacturers' data sheets;
    7. Timespans over which individual instruments and digitisers occupied particular sites;

9. BDS Changes

We perform the following change work:

- Add the Event object storage and manipulation to the BdsServer (and hence API)
- Add the ability to select based on an EventId or EventTitle to the BdsServer.
- Add support for adding, viewing and editing Event object info to the bdsAdminGui program.
- Add support for selecting Sensor data and Metadata by EventId or EventTitle to all GUI and command line programs.
- Create a simple bdsEvent command line program to import and export an Events metadata and perhaps the associated datafiles and Metadata.
- Create a bdsImportMetadata program to support import of Metadata in StationXML or other formats.