```
#ifndef _ca_driver_h_

#define _ca_driver_h_


#include "gridpack/include/gridpack.hpp"

#include "gridpack/applications/modules/powerflow/pf_app_module.hpp"


namespace gridpack {
namespace contingency_analysis {

enum ContingencyType{Generator, Branch};

/* Defininition of contingency data structure (from powerflow module)
struct Contingency
{
  int p_type;
  std::string p_name;
  // Line contingencies
  std::vector<int> p_from;
  std::vector<int> p_to;
  std::vector<std::string> p_ckt;
  // Status of line before contingency
  std::vector<bool> p_saveLineStatus;
  // Generator contingencies
  std::vector<int> p_busid;
  std::vector<std::string> p_genid;
  // Status of generator before contingency
  std::vector<bool> p_saveGenStatus;
};
*/

// Calling program for contingency analysis application
class CADriver
{
  public:
```

```
Basic constructor

    CADriver(void);


Basic destructor

    ~CADriver(void);


Get list of contingencies from external file
@param cursor pointer to contingencies in input deck
@return vector of contingencies

    std::vector<gridpack::powerflow::Contingency> getContingencies(
        gridpack::utility::Configuration::ChildCursors contingencies);


Execute application
@param argc number of arguments
@param argv list of character strings

    void execute(int argc, char** argv);

    private:
};

} // contingency analysis
} // gridpack

#endif
```