

GCM

Gary - Chicago - Milwaukee ITS Priority Corridor

Corridor Transportation Information Center

Interface Control Specification Document # 9932.04

**PROPRIETARY AND
CONFIDENTIAL**

Prepared by: De Leuw, Cather & Company
University of Illinois at Chicago
Electrical Engineering and
Computer Science Department

Issue Date: April 1, 1997

**GARY-CHICAGO-MILWAUKEE CORRIDOR
CORRIDOR TRANSPORTATION INFORMATION CENTER
INTERFACE CONTROL SPECIFICATION DOCUMENT**

TABLE OF CONTENTS

1	INTRODUCTION	1-1
1.1	PURPOSE	1-1
1.1.1	Goals of this Document.....	1-1
1.1.2	Intended Audience	1-1
1.1.3	Document Organization.....	1-1
1.2	SCOPE	1-1
1.3	DEFINITIONS, ACRONYMS, AND ABBREVIATIONS	1-2
1.4	REFERENCES.....	1-2
2	C-TIC DATA STRUCTURES.....	2-1
2.1	OVERVIEW	2-1
2.2	MESSAGE TYPES.....	2-1
2.2.1	DATASERVER.....	2-1
2.2.1.1	Incident.....	2-2
2.2.1.2	Annotation.....	2-3
2.2.1.3	Travel Times	2-4
2.2.2	CORBA	2-5
2.2.2.1	Incidents	2-5
2.2.2.1.1	NWCD	2-6
2.2.2.1.2	*999	2-6
2.2.2.1.3	Monitor	2-7
2.2.2.2	Detector Data	2-7
2.2.2.3	VMS.....	2-7
2.2.2.4	Route Travel Times	2-8
2.2.2.5	SSI.....	2-8
2.2.3	TSC.....	2-9
2.3	LOCATION REFERENCING	2-9
3	INBOUND MESSAGES.....	3-1
3.1	ILLINOIS.....	3-1
3.1.1	TSC Data Stream.....	3-1
3.1.1.1	General Description	3-1
3.1.1.2	Message Structure and Content	3-1
3.1.1.3	Triggering Event	3-2
3.1.1.4	Responses.....	3-2
3.1.1.5	System Issues	3-2
3.1.2	Northwest Central Dispatch.....	3-2
3.1.2.1	General Description	3-2

3.1.2.2	Information Content.....	3-2
3.1.2.3	Message Structure and Content	3-3
3.1.2.4	Triggering Event	3-5
3.1.2.5	Responses.....	3-5
3.1.2.6	System Issues	3-5
3.1.3	IDOT Closed Loop Signal System (CLSS)	3-5
3.1.3.1	General Description	3-5
3.1.3.2	Message Structure and Content	3-5
3.1.3.3	Triggering Event	3-6
3.1.3.4	Responses.....	3-6
3.1.3.5	System Issues	3-6
3.1.4	SSI Interface	3-6
3.1.4.1	General Description	3-6
3.1.4.2	Information content.....	3-6
3.1.4.3	Message Structure and Content	3-7
3.1.4.4	Triggering Event	3-11
3.1.4.5	Responses.....	3-12
3.1.4.6	System Issues	3-12
3.1.5	*999	3-12
3.1.5.1	General Description	3-12
3.1.5.2	Message Structure and Content	3-12
3.1.5.3	Triggering Event	3-12
3.1.5.4	Responses.....	3-12
3.1.5.5	System Issues	3-13
3.1.6	Illinois State Toll Highway Authority (ISTHA)	3-13
3.1.6.1	General Description	3-13
3.1.6.2	Message Structure and Content	3-13
3.1.6.3	Triggering Event	3-13
3.1.6.4	Responses.....	3-13
3.1.6.5	System Issues	3-14
3.1.7	Regional Transportation Authority (RTA)	3-14
3.1.7.1	General Description	3-14
3.1.7.2	Message Structure and Content	3-14
3.1.7.3	Triggering Event	3-14
3.1.7.4	Responses.....	3-14
3.1.7.5	System Issues	3-14
3.1.8	911	3-14
3.1.8.1	General Description	3-14
3.1.8.2	Message Structure and Content	3-14
3.1.8.3	Triggering Event	3-14
3.1.8.4	Responses.....	3-14
3.1.8.5	System Issues	3-14
3.1.9	IDOT Communications Center	3-14
3.1.9.1	General Description	3-14

3.1.9.2	Message Structure and Content	3-15
3.1.9.3	Triggering Event	3-15
3.1.9.4	Responses.....	3-15
3.1.9.5	System Issues	3-15
3.1.10	State/Local Police	3-15
3.1.10.1	General Description	3-15
3.1.10.2	Message Structure and Content	3-15
3.1.10.3	Triggering Event	3-15
3.1.10.4	Responses.....	3-15
3.1.10.5	System Issues	3-15
3.1.11	Private Broadcasting.....	3-15
3.1.11.1	General Description	3-15
3.1.11.2	Message Structure and Content	3-15
3.1.11.3	Triggering Event	3-15
3.1.11.4	Responses.....	3-15
3.1.11.5	System Issues	3-16
3.1.12	Chicago Skyway	3-16
3.1.12.1	General Description	3-16
3.1.12.2	Message Structure and Content	3-16
3.1.12.3	Triggering Event	3-16
3.1.12.4	Responses.....	3-16
3.1.12.5	System Issues	3-16
3.1.13	Other Traffic Signal Systems	3-16
3.1.13.1	General Description	3-16
3.1.13.2	Message Structure and Content	3-16
3.1.13.3	Triggering Event	3-16
3.1.13.4	Responses.....	3-16
3.1.13.5	System Issues	3-16
3.2	INDIANA.....	3-16
3.2.1	Borman Expressway Traffic Management Center.....	3-16
3.2.1.1	General Description	3-16
3.2.1.2	Message Structure and Content	3-17
3.2.1.3	Triggering Event	3-17
3.2.1.4	Responses.....	3-17
3.2.1.5	System Issues	3-17
3.3	WISCONSIN.....	3-17
3.3.1	MONITOR Traffic Management Center	3-17
3.3.1.1	General Description	3-17
3.3.1.2	Information Content.....	3-17
3.3.1.3	Message Structure.....	3-19
3.3.1.4	Triggering event.....	3-21
3.3.1.5	Responses.....	3-21
4	OUTBOUND MESSAGES.....	4-1

4.1 INTERNET.....4-1
4.2 MEDIA INTERFACE.....4-2

LIST OF FIGURES

Figure 3-1 TSC Loop Detector Data.....3-1

LIST OF TABLES

Table 3-1 NWCD Information Content3-3
Table 3-2 NWCD Message Structure3-4
Table 3-3 SSI Information Content.....3-6
Table 3-4 SSI Message Structure.....3-7
Table 3-5 SSI Precipitation Types and Codes 3-10
Table 3-6 SSI Precipitation Intensities and Codes..... 3-11
Table 3-7 SSI Surface Conditions and Codes..... 3-11
Table 3-8 MONITOR Information Content 3-17
Table 3-9 Validated Lane Data Message Structure 3-20
Table 3-10 Route Travel Time Message Structure..... 3-20
Table 3-11 Incident Information Message Structure 3-21
Table 3-12 VMS Information Message Structure 3-21

**GARY-CHICAGO-MILWAUKEE CORRIDOR
CORRIDOR TRANSPORTATION INFORMATION CENTER
INTERFACE CONTROL SPECIFICATION DOCUMENT**

1 INTRODUCTION

The Requirements Specification Document (#9933.07) identifies the overall system level requirements of the Gary-Chicago-Milwaukee (GCM) Corridor Transportation Information Center (C-TIC). This document provides details about the data and control flow contents among the C-TIC internal subsystems and external data connections.

1.1 PURPOSE

1.1.1 Goals of this Document

The GCM C-TIC Interface Control Specification has the goal of establishing the basis for agreement between the system designers and developers on how each interface is to perform. Additionally, it provides the interface control documentation process to permit the interface design to evolve along with the general project design. This specification reflects current knowledge of C-TIC interfaces.

1.1.2 Intended Audience

The GCM C-TIC Interface Control Specification is intended for:

- The GCM Architecture, Communication, and Information Work Group, in that it provides a system overview of the C-TIC concept.
- Members of the various design groups that have requirements and development responsibility.
- Other interested parties who may be contemplating the design of a similar transportation information clearinghouse system.

1.1.3 Document Organization

This document contains four sections. The first section is an introduction to the document and the GCM C-TIC system. Section 2 describes the C-TIC data structures used for internal and external interfaces. Section 3 describes the inbound messages and is divided by state and source. Section 4 describes the outbound communication structures.

1.2 SCOPE

This document provides interface definitions for flows required in the GCM C-TIC system. This document is concerned with flows between automated components that are internal and external to the C-TIC system.

1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

Document # 9936.02 contains all definitions, acronyms, and abbreviations associated with this project. It also contains information relating to ITS, communications, and other standards.

1.4 REFERENCES

2 C-TIC DATA STRUCTURES

2.1 OVERVIEW

The data structures described in this section are those concerned with the distribution of data either internally or externally. The Message Types section deals with the format of the data as returned through the client process. The Location Referencing section deals with the format for describing the location of traffic data in a uniform, non-proprietary manner.

2.2 MESSAGE TYPES

The C-TIC uses three separate systems for receiving data. The TSCDRV and TSCDRVclient are used to distribute TSC detector data. The DataServer is used to distribute incidents, annotations, and travel times. Finally, the CORBA based classes distribute incidents, weather data, travel times, VMS data, detector data, and route data. Some of the data distributed using CORBA is also distributed using the DataServer. In the following sections, the header files that are shown have been reduced to only the public member functions and data members. Also, the code has been simplified for readability and size, but all of the relevant code necessary for interfacing using C++ or Tcl/Tk has remained. In the cases where there are both C++ and Tcl/Tk interfaces, the C++ is presented.

2.2.1 DataServer

The DataServer sends data to clients who register with the DataClient class. The instantiator of the DataClient class can receive incident data, annotations, or travel time data.

```
enum DataToRtrv {LD=2, INCIDENT=8, ANNOTATION=32, TRAVELTIME=255};

class DataClient : public VIRTUAL NIHObject {
public:
    DataClient(DataToRtrv dtr, unsigned updateInterval =300);
    bool isOK(void);
    bool isDataReady(bool blocking =YES);
    Collection *getLDrpts();
    Collection *getIncidents();
    Collection *getAnnotations();
    Collection *getTravelTimes();
    bool putObjectsInto(Collection& c);
    DataToRtrv getWhatData();
    void setDataCallback(TICAction* action);
    LinkVstr<LoopDetectorReport> locateLoopDetectorReports();
    LinkVstr<Incident> locateIncidents();
    LinkVstr<Annotation> locateAnnotations();
    LinkVstr<TravelTime> locateTravelTimes();
    bool annrefresh();
    bool inforefresh();
    void dataHandler(Connection*);
};
```

DataClient header file and enumeration definition.

2.2.1.1 Incidents

The incidents returned by the DataClient are in a collection and are typically retrieved as links to the objects in the database.

```

class Incident : public PObject, public VIRTUAL NIHObject {
public:

enum IncidentDesc {MOTOR, ACPD, ACPI, ACEN, HMAT, CTF, AFA, OTHER, TS, DV,
                  DUI, PED, CRIME, HITNRUN, ANIMAL, CARCASS, ROADCLOSURE,
                  LANECLASURE, SHOULDERCLOSURE, MOVINGOPER, LANESHIFT,
                  RAMPCLOSURE};
enum IncidentStatus {UNKNOWN,CLEAR,CONFIRMED};
enum IncidentSource {NWCD, AUTO, STAR999, WISDOT, INDOT, OTHERSRC, IDOT,
                    TOLLWAY};

static Link<Incident> createPersistent(Vstr<SegmentID> ids,IncidentDesc
                                     de,IncidentStatus st,IncidentSource sc,unsigned
                                     duration=0,unsigned intensity=0,float impact=0,const char*
                                     adrs="",const char* op="",const char* xref="");
static LinkVstr<Incident> locateAllOlderThan(const NIHTime& t);
static LinkVstr<Incident> locateActive();
static Link<Incident> findLastActive();
VString address();
VString operatorName();
VString xref();
NIHTime timeStamp();
unsigned duration();
void setDuration(unsigned d);
IncidentDesc type();
void setType(IncidentDesc t);
IncidentStatus status();
void setStatus(IncidentStatus t);
IncidentSource source();
void setSource(IncidentSource t);
const LinkVstr<DiSegment> disegments();
unsigned intensity();
void setIntensity(unsigned d);
unsigned lanesClosed();
void setLanesClosed(unsigned d);
float impact();
void setImpact(float i);
o_bool isActive(const NIHTime* atTime =0);
o_bool hasExpired(const NIHTime* atTime =0);
NIHTime startsAt(void);
NIHTime endsAt(void);
void timeStamp(NIHTime t);
void printLocationOn(ostream& s);
static const char* const* incidentTypeNames();
static int numIncidentTypes();
static const char* const* incidentStatusNames();
static int numIncidentStatus();
static const char* const* incidentSourceNames();
static int numIncidentSources();
};

```

Incident header file

2.2.1.2 Annotations

The annotations returned by the DataClient are in a collection and are typically retrieved as links to the objects in the database.

```

class TCL_OBJECT("annotation",Annotation,object,"Map Annotation") : public
    Pobject {

public:

enum Flags {GIF=1, GUI=2, INTERNET=4, ANCHORED=8, LINKED=16};
enum AnchorType {ANCHOR_E=0, ANCHOR_NE=1, ANCHOR_N=2, ANCHOR_NW=3,
    ANCHOR_W=4, ANCHOR_SW=5, ANCHOR_S=6, ANCHOR_SE=7};
static const char* const* anchorDirNames();
static int numAnchorDirNames();

Annotation(Tcl_Interp*);
TCL_SLOT_FULL(CUSTOM, VString, data_, "-data", "data", "Data", "", 0,
    &VString_ConfigOption, "filename or text", "");
TCL_SLOT_FULL(CUSTOM, VString, font_, "-font", "font", "Font", "", 0,
    &VString_ConfigOption, "name of font to use for text
    annotations", "");
TCL_SLOT_FULL(CUSTOM, VString, color_, "-color", "color", "Color", "", 0,
    &VString_ConfigOption, "name of color to use for text
    annotations", "");
TCL_SLOT_FULL(DOUBLE, double, xjustify_, "-xjustify", "xjustify",
    "Justify", "0.5", 0, NULL, "Justification in x dir", "");
TCL_SLOT_FULL(DOUBLE, double, yjustify_, "-yjustify", "yjustify",
    "Justify", "0.5", 0, NULL, "Justification in y dir", "");
TCL_SLOT_FULL(CUSTOM, Coordinate, loc_, "-location", "location",
    "Location", "", 0, &Coordinate_ConfigOption, "Location of
    annotation", "");
TCL_SLOT_FULL(DOUBLE, double, minmag_, "-minmag", "minmag",
    "Magnification", "0.0", 0, NULL, "Minimum magnification
    annotation is valid for", "");
TCL_SLOT_FULL(DOUBLE, double, maxmag_, "-maxmag", "maxmag",
    "Magnification", "10.0", 0, NULL, "Maximum magnification
    annotation is valid for", "");
TCL_SLOT_FULL(INT, int, level_, "-level", "level", "Level", "0", 0, NULL,
    "Level annotation is valid for", "");
TCL_SLOT_FULL(INT, int, flags_, "-flags", "flags", "Flags", "0", 0, NULL,
    "Misc. flags", "");
TCL_SLOT_FULL(INT, int, anchordist_, "-anchordist", "anchordist",
    "Anchordist", "10", 0, NULL, "Distance from anchor in pixels",
    "");
TCL_SLOT_FULL(CUSTOM, int, anchordir_, "-anchordir", "anchordir",
    "Anchordir", "n", 0, AnchorDir_ConfigOption, "Direction from
    anchor to text/image", "");
TCL_SLOT_FULL(CUSTOM, VString, url_, "-url", "url", "URL", "", 0,
    &VString_ConfigOption, "URL of web page to link to", "");
TCL_PROC("anchorDirNames", AnchorDirNamesCmd(Tcl_Interp*), "Returns list of
    all possible anchorDir options");
};

```

Annotation header file

2.2.1.3 Travel Times

The travel times returned by the DataClient are in a collection and are typically retrieved as links to the objects in the database.

```
class TravelTime : public DataTimestamp {
public:
enum TTSource {TOLLWAY};
static Link<TravelTime> createPersistent(SegmentID sid,short tt,float
    spd,short vol,short cc=0,TTSource tts=TOLLWAY);
static Link<TravelTime> createPersistent(SegmentID sid,short tt,float
    spd,short vol, const NIHTime ts, short cc=0,TTSource
    tts=TOLLWAY);
short travelTime();
void travelTime(short t);
float speed();
void speed(float f);
short volume();
void volume(short v);
short congestionClass();
void congestionClass(short c);
TTSource source();
const Link<DiSegment> diSegment();
static const char* const* travelTimeTypeNames();
static int numTravelTimeTypes();
static LinkVstr<TravelTime> locateAllOlderThan(const NIHTime& t);
static LinkVstr<TravelTime> locateAllNewerThan(const NIHTime& t);
};
```

TravelTime header file

```
class DataTimestamp: public PObject, public VIRTUAL NIHObject {
public:
DataTimestamp();
DataTimestamp(const NIHTime &t);
DataTimestamp(const NIHTime &t,const NIHTime &t2);
DataTimestamp(const DataTimestamp &d);
NIHTime timeStamp();
void setTimeStamp(const NIHTime &t);
NIHTime cTICTimeStamp();
void setCTICTimeStamp(const NIHTime &t);
static LinkVstrAny locateAll(const char *n="DataTimestamp");
static LinkVstrAny locateAllOlderThan(const NIHTime& t,const char
    *n="DataTimestamp");
static LinkVstrAny locateAllNewerThan(const NIHTime& t,const char
    *n="DataTimestamp");
static LinkVstrAny locateInInterval(const TimeInterval& t,const char
    *n="DataTimestamp");
};
```

DataTimestamp header files

2.2.2 CORBA

The driver processes receive the raw data from a modem or a file and send the data to a corresponding client object. For most of the data passed, there are corresponding members in the database, but unlike the DataServer, collections of links are not received by the client. Each client receives the data in the form of a data object from the data driver. For instance, the client for NWCD data would receive NWCDdata objects from the NWCD driver.

2.2.2.1 Incidents

Each incident source has a separate data class (i.e. NWCDdata, S999data, MonitorIncData), but they all derive from the IncidentData class.

```
class IncidentData: public DataTimestamp {
public:
    enum IncidentDataStatus { CLEAR, NEW_INCIDENT, CHANGE_VEHICLE_COUNT };
    IncidentData(SegmentID id, Incident::IncidentDesc t, IncidentDataStatus s,
                NIHTime ts, NIHTime ts2);
    IncidentData(const IncidentData& t);
    IncidentDataStatus status();
    void status(IncidentDataStatus i);
    Incident::IncidentDesc incType();
    void incType(Incident::IncidentDesc i);
    SegmentID segmentID();
    void segmentID(SegmentID s);
};
```

IncidentData header file

```
class IncidentClient {
public:
    enum ISource { NWCD, STAR999, RESOLVER, BGRESOLVER, MONITOR };
    enum IType { RAW, RESOLVED };
    IncidentClient();
    static void initialize(IncidentClient *ptr);
    static IncidentClient* instance();
    static void cleanUp();
    static String typeToString(Incident::IncidentDesc t);
    void rcvData(ISource s, IType t, OCEvent *e);
    void removeClient(ISource s, IType t);
    IncidentData *lastData(ISource s, IType t);
    void run();
    void loadFilterFile(ISource s, IType t, String fname);
    void setFiltering(ISource s, IType t, int offOn);
    int isFilteringOn(ISource s, IType t);
    Set filters(ISource s, IType t);
    void addFilter(ISource s, IType t, String f);
    bool removeFilter(ISource s, IType t, String f);
};
```

IncidentClient header file

2.2.2.1.1 NWCD

```
class NWCDdata: public IncidentData {
public:
NWCDdata(char *id, char *xref, char *xref2, NIHTime ts1, NIHTime ts2, char
        *loc, char *locinfo, Incident::IncidentDesc t,
        IncidentData::IncidentDataStatus s, int respChng, SegmentID sid,
        int cu=0, int ns=0);
NWCDdata();
NWCDdataStruct nstruct();
String id();
void id(String i);
String xref();
void xref(String x);
String xref2();
void xref2(String x);
String address();
void address(String a);
String xLocInfo();
void xLocInfo(String x);
int vehicles();
void vChange(int i);
int currentUnits();
void addUnit();
void removeUnit();
int incStatus();
void incStatus(int x);
String removeRet(String x);
};
```

NWCDdata header file

2.2.2.1.2 *999

```
class S999Data: public IncidentData {
public:
S999Data(long id, NIHTime ts1, NIHTime ts2, char *loc, char *xstr, char
        *city, char *cnty, char *st, Incident::IncidentDesc t,
        IncidentData::IncidentDataStatus s, SegmentID sid);

long id();
String address();
String xstreet();
String city();
String county();
String state();

};
```

S999Data header file

2.2.2.1.3 Monitor

```
class MonitorIncData : public IncidentData {
public:
MonitorIncData(long li,char *incid, char *t, short s, short p, char *desc,
NIHTime ts);
String id();
void id(String i);
String description();
void description(String x);
short severity();
void severity(short i);
int plan();
void plan(short i);
};
```

MonitorIncData header file

2.2.2.2 Detector Data

```
class DetectorClient {
public:
DetectorClient(OCEvent *e);
static void initialize(OCEvent *e);
static DetectorClient* instance();
static void cleanUp();
};
```

DetectorClient header file

```
class MonitorLDData: public VIRTUAL NIHObject {
public:
MonitorLDData(int size);
int add(long id,short spd,short vol,short occ,NIHTime ts);
int numberOfLDreports();
long id(int i);
short volume(int i);
short occupancy(int i);
short speed(int i);
NIHTime* timeStamp(int i);
};
```

MonitorLDData header file

2.2.2.3 VMS

```
class SignClient {
public:
```

```
SignClient(OCEvent *e);
static void initialize(OCEvent *e);
static SignClient* instance();
static void cleanUp();
};
```

SignClient header file

```
class SignData: public VIRTUAL NIHObject {
public:
    SignData(int size);
    int add(long i,short s,char *m,NIHTime n);
    int numberOfSignReports();
    String *message(int i);
    long id(int i);
    NIHTime *timeStamp(int i);
    short status(int i);
};
```

SignData header file

2.2.2.4 Route

```
class RouteClient {
public:
    RouteClient(OCEvent *e);
    static void initialize(OCEvent *e);
    static RouteClient* instance();
    static void cleanUp();
};
```

RouteClient header file

```
class RouteData: public VIRTUAL NIHObject {
public:
    RouteData(long i,short s,NIHTime n);
    long route_id();
    NIHTime timeStamp();
    short travelTime();
};
```

RouteData header file

2.2.2.5 SSI

```
class SSIclient {
public:
```

```
SSIClient(OCEvent *e);
static void initialize(OCEvent *e);
static SSIClient* instance();
static void removeInstance();
};
```

SSIClient header file

```
class SSIData: public VIRTUAL NIHObject {
public:
    SSIData(SSIDataStruct n);
    int message();
    int is_alpha();
    NIHTime timeStamp();
    String name();
    String c_value();
    double n_value();
};
```

SSIData header file

2.2.3 TSC

The TSC data can be received using a slightly different interface. The TSC differs from the above methods due to its particular interface requirements and the time when the interface was developed. The client for the TSC detector data must instantiate a TSCclient object and then will receive the data through a socket interface.

```
class TSCDRVClient {
public:
    TSCDRVClient(const char* user, const char* passwd, TICAction* a =0, const
                char* hostname = "locate");
    const TICAction* dataCallback(void);
    void dataCallback(TICAction* a);
    int numberOfDetectors(void);
    int occupancy(int index);
    int volume(int index);
    NIHTime timeStamp(void);
    int version(void);
};
```

TSCDRVClient header file

2.3 LOCATION REFERENCING

TBD.

3 INBOUND MESSAGES

3.1 ILLINOIS

3.1.1 TSC Data Stream

This flow represents the real-time loop detector data collected by the IDOT Traffic System Center (TSC.) The data includes volume and occupancy data collected by loop detectors on specific IDOT controlled expressways.

3.1.1.1 General Description

Volume and occupancy data is transmitted to the C-TIC through this data stream. Incident information and construction/maintenance information will be provided to the C-TIC through alternate means.

Data from the TSC is received via a telephone line/modem connection. The line operates at 9600 baud, 8 data bits, no parity, one start bit, one stop bit. The C-TIC connection is configured as a DTE, emulating a VT-340 terminal.

3.1.1.2 Message Structure and Content

```
$@V__  
hh:mm:ss DD-MMM-YY__  
XXXV1V2V3V4V5V6V7....Vn__  
...  
XXXV1V2V3V4V5V6V7....Vn__  
#.V__  
$@O__  
hh:mm:ss DD-MMM-YY__  
XXXO1O2O3O4O5O6O7....On__  
...  
XXXO1O2O3O4O5O6O7....On__  
#.O__
```

The message occurs once per minute. The \$@V and #.V lines bracket the volume information. The \$@O and \$.O lines bracket the occupancy information. hh:mm:ss DD-MMM-YY are the time and date of the report, which will be identical for volume and occupancy. Each data line consists of an *expressway code* (XXX) (e.g., CAO is Calumet Outbound) followed by 2 digit numbers representing the volume (Vn) or occupancy (On) for each detector. Inoperative detectors are represented by '-1'. Leading zero digits are represented by space. Lines over 79 characters long are split onto the next line.

3.1.1.3 Triggering Event

C-TIC software initiates and maintains the connection to the TSC. Emulating a VT-340 terminal, the C-TIC software dials the TSC, negotiates the baud rate, and establishes connections. The software emulates human interaction to access the Realtime Data 1 minute volume and occupancy screens. Thereafter, the TSC autonomously outputs a complete data set every minute.

3.1.1.4 Responses

If the C-TIC loses the connection to the TSC it will then send an alert to the console operator and attempt to reestablish communications. The C-TIC will log the interruption and resumption of TSC communications on the C-TIC log file.

The TSC may occasionally change the format of the message, especially when new detectors are added. If the message format received by the C-TIC does not match the expected format, a C-TIC operator message will be generated (and logged), and the affected expressway data will be treated as though the detectors are inoperative.

3.1.1.5 System Issues

None.

3.1.2 Northwest Central Dispatch

The C-TIC is connected to the computer aided dispatch center for Northwest Central Dispatch (NWCD) in Arlington Heights, IL. NWCD serves as the Police and Fire dispatch agency for six (6) Northwest Chicago communities. NWCD handles traffic and non-traffic related emergency incidents. Traffic related incidents are filtered at NWCD and sent to the C-TIC.

A leased line exists between NWCD and the C-TIC. A PC was placed at NWCD to monitor records printed on the log printer. When an incident occurs that passes through the PC filter at NWCD, it is then transmitted to the C-TIC.

3.1.2.1 General Description

When an incident is entered into the NWCD database it is also received by the C-TIC software at NWCD. The NWCD PC acts as a filter between the NWCD log printer and the C-TIC. It examines log printer data and classifies each incident observed as filtered or non-filtered. If filtered (i.e., non-traffic incidents such as burglaries or domestics), then the PC discards the information on that line of the printer. If non-filtered (i.e., traffic related incident), then the PC forwards each line to the C-TIC.

3.1.2.2 Information Content

	Interface Type	Duration/Frequency	Release
<ul style="list-style-type: none"> · Incident Declaration Information <ul style="list-style-type: none"> - Incident number - Date and time of incident entry - Entry operator ID - Date and time of incident dispatch - Dispatched operator ID 	Async	Event Driven	1
<ul style="list-style-type: none"> · Incident Response Activity <ul style="list-style-type: none"> - Incident number - En route time and data - On scene time and data 	Async	Event Driven	1
<ul style="list-style-type: none"> · Incident Information <ul style="list-style-type: none"> - Incident number - Initial incident type - Initial Alarm Level - Final incident type - Final alarm Level - Priority - Disposition - Mpage - Group - Source - Location of incident 	Async	Event Driven	1
<ul style="list-style-type: none"> · Location Information: <ul style="list-style-type: none"> - Incident number - Name - Address - Duration - Phone number 	Async	Event Driven	1

Table 3-1 NWCD Information Content

3.1.2.3 Message Structure and Content

Each message starts with the control character "STX" and ends with the control character "ETX". The byte after the ETX is a checksum byte that is used to determine if the message was received intact or not. The LRC is calculated at the C-TIC in the same manner as it is calculated at NWCD.

STX	char	8	\$02
msg data	char	8*n	one line
ETX	char	8	\$03
LRC	char	8	XOR sum of all msg data

If the received LRC character matches at the C-TIC, an ACK character is sent back to the NWCD PC. If the LRC does not match, a NAK is sent and the NWCD PC sends the line again. The NWCD PC also resends a line if no response is received from the C-TIC after one (1) second. After three (3) retries, the NWCD PC discards the line and continues with the next line.

NWCD Incident Message			
Data Structure	Incident No+Date of Dec+Tim of Dec+Dec Operator ID+Date of Dis+Time of Dis+Dis Operator ID+Initial incident type+Initial Alarm level+Final incident type+Final Alarm level+Mpage+Group+Beat + policeIHN + fireIHN + streetName + crossStreet + streetAddr + textMsg		
field	type	bits	description
Incident number	char	32	Incident History Number
Date of declaration	char	16	Date that incident was declared
Time of declaration	char	16	Time of day that incident was declared
Declaration operator ID	char	32	Operator ID
Date of dispatch	char	16	Date that incident was dispatched
Time of dispatch	char	16	Time of day that incident was dispatched
Dispatched operator ID	char	32	ID of operator that dispatched the call
Initial incident type	unsigned	8	Initial determination of incident type
Initial alarm level	unsigned	8	Initial determination of incident alarm level
Final incident type	unsigned	8	Final determination of incident type
Final alarm level	unsigned	8	Final determination of incident alarm level
Mpage	unsigned	8	
Group	unsigned	8	
Beat	unsigned	8	
policeIHN	char	128	NWCD Police Incident History Number
fireIHN	char	128	NWCD Fire Incident History Number
streetName	char	64	Major street name

crossStreet	char	64	Cross street name
streetAddr	char	8	Street address number
textMsg	char	256	A message indicating details of an incident
Message size	864 bits or 108 bytes		

Table 3-2 NWCD Message Structure

3.1.2.4 Triggering Event

The NWCD incident database log printer is continuously monitored. The Incident Message is generated by the NWCD preprocessor when a new traffic related incident has been received and entered into the NWCD system.

3.1.2.5 Responses

The C-TIC will store this information in its database and automatically generate an incident report on the C-TIC. If the C-TIC is unable to automatically identify the location, then the operator will enter the information as an anecdotal input to the system.

3.1.2.6 System Issues

None.

3.1.3 IDOT Closed Loop Signal System (CLSS)

This flow represents the information generated by the Closed Loop Signal System on Dundee Road. The data is currently not documented here. This connection is currently scheduled to be implemented in Release 4.0 or in another future release.

3.1.3.1 General Description

Every five (5) minutes each of the two (2) masters will dial the C-TIC. The C-TIC contains the communication protocols used by the Econolite software package. The C-TIC will receive smoothed volumes and occupancies for each available detector. The data received will be in ASCII format as specified by Econolite and is not documented in this document. The volumes and occupancies will then be converted to their raw values and used to calculate travel times along Dundee Road from IL-53 to Milwaukee Avenue.

3.1.3.2 Message Structure and Content

The message structure is specified by Econolite and is proprietary.

3.1.3.3 Triggering Event

Every five (5) minutes, each of the masters will dial the phone line connected to the C-TIC. There are no retries for the five (5) minute data. If the connection cannot be made, then the current five (5) minute data set will be lost.

3.1.3.4 Responses

The C-TIC will receive the ASCII data and will have to parse out the volumes and occupancies.

3.1.3.5 System Issues

None.

3.1.4 SSI Interface

3.1.4.1 General Description

An SSI computer exists at the IDOT District 1 building that communicates with remote weather sensors. SSI will develop special software that will gather the weather data and store it in a file. When polled, it will transmit the file to the C-TIC. The weather data is then displayed in a window for the operator.

The C-TIC-SSI connection will use the PPP (Point-to-Point Protocol) to establish a LAN (local area network) connection using the TCP/IP network layer protocol.

3.1.4.2 Information content

	Interface Type	Duration/Frequency	Release
<ul style="list-style-type: none"> · Report Messages <ul style="list-style-type: none"> - Strong winds - Low visibility - Snowing - Blizzard alert - Blowing snow - Heavy rain - Air temperature - Wet roads - Possible icing - Possible frost - Freezing rain 	Sync	Event Driven	1
<ul style="list-style-type: none"> · Surface Condition <ul style="list-style-type: none"> - System initialization 	Sync	Event Driven	1

	Interface Type	Duration/Frequency	Release
<ul style="list-style-type: none"> - Communication fail - Dry - Wet - Chem Wet - Snow/Ice - Absorptn - Absrptn2 - Dew - Frost - Ab@DewPt - Frost2 - S/I Alrt 			
<ul style="list-style-type: none"> • Precipitation Type <ul style="list-style-type: none"> - No precipitation - Precipitation present but unclassified - Rain - Snow - Mixed rain and snow - RPU-to-sensor communications failure - Sensor failure · Precipitation Intensity <ul style="list-style-type: none"> - Precipitation intensity information not available - Light - Moderate - Heavy 	Async	Event Driven	1
	Async	Event Driven	1

Table 3-3 SSI Information Content

3.1.4.3 Message Structure and Content

The C-TIC transfers the SSI RPU (Remote Processing Unit) data file every five (5) minutes using the FTP (File Transfer Protocol) over the leased line.

The file structure is determined by SSI.

Weather Message			
Data Structure			
field	type	bits	description
RpId	unsigned	8	Remote Processing Unit (RPU) ID

Weather Message			
Data Structure			
field	type	bits	description
RpDtTm	unsigned	8	RPU timestamp (GMT date/time)
RpCTim	unsigned	8	RPU internal timestamp (GMT sec. since 1/1/70)
ApAirT	unsigned	8	Air temperature (deg C)
ApDewT	unsigned	8	Dew point temperature
ApPcpR	unsigned	8	Precipitation rate (mm/hr)
ApPcpA	unsigned	8	Precipitation accum. since midnight local time (mm)
ApWsAv	unsigned	8	Wind speed average (km/hr)
ApWsGu	unsigned	8	Wind speed gust (km/hr)
ApDirN	unsigned	8	Wind direction (minimum speed) (deg from N)
ApDir	unsigned	8	Wind direction (deg from N)
ApDirX	unsigned	8	Wind direction (maximum speed) (deg from N)
ApRh	unsigned	8	Relative humidity (%)
ApPcT	unsigned	8	Elapsed time since last precipitation (minutes)
ApPcY0	unsigned	8	Precipitation type; sensor #1 (see Table 3-5)
ApPcI0	unsigned	8	Precipitation intensity; sensor #1 (see Table 3-6)
ApPcY1	unsigned	8	Precipitation type; sensor #2 (see Table 3-5)
ApPcI1	unsigned	8	Precipitation intensity; sensor #2 (see Table 3-6)
ApVis	unsigned	8	Surface visibility (km)
SfTemp0	unsigned	8	Pavement surface temp.: sensor #1 (deg C)
SfsubT0	unsigned	8	Pavement subsurface temp.: sensor #1 (deg C)
SfChem0	unsigned	8	Pavement surface chemical factor: sensor #1
SfStat0	unsigned	8	Pavement surface status: sensor #1 (see Table 3-7)

Weather Message			
Data Structure			
field	type	bits	description
SfDep0	unsigned	8	Pavement surface precip. depth: sensor #1 (mm)
SfIce0	unsigned	8	Pavement surface ice index: sensor #1
SfFrzT0	unsigned	8	Pavement surface precip. initial freezing temp: sensor #1 (deg C)
SfCpct0	unsigned	8	
SfTemp1	unsigned	8	Pavement surface temp.: sensor #2 (deg C)
SfSubT1	unsigned	8	Pavement subsurface temp.: sensor #2 (deg C)
SfChem1	unsigned	8	Pavement surface chemical factor: sensor #2
SfStat1	unsigned	8	Pavement surface status: sensor #2 (see Table 3-7)
SfDep1	unsigned	8	Pavement surface precip. depth: sensor #2 (mm)
SfIce1	unsigned	8	Pavement surface ice index: sensor #2
SfFrzT1	unsigned	8	Pavement surface precip. initial freezing temp: sensor #2 (deg C)
SfCpct1	unsigned	8	
SfTemp2	unsigned	8	Pavement surface temp.: sensor #3 (deg C)
SfSubT2	unsigned	8	Pavement subsurface temp.: sensor #3 (deg C)
SfChem2	unsigned	8	Pavement surface chemical factor: sensor #3
SfStat2	unsigned	8	Pavement surface status: sensor #3 (see Table 3-7)
SfDep2	unsigned	8	Pavement surface precip. depth: sensor #3 (mm)
SfIce2	unsigned	8	Pavement surface ice index: sensor #3
SfFrzT2	unsigned	8	Pavement surface precip. initial freezing temp: sensor #3 (deg C)
SfCpct2	unsigned	8	
SfTemp3	unsigned	8	Pavement surface temp.: sensor #4 (deg C)

Weather Message			
Data Structure			
field	type	bits	description
SfSubT3	unsigned	8	Pavement subsurface temp.: sensor #4 (deg C)
SfChem3	unsigned	8	Pavement surface chemical factor: sensor #4
SfStat3	unsigned	8	Pavement surface status: sensor #4 (see Table 3-7)
SfDep3	unsigned	8	Pavement surface precip. depth: sensor #4 (mm)
SfIceI3	unsigned	8	Pavement surface ice index: sensor #4
SfFrzT3	unsigned	8	Pavement surface precip. initial freezing temp: sensor #4 (deg C)
SfCpct3	unsigned	8	
Message size	408 bits or 51 bytes		

Table 3-4 SSI Message Structure

Precipitation Type	Code
No precipitation	0
Precipitation present but not classified	1
Rain	2
Snow	3
Mixed rain and snow	4
RPU-to-sensor communications failure	29
Sensor failure	30

Table 3-5 SSI Precipitation Types and Codes

Precipitation Intensity	Code
Precipitation intensity information not available	0
Light	2
Moderate	3
Heavy	4

Table 3-6 SSI Precipitation Intensities and Codes

Surface Condition	Code
System initialization	0
Communication failure	1
Dry	2
Not Used	3
Wet	4
Chem Wet	5
Snow/Ice	6
Absorptn	7
Absrptn2	8
Dew	9
Frost	10
Ab@DewPt	11
Frost2	12
S/I Alrt	13

Table 3-7 SSI Surface Confitions and Codes

3.1.4.4 Triggering Event

None.

3.1.4.5 Responses

The C-TIC will receive a file from SSI containing weather information from sensors in the test area. The data will then be able to be displayed on the C-TIC operator's screen.

3.1.4.6 System Issues

None.

3.1.5 *999

3.1.5.1 General Description

This message will contain incident information for individual incidents which have been entered into the *999 Incident Processing System located at the *999 Operations Center. The data will include the incident class, incident ID, main roadway name, crossroad or address, segment ID, timestamp, city, county and state.

3.1.5.2 Message Structure and Content

Each message starts with the control character "STX" and ends with the control character "ETX". The byte after the ETX is a checksum byte that is used to determine if the message was received intact or not. The LRC is calculated at the C-TIC in the same manner as described in the NWCD section. The message from the *999 system to the C-TIC has the following format:

```

STX          char    8      $02
message data char   252
ETX          char    8      $03
LRC          char    8      XOR sum of all msg data

```

If the received LRC character matches at the C-TIC, an ACK character is sent back to the *999 PC. If the LRC does not match, a NAK is sent and the *999 PC sends the line again. The *999 PC resends a line if no response is received from the C-TIC after one (1) second. After three (3) retries, the *999 PC discards the line and continues with the next line.

The data portion of the message sent has the following format:

ID	Address	City	County	State	X-Street	Inc. Class	SegmentID	Timestamp
(4 chars)	(62 chars)	(35 chars)	(35 chars)	(2 chars)	(62 chars)	(30 chars)	(8 chars)	(14 chars)

3.1.5.3 Triggering Event

A trigger will be set up in the database at *999 to extract the desired incident information when a new incident is entered in the database. An additional process will then packetize the incident information and send it to the modem.

3.1.5.4 Responses

Incidents are entered into the database as received at the C-TIC. The incident's location may need to be resolved if the segment id is not included.

3.1.5.5 System Issues

The creation of a cellular 911 system has raised some doubts about the necessity of the *999 system.

3.1.6 Illinois State Toll Highway Authority (ISTHA)

3.1.6.1 General Description

This message will contain travel times on the Illinois Tollway that are collected using the I-Pass electronic toll collection system on I-355. ISTHA will also provide construction and maintenance information daily via Fax.

3.1.6.2 Message Structure and Content

The file received at the C-TIC has the following format:

```
<timestamp>  
s1, s2, d, spd, tt, vol, cc  
... (for all pairs of adjacent stations)
```

where s1 is the entrance station, s2 is the exit station, d is a directional bit, tt is the travel time in seconds, vol is the number of vehicles, and cc is the congestion class.

3.1.6.3 Triggering Event

A file is copied remotely to the home directory under the "tds" username.

Construction and maintenance information is faxed to the operators.

3.1.6.4 Responses

Periodically, the "tds" home directory is checked for new files. When new files are detected, they are parsed, and the data is entered into the database.

Construction and maintenance information is entered by the operators using the incident entry window.

3.1.6.5 System Issues

The tollway travel time system has not been reliable.

3.1.7 Regional Transportation Authority (RTA)

3.1.7.1 General Description

The RTA will provide static and dynamic schedule information for the following agencies: RTA, Pace, Metra, and the CTA. The RTA connection is expected to be operational sometime after Release 4.0.

3.1.7.2 Message Structure and Content

3.1.7.3 Triggering Event

3.1.7.4 Responses

3.1.7.5 System Issues

3.1.8 911

3.1.8.1 General Description

This message will contain incident information which is determined to have an impact on traffic conditions. The information will include incident location and type. The 911 connection is expected to be operational sometime after Release 4.0.

3.1.8.2 Message Structure and Content

3.1.8.3 Triggering Event

3.1.8.4 Responses

3.1.8.5 System Issues

It is assumed that this interface will be similar to that for NWCD.

3.1.9 IDOT Communications Center

3.1.9.1 General Description

This message will contain information from the following sources:

- Highway Advisory Radio (HAR)
- Minute Men
- IDOT Maintenance and Construction

This source will provide incident locations and expected durations, weather information, and construction information. IDOT maintenance and construction information will be provided daily via Fax into the C-TIC for Release 2.0. An electronic connection is not expected until at least Release 4.0.

3.1.9.2 Message Structure and Content

3.1.9.3 Triggering Event

3.1.9.4 Responses

Construction and maintenance information is entered by the operators using the incident entry window.

3.1.9.5 System Issues

3.1.10 State/Local Police

3.1.10.1 General Description

An electronic connection is not expected until at least Release 4.0.

3.1.10.2 Message Structure and Content

3.1.10.3 Triggering Event

3.1.10.4 Responses

3.1.10.5 System Issues

3.1.11 Private Broadcasting

3.1.11.1 General Description

An electronic connection is not expected until at least Release 4.0.

3.1.11.2 Message Structure and Content

3.1.11.3 Triggering Event

3.1.11.4 Responses

3.1.11.5 System Issues

3.1.12 Chicago Skyway

3.1.12.1 General Description

An electronic connection is not expected until at least Release 4.0.

3.1.12.2 Message Structure and Content

3.1.12.3 Triggering Event

3.1.12.4 Responses

3.1.12.5 System Issues

3.1.13 Other Traffic Signal Systems

3.1.13.1 General Description

An electronic connection is not expected until at least Release 4.0.

3.1.13.2 Message Structure and Content

3.1.13.3 Triggering Event

3.1.13.4 Responses

3.1.13.5 System Issues

3.2 INDIANA

3.2.1 Borman Expressway Traffic Management Center

3.2.1.1 General Description

The Borman Expressway TMC acts as the focal point for all information from Indiana which is sent to the C-TIC. Although this is the only message inbound from Indiana to the C-TIC, it contains data streams from a variety of sources including:

- State/Local Police
- 911
- Private Broadcasting organizations

- Borman Expressway TMC (VDS, RMS, CMS, HAR, Hoosier Helpers)
- Transit systems
- Weather systems
- InDOT maintenance
- Indiana Toll Authority

An electronic connection is not expected until at least Release 4.0.

3.2.1.2 Message Structure and Content

3.2.1.3 Triggering Event

3.2.1.4 Responses

3.2.1.5 System Issues

3.3 WISCONSIN

3.3.1 MONITOR Traffic Management Center

3.3.1.1 General Description

The MONITOR TMC acts as the focal point for all information from Wisconsin which is sent to the C-TIC. Although this is the only message inbound from Wisconsin to the C-TIC, it will contain data streams from a variety of sources. Releases 3.0 and 4.0 of the C-TIC implementation include the following information types:

- MONITOR TMC (VDS, RMS, CMS, CCTV)
- WISDOT maintenance/construction (scheduled events)

The design of the communication protocol from MONITOR to the C-TIC is described in the "Milwaukee One-Way Data Broadcast Protocol Document." As evident from the title, the data is repeatedly broadcast over a leased line. A checksum is used to determine if the data arrives intact. A new data set is broadcast at one minute intervals.

3.3.1.2 Information Content

For the duration and frequency column, please note that this indicates when the data will change but the C-TIC can only receive the new data when the next 1 minute cycle begins.

	Duration/ Frequency	Release
Validated Link Data:	Every 1 min.	3

	Duration/ Frequency	Release
<ul style="list-style-type: none"> - Time stamp - Link ID - Volume - Occupancy - Speed data - Status 		
<ul style="list-style-type: none"> · Route Travel Times: <ul style="list-style-type: none"> - Time stamp - Route ID - Travel time - Status 	Every 5 min.	3
<ul style="list-style-type: none"> · Incident Information: <ul style="list-style-type: none"> - Time stamp - Link ID - Incident ID - Incident type - Duration - Impact - Description - Status 	Event Driven	3
<ul style="list-style-type: none"> · VMS information <ul style="list-style-type: none"> - Time stamp - Sign ID - Sign text - Status 	Every 5 min.	3
<ul style="list-style-type: none"> · WisDOT maintenance/construction information. This would contain as a minimum, the following schedule event information: <ul style="list-style-type: none"> - Event type - Location time - Duration - Estimated impact 	Event Driven	4+
<ul style="list-style-type: none"> · Weather Data 	Every 15 min.	4+
<ul style="list-style-type: none"> · State, County and Local Police Incident Information. This will contain, as a minimum the following: <ul style="list-style-type: none"> - Incident type - Incident location - Time of detection - Information on roadway conditions 	Event Driven	4+

	Duration/ Frequency	Release
<ul style="list-style-type: none"> · 911 Systems. These systems will provide information on the following: <ul style="list-style-type: none"> - Incident type - Incident location - Time of detection 	Event Driven	4+
<ul style="list-style-type: none"> · Metro Traffic/Shadow Traffic/other will provide information on roadway congestion and incidents such as: <ul style="list-style-type: none"> - Location - Estimated travel time 	As Available	4+
<ul style="list-style-type: none"> · Traffic Signal System will provide information on incident/malfunctions including: <ul style="list-style-type: none"> - Type - Incident type - Location - Estimated duration - from now - Detection time - Clearance time - Travel time data based on a link number on a five minute basis - 5/15 minute detector data 	Event Driven for Incidents Every 5 min. for detector data	4+
<ul style="list-style-type: none"> · WisDOT will provide information on Hazmat monitoring data which will contain: <ul style="list-style-type: none"> - Detailed Hazmat information (i.e., route, time, type of load) - Incident type, incident location, etc. 	Event Driven	4+
<ul style="list-style-type: none"> · Regional Bus/Railway Authority Information. These systems would provide information on: <ul style="list-style-type: none"> - Current bus/rail schedules - Real-time transit information - Incident information such as location, type, estimated duration. 	Event Driven and Every 15 min. (As required for schedules)	4+

Table 3-8 MONITOR Information Content

3.3.1.3 Message Structure

For Release 3.0, there are four (4) distinct messages that will be sent to the C-TIC from MONITOR. These messages are:

- Validated Lane Data
- Route Travel Times
- Incident Information
- VMS Information

The following describes the contents of the above messages. The specific format is contained in the above mentioned document.

field	description
Link ID	MONITOR link ID
Status	Status of loop detector information
volume	Average Volume Veh/Hour
occupancy	Average occupancy
speed	Average Speed Miles/Hour

Table 3-9 Validated Lane Data Message Content

field	description
Route ID	Route ID
Travel time	Travel time on Route ID

Table 3-10 Route Travel Time Message Content

field	description
Incident ID	Incident ID assigned at MONITOR
Link ID	MONITOR link ID
Lanes	Lanes affected
Position	Location along link
Severity	Impact on travel time
Stime	Start time
Etime	End time
Involves	Number of vehicles involved

field	description
incd_type	Incident type
source	Text source description
source_id	Source of incident data
est_source	
manager	Manager at TMC
qualifier	
description	Incident description
plan_id	Incident plan used at TMC

Table 3-11 Incident Information Message Content

field	description
Status	Status of VMS information
ph2Idx	Time when data was created
Sign ID	Sign identifier
Starttime	Start of message display
endtime	End display of message
Sign Text	Sign message

Table 3-12 VMS Information Message Content

3.3.1.4 Triggering event

The MONITOR system will send data continuously. The C-TIC MONITOR driver process will read and interpret the data stream. Separate client processes will receive the data and enter it into the database or make it available to the web.

3.3.1.5 Responses

If the C-TIC loses the connection to MONITOR it will attempt to reestablish communications. The C-TIC will log the interruption and resumption of MONITOR communications in the C-TIC log file.

MONITOR may occasionally change the format of the message, especially when new detectors are added. If the message format received by the C-TIC does not match the expected format, a C-TIC log message will be generated (and logged), and the affected data will be treated as though the detectors are inoperative.

4 OUTBOUND MESSAGES

This section details the messages generated within the C-TIC and distributed over various media to other agencies, value added resellers, and to the public.

4.1 INTERNET

The Internet connection consists of the entire Gary-Chicago-Milwaukee Web site. The GCM Web site is produced using hypertext markup language (HTML) files and graphics interchange format (GIF) images which reflect the current traffic conditions.

Congestion and travel time information will be obtained from the IDOT TSC, MONITOR TMC, the Borman TMC, and the Illinois State Toll Highway Authority TDS computer. This information will be presented to the general public in the form of congestion maps which contain color-coded roadways in the Gary, Chicago and Milwaukee areas.

The password protected web pages will contain incident details, weather data, VMS data, route travel times, and maps which are only accessible to participating agencies. The general public will have access to congestion information, construction and maintenance data, and incident type and location information.

Release 1 will present the following information on the Internet:

- Illinois congestion information
- Illinois route travel time.

Release 2 will add the following to the information from Release 1:

- Maintenance and construction information
- Incident information (protected page).

Release 2a will add the following to the information from the previous releases:

- PIC server hot-link

Release 3 will add the following to the information from the previous releases:

- Multiple maps
- Wisconsin congestion (protected)
- Wisconsin travel time (protected)
- Wisconsin incidents (protected)
- Wisconsin VMS (protected)
- *999 incidents (protected)
- Indiana VMS/HAR (protected)
- Weather (protected)

Release 4 will add the following to the information from the previous releases:

- Indiana congestion (protected)

- Indiana travel time (protected)
- Indiana incidents (protected)
- Zoomable maps (protected)

4.2 *MEDIA INTERFACE*

The Media Interface will contain data to be distributed to the general public. The information will be presented using the standard location referencing system. Interface for this data has not been determined.