

TITAN USERS GUIDE

Mike Dixon
Research Applications Laboratory
National Center for Atmospheric Research
Boulder Colorado USA

October 2005

Data flow diagrams

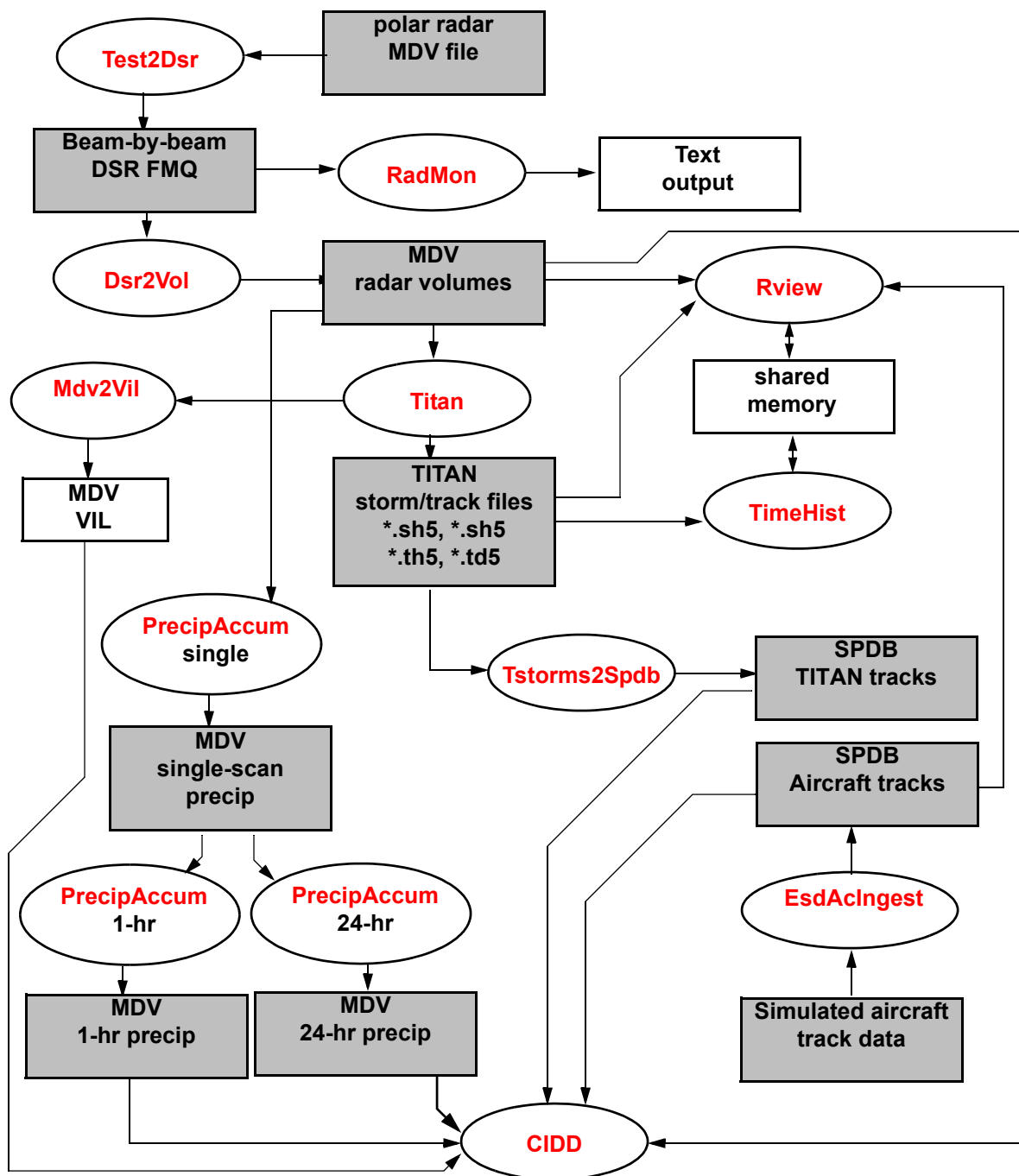
This section contains data-flow diagrams which illustrate some of the more common processing sequences which can be performed by TITAN. The visual representation of the diagrams may convey the concepts more clearly than a detailed text description.

We start with an example of an end-to-end processing sequence, the test project which was introduced in the installation section of this manual. The test project is included with TITAN as a template which users can modify to suite their needs. This test project uses an MDV radar file and an application called **Test2Gate** to simulate a single-radar system. This is a good example because it shows how the various applications may be linked together to form a processing chain.

Subsequent data flow diagrams will show more detailed aspects of the capabilities of the TITAN applications. Processing sequences such as those shown can then be linked together to form more complex systems which handle the user's needs.

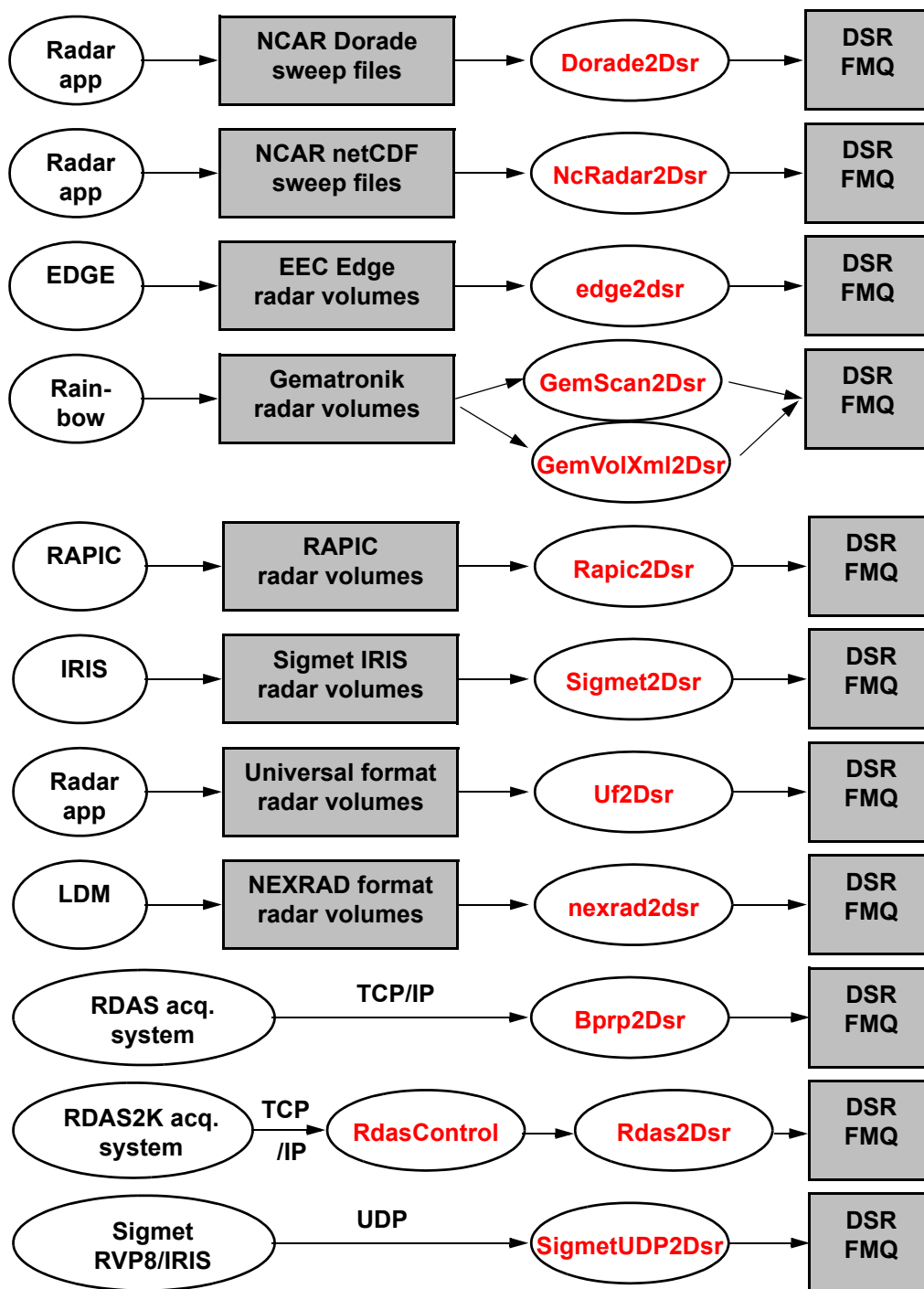
Example of single-radar system - the TITAN test project

The following is the data flow for the test project installed from the TITAN distribution. It shows a typical end-to-end single-radar system. In a real project, the test radar data ingest application will be replaced by a real radar ingest application, which will vary with radar type.



Beam-by-beam radar data to DSR FMQ

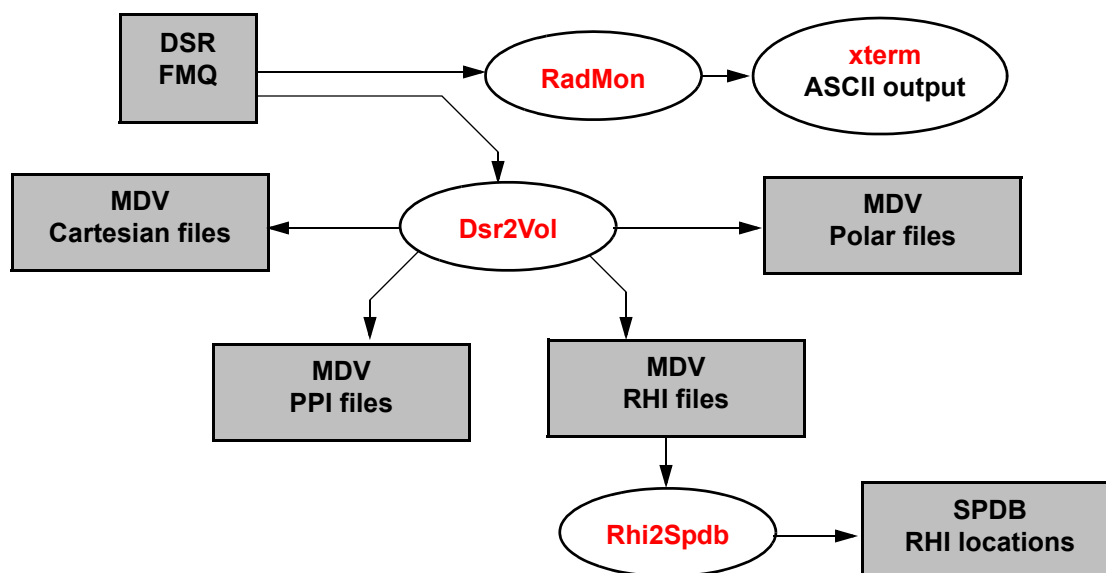
The DSR file message queue (FMQ) forms the standard interface between the details of individual radar systems and TITAN. The following applications are used to convert from radar-specific files or servers to DSR format. DSR2Vol is then used to convert from DSR beam-by-beam format into MDV radar volumes.



Converting beam-by-beam radar data into MDV

The **Dsr2Vol** application is used to convert raw radar data, in beam-by-beam DSR format, into MDV volumes. The output volume can have 4 possible coordinate type:

- Cartesian coordinates: z in km. MSL, y in km., x in km.;
- PPI coordinates: z in degrees elevation, y in km., x in km. This is essentially a Cartesian grid, but with elevation angles in the vertical instead of km.;
- Polar coordinates: z in degrees elevation, y in degrees azimuth, x in km. range. This is the native radar coordinate system;
- RHI coordinates: z in degrees azimuth, y in degrees elevation, x in km. range.



For Cartesian and PPI coordinates, **Dsr2Vol** has the option for either nearest-neighbor or 8-point bilinear interpolation for associating a grid point with radar gate data. For Polar data, the azimuth is rounded to the nearest azimuth resolution unit, typically 1 degree.

Dsr2Vol computes a lookup table of weights for the transformation from radar coordinates to other coordinates. If the radar geometry and scan strategy does not change, this lookup table is re-used from one scan to another. If the geometry or scan strategy changes, a new lookup table is computed.

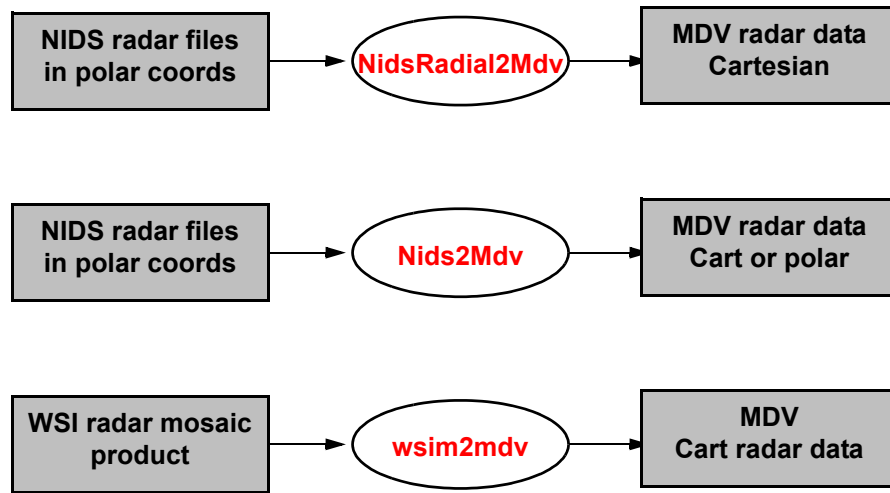
RHI data is stored out with Z representing azimuth, because in MDV data the Z coordinate can accept data on an irregular grid while Y and X only support a regular grid.

The SPDB database is used to store information about the RHI locations, so that these can be rendered on a display, such as **CIDD**. This informs the user about the location of available RHIs, which can then be selected using the vertical section capability of **CIDD**.

Gridded radar data to MDV

Some external radar data is provided in gridded format rather than in beam-by-beam format. For example, some vendors merge data from multiple radars into a Cartesian grid, often referred to as a mosaic. Also, the NEXRAD data often comes in the form of so-called 'level-3' products, in a data stream known as NIDS.

The following applications convert gridded radar data into MDV format.



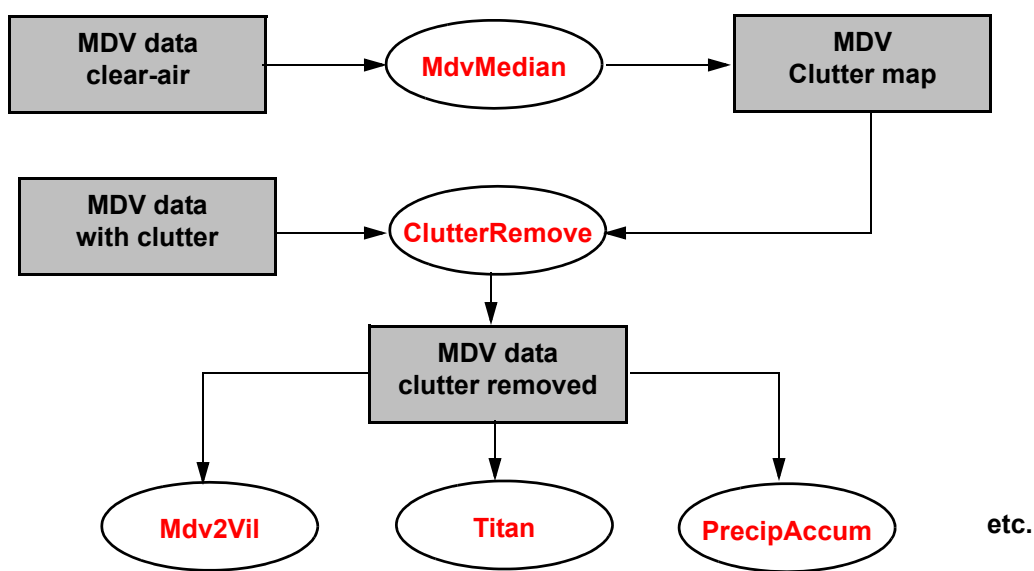
Clutter mitigation

The standard clutter removal technique supported by TITAN involves computing a clutter map from a number of MDV volumes containing ‘clear air’ data, i.e. data with no weather present. 30 to 40 volumes are sufficient for this purpose.

The `MdvMedian` application is used to compute the median reflectivity values for the clear-air scans. These are stored as a clutter map in MDV format.

Then, the `ClutterRemove` application is used to filter out clutter data from radar volumes. Only reflectivity which exceeds the median value by a specified amount (typically 3 or 5 dB) will be retained.

Note that the grid geometry must be consistent throughout. If you want to remove clutter from both polar and Cartesian data volumes, you will need a clear-air data set and a clutter map for both grids.



`MdvMedian` should be set to write the clutter map as a time-stamped MDV file, rather than with a specified path. To do so, set the parameters `write_to_output_path` to `FALSE` and `write_to_output_dir` set to `TRUE`.

Then, in the `ClutterRemove` parameter file, set `use_latest_clutter_file` to `TRUE` and set `clutter_path_url` to point to the directory to which `MdvMedian` wrote its output files.

If set up this way, `ClutterRemove` will find the latest clutter map on disk. If you update the clutter map by collecting new clear-air scans and re-run `MdvMedian`, `ClutterRemove` will automatically find the new map.

Anomalous Propagation (AP) mitigation

Anomalous propagation may be identified by the pattern of the reflectivity echo and by the vertical gradient of reflectivity between consecutive tilts of the radar.

Because of the dependence on relationship of one tilt to another, AP must be filtered in the original polar coordinates. Therefore we do it on the FMQ data stream.

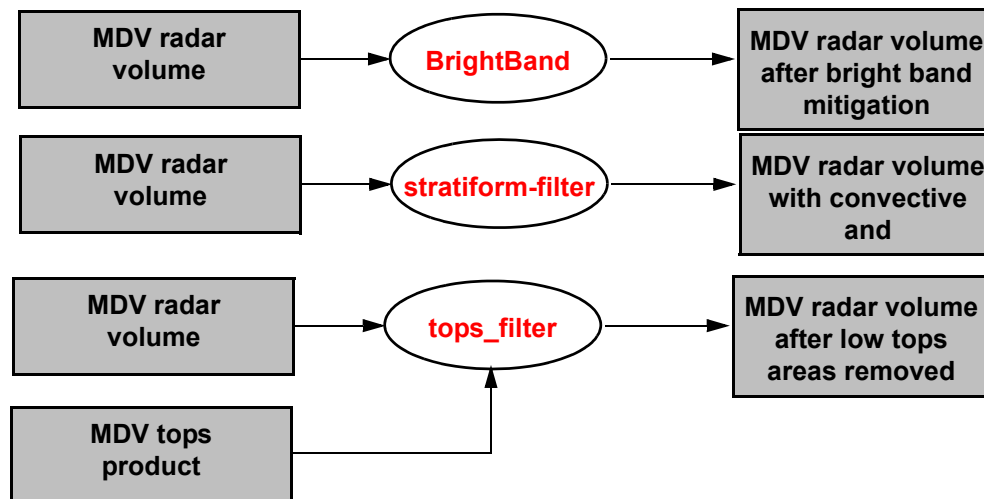


Cartesian volume filtering

There are a number of filtering options which apply to radar in Cartesian volumes. These are:

- **BrightBand**: mitigate the effects of the melting layer (bright band) by trying to identify the bright band and interpolate the reflectivity across it;
- **stratiform_filter**: partition a radar volume into stratiform and convective regions, writing out an MDV file with one field for stratiform regions and one for convective;
- **tops_filter**: based on a gridded top product, filter out regions with tops below a specified value.

All of these filters read data in MDV format and create filtered products also in MDV format.



Computing Vertically Integrated Liquid (VIL) and storm severity

Computing VIL is equivalent to using the following **Z-m** relationship, where m is the vapor density in g/m³, and integrating over the column:

$$Z = 20465 * m^{1.75}$$

Storms which are severe sometimes exhibit the so-called Weak Echo Region (WER) which is a region of low reflectivity surrounded by regions of higher reflectivity. This marks the updraft of the storm. In this region the liquid in the column is concentrated at the upper levels of the storm. We compute a modified VIL field, DVIL, which is computed as the difference between the VIL above some level (say -5C) and the VIL below that level. DVIL is a 2-D field which helps to high-light the most severe storms in a volume.

Mdv2Vil reads in a 3-D radar volume, computes VIL and DVIL and writes out the results to MDV.



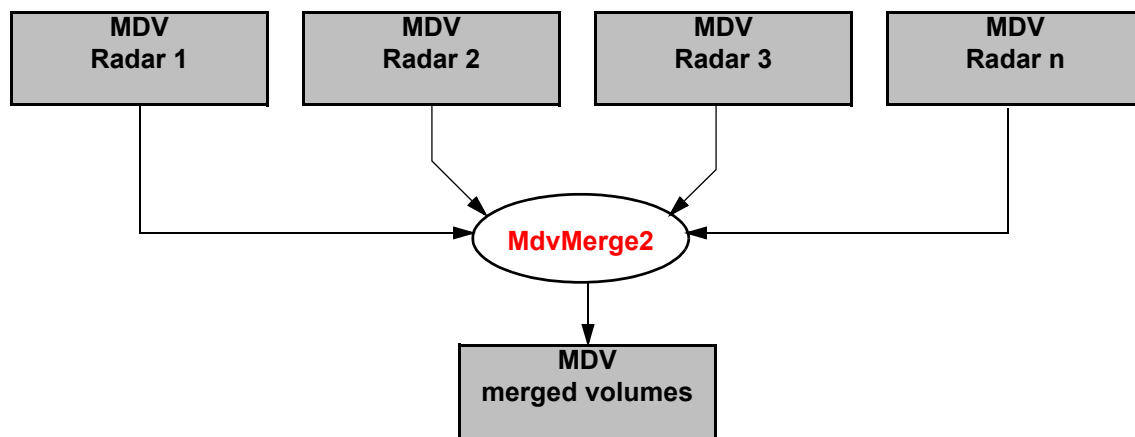
Merging multiple radars

The **MdvMerge2** application allows you to merge grids from individual radars into a single common mosaic.

The projection of the merged grid is independent of the individual grids, so you can change projection during the merging process. You can also specify which fields to merge, and at which precision to write them out.

MdvMerge2 supports full 3-D grids. If the vertical levels of the output grid do not match the input data exactly, the closest plane will be used.

The most difficult aspect of merging in real-time is deciding how to trigger the merge. You can trigger simply based on time, in which case **MdvMerge2** will take the latest data available from each radar at the trigger time. You can also trigger off a primary radar, in which case the merge will occur each time the data from that radar arrives. Read through the parameter file carefully to decide how to set up the trigger.



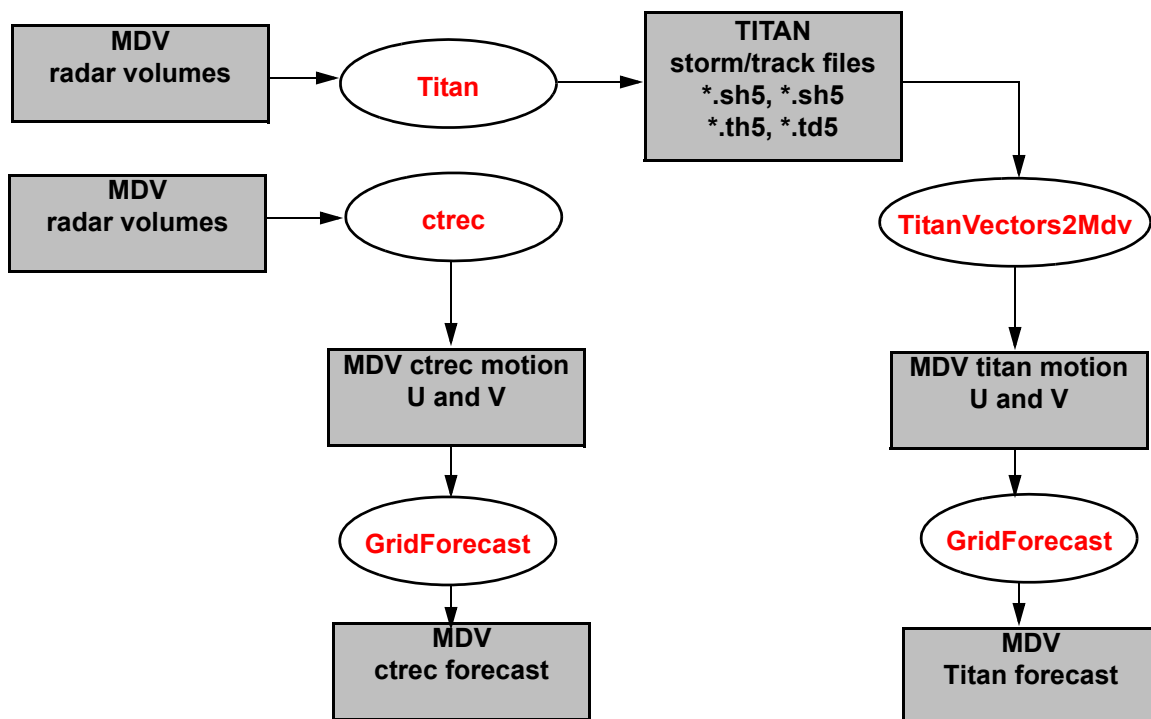
Storm and echo tracking and forecasting

The **Titan** application identifies and tracks storms, dealing with them as distinct entities. It works best with storms which have good edge definition, such as thunderstorms. **Titan** stores its output in storm and track files. The tracking provides information on storm motion, growth and decay, which can be used for producing forecasts based on extrapolation.

Titan does not work as well in stratiform situations. In these cases, the cross-correlation tracker TREC (Tracking Radar Echoes by Correlation) is a good candidate. **ctrec** is the current implementation of TREC. **ctrec** uses cross-correlation pattern recognition to estimate the movement of features in a radar volume from one time to another. **ctrec** produces an MDV file containing the estimated motion in terms of U and V.

The **Titan** track information can also be converted to a U and V field, using the application **TitanVectors2Mdv**. This produces an MDV file similar to **ctrec**, with U and V motion components.

The application **GridForecast** will produce a reflectivity forecast based on a current reflectivity volume and a U and V motion field. The motion vectors are used to advect the current reflectivity into the future using extrapolation.



Precipitation estimation

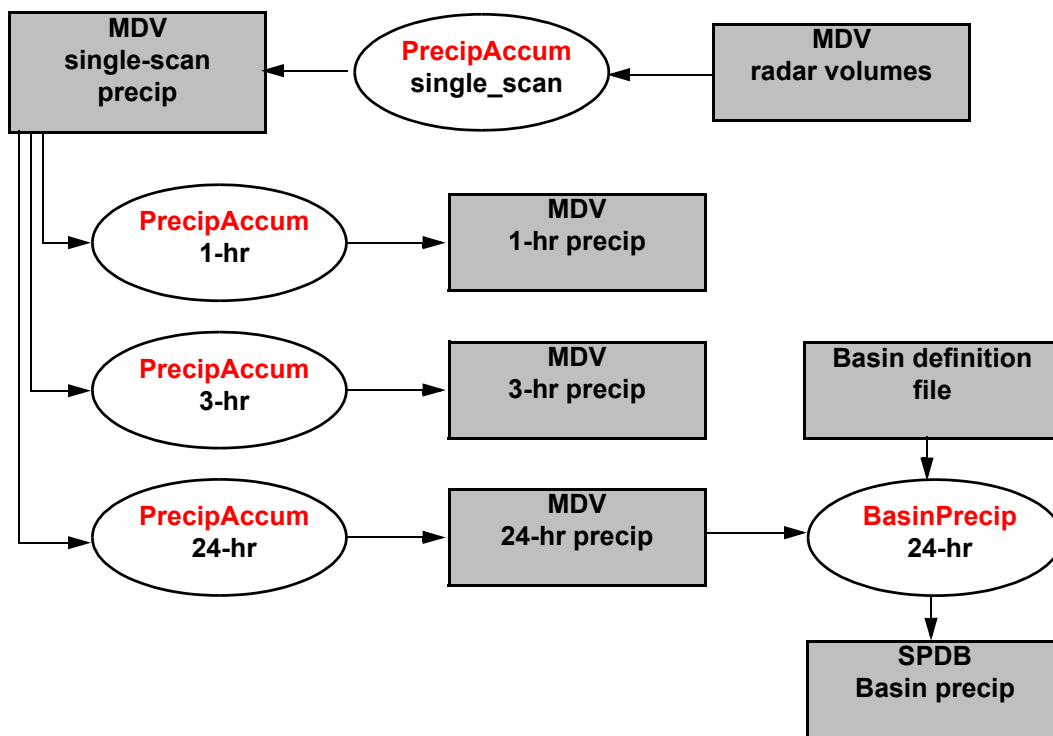
In **Titan** precipitation estimation is, at present, limited to the application of a Z-R relationship to a Cartesian volume to compute precipitation rate and then computing the accumulated precipitation by integrating the rate over time.

The application **PrecipAccum** is used both to compute the rate and perform the accumulation. Several instances of **PrecipAccum** are generally run for this purpose. The reason for computing the rate in a separate step is that this is the most CPU-intensive step and should not be repeated unnecessarily. The rate is actually stored as a depth accumulated for the duration of a single file. The accumulation procedure will use data from the same files many times. Therefore it saves computer resources to compute the rate once.

Clutter causes a problem for precipitation accumulation calculations because a single clutter point will create up as a large precipitation value when the clutter reflectivity is converted into precipitation rate and integrated over time. Therefore, it is preferable to compute precipitation on data from which clutter has been filtered.

The following shows an example of computing the single-scan depth (rate) using one instance of **PrecipAccum** and then running 3 separate instances to compute running accumulation totals for 1-hour, 3-hour and 24-hour products.

We also show an example of running the **BasinPrecip** application to compute 24-hr accumulated precipitation over basins defined in the basin definition file.



Analysis of TITAN storm tracks

The **Rview** and **TimeHist** display pair are designed specifically for viewing the details of the **Titan** storm tracks and their forecasts. **Rview** provides the plan view and vertical section capability, while **TimeHist** provides windows which show the time history, time-height profile, reflectivity distribution and hail metrics for any selected storm.

Tracks2Ascii is an application which writes ASCII data to the terminal window in comma-delimited column format. This format is suitable for transfer of the track and storm properties to spread-sheets, data-bases, plotting packages etc.

TrackGridStats computes the spatial statistics for **Titan** storm tracks and creates MDV files as output. You can then use **Rview** or **CIDD** to view the spatial properties.

VerifyTracks scores the **Titan** forecasts on a track-by-track basis, comparing the forecast position and shape of the storm with that which actually occurred later in the track. It assumes perfect identification and tracking, i.e. perfect knowledge of the past and present, but imperfect knowledge of the future. For a more rigorous verification, the **VerifyGrid** application should be used to score grid forecasts computed by Titan.

