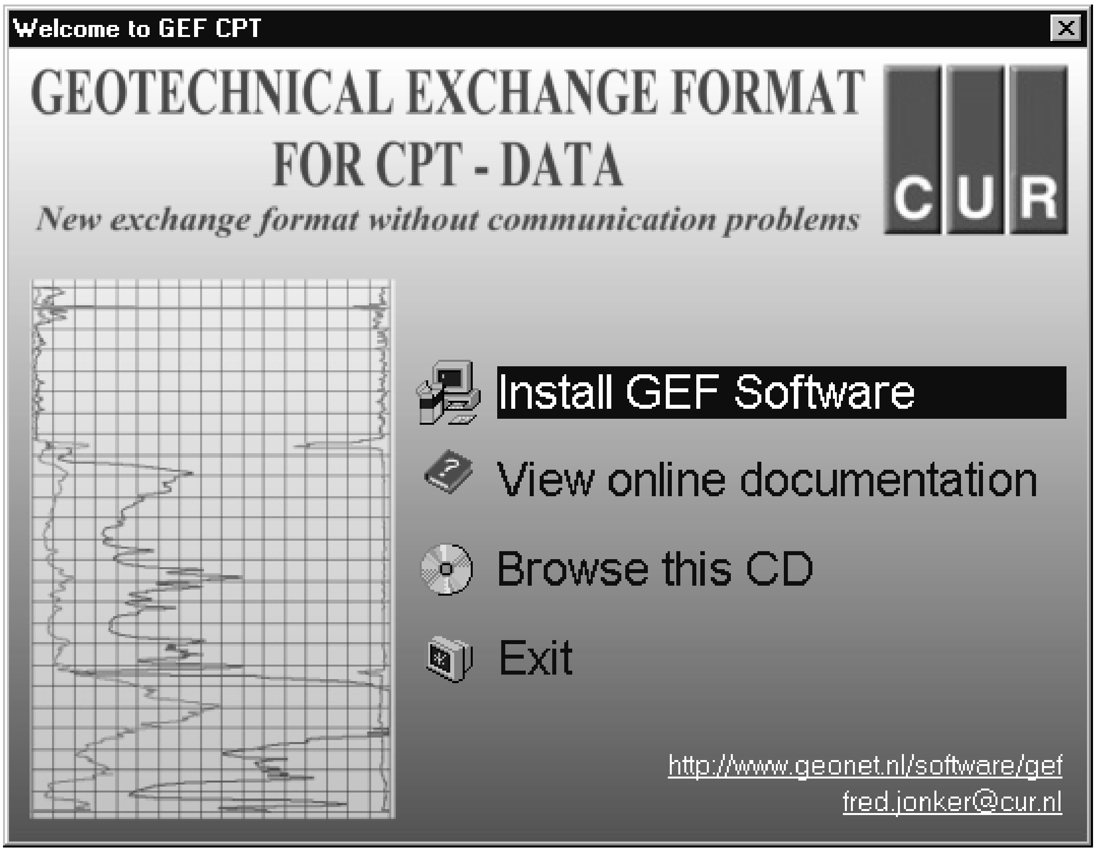
GEOTECHNICAL EXCHANGE FORMAT FOR CPT-DATA



June 1999

18051999 1626

PREFACE

For geotechnical in-situ investigation the cone penetration test has been used world-wide. Based on the results of this type of soil investigation insight in the build-up and the bearing capacity of the subsoil will be gathered.
Usually the output of the cone penetration test is available in a digital form according to a customer specific format. This seems to be an ideal situation, as each customer gets this output in the format he asked for. However, the result is that there is still a number of formats, which are hard to be interchanged thus being a hindering for a sustainable storage of these data. Moreover, due to the variation in formats there is a continuous need for different conversion programmes for the use of these digital cone penetration data in the design of structures.
Besides the mentioned disadvantages, this variation in formats leads to waste of time, money and energy. For a start in the Netherlands these costs amount to at least NLG 600.000 or Euro• 270.000 a year (price level 1999).

This situation has induced the development of a standard format.
The first step in this process was an inquiry among a large number of users in order to determine the content of the standard format. The next step was a research into the available international formats. Comparison of this research with the results of the inquiry leads to the choice of one existing format as a basis to derive the new standard. Besides the new standard format a number of tools are developed, a/o. verification software and a viewer. By now several Dutch companies and organisations are using this standard format. In the mean time a number of computer applications is making use of this new standard format.

At the time of compiling of this manual, the research committee “Standardisation digital CPT-results” comprised:

Dr. H. den Adel	GeoDelft
P. van Asten	A.P. v.d. Berg
C.N. Bremmer	Netherlands Institute of Applied Geoscience TNO
F.J.J.A. van Dam, *chairman*	Ministry of Transport, Public Works and Water Management, Directorate-General for Public Works and Water Management, Civil Engineering Division
F.P.H. Engering	GeoDelft
A. Jonker, *coordinator*	CUR
S.K. Jonker	Ministry of Transport, Public Works and Water Management, Directorate-General for Public Works and Water Management, Road and Hydraulic Engineering Division
A.G.M. Knibbeler	MOS Geotechnics
W.J. van Niekerk	Ballast Nedam Engineering
W.A. Nohl, *secretary*	Fugro Ingenieursbureau
Prof. J.K. Vrijling, *mentor*	CUR

Besides these members of the research committee a number of corresponding members were involved:

W.M. Bielefeld	IJsselmeer Beton
L. Buth	The Netherlands Standardisation Institute
M.G. van Dierendonck	Adviesbureau Tjaden
P.G. van Duijnen	Holland Railconsult
M.G.A. van den Elzen	T & E Consult
P. van Goudoever	VOTB
H.J.M. Lamers	Ascal
F.W.M. Mol	Ministry of Transport, Public Works and Water Management, Directorate-General for Public Works and Water Management, Civil Engineering Division
K. Peppelman	Technosoft
I.L. Ritsema	Netherlands Institute of Applied Geoscience TNO
E.J.C. Wassenaar	GeoMil Equipment

The finances for developing the standard format were provided by
- Ministry of Transport, Public Works and Water Management, Directorate-General for Public Works and Water Management, Civil Engineering Division
- NS Rail Infrastructure
- Ballast Nedam Engineering
- Public Works Rotterdam
- Fugro Ingenieursbureau
- GeoDelft
- MOS Geotechnics
- NITG TNO
- DHV Environment and Infrastructure
- Haskoning
- LWI – Land Water Environment Information Technology
- CUR – Centre for Civil Engineering Research and Codes

The software and this report are available free at www.geonet.nl/software/gef, with the possibility to make a download.
The CUR-organisation guarantees that the format will be maintained; if there are any questions or tips, please inform us.
Infodesk: Fred Jonker, ☎ +31 182 54 06 30

June 1999 The Board of the CUR

TABLE OF CONTENTS

1 INTRODUCTION

A new exchange format for digital cone penetration test data

Cone penetration tests are extensively used in soil mechanics to determine the layer structure and the bearing capacity of the subsoil. Data from the cone penetration test is stored digitally and supplied to the client in the format they request. The result is a large variety of formats which makes mutual exchange problematic and hinders durable storage of penetration test data. The standard format in accordance with GEF-CPT-Report brings an end to this confusing situation.

Why a new format?

Existing formats are specifically directed at a client and are therefore not unambiguous and exchangeable. In cases where there are large quantities of penetration test data delivered in a variety of formats, this not only leads to communication problems but also delays and increased costs.

New format in accordance with GEF-CPT-Report

Uniform:

Conversions belong to the past; the format offers completeness, as the structure stores all relevant information.

Flexible:

In addition to certain minimal information, the user can add specific data.

Unambiguous:

Storage of the measurement data is unambiguous. In the case of measured pore water pressure, for instance, the level at which this is measured is also stated (in the cone or at a higher level). It is also possible to include more than one pore water pressure measurement.

Directly applicable:

'Links" with the GEF-CPT-Report were made for various companies during the development traject. This means that there is already application software which can use penetration test data directly from the GEF-CPT-Report. A "link" is also supplied to convert GEF-CPT-Report files into NENGEO format (and conversely). Furthermore, the direct applicability is enhanced by the availability of an "Excel-link" for importing and processing penetration test data as required.

Freely available tools:

The format is provided with freely available tools for verifying the supplied data and for making "links". The verification software checks whether the structure of the data file complies with that of the GEF-CPT-Report (not a contents verification of the data file, but a check whether the structure satisfies GEF-CPT-Report). For verification purposes, a simple viewer is also provided.

Usefulness
The continuous development traject, involving a large number of experts, has resulted in high quality and efficiency. It will be possible to use the new format efficiently, particularly on large projects involving many parties.

GEF and GEF-CPT-Report

- GEF (Geotechnical Exchange Format) is a general language structure for storing and transferring geotechnical information. GEF states in a procedural way how a new structure (a set of rules for the storage of measurements) should be composed. This report provides a basis for a general exchange of digital cone penetration test data.
- GEF-CPT-Report is the specific structure for reporting cone penetration test data.

GEF Verification is a stand-alone application which can be used to verify if files comply with the new GEF-CPT-Report Format. Verification consists of checking for syntax errors and contextual inconsistencies in digital CPT-files using the GEF-CPT-Report Format. The software opens a GEF-file and performs a verification. A log of the errors that are found is given. Batch verification of GEF-files is also possible. In the case of the verification of a single file, the file itself as well as a graph, displaying the most important data, viz. cone resistance, sleeve friction, friction ratio and pore water pressure, can be displayed.

GEF Verification 1.0 was developed by NITG-TNO for LWI and released in September 1998.

GeoDelft and NITG-TNO have assembled a library which can be used to create conversion programs between other formats and GEF. This library was named GEFLIB.

A conversion tool named ***GEF-NENGEO/NENGEO-GEF Conversion*** was developed to convert between the CPT-file format NENGEO and the GEF-CPT-Report format.

This document will describe the following:

1. The GEF language
2. The format for reporting CPT results
3. The GEF verification tool
4. The GEF library
5. The GEF-NENGEO/NENGEO-GEF conversion tool

2 FORMAT OF A GEF-CPT-REPORT FILE

In accordance with GEF, the file contents is divided into two main components: the header and the data block. The header contains the following information:

- summary of the original information sources (file tracing)
- description of the cone penetration data (data descriptive)
- description of the subsequent measurement and calculation procedures (procedure descriptive)
- structure information about the file (file descriptive).

The data block is a table containing the measured data and the calculated information.

2.1 Header

A complete list of the items which can be used and which may appear in the GEF-CPT-report header is given below. A default value is given for several items. If this default value is applicable, the item can be omitted.

Items printed in **bold text** in the following sections are **always** to be included in a GEF-CPT-report file, while the other items are optional.

2.1.1 File tracing

- **Name of executing company**
- **Order number**
- **Date when GEF-CPT-Report file is created**
- **Identification number of the cone penetration test**
- **Employee, responsible for the data**

2.1.2 Data descriptive

- Data format (this must be ASCII for a GEF-CPT-Report)
- **Number of columns in the data block**
- **Number of lines in the data block**
- **Column information = description for each column**
- Definition of “invalid measurement value” for each column.

2.1.3 Procedure descriptive

Project information
- Project name
- Client name
- Name of location.

Apparatus and procedures
- Type and serial number of the penetration cone
- Nominal surface area of the cone tip in mm^2
- Nominal surface area of the friction casing in mm^2
- Net surface quotient of cone tip and friction casing
- Distance between cone and centre of friction casing in mm
- Location of the PPT filter (u1, u2 or u3)
- Type of cone penetration test (Cone, possibly supplemented with Friction, PPT and/or inclination)
- Use of back-flow compensator (yes/no)
- Reference to standard (NEN 5140 or 3680)
- Class of penetration test in accordance with NEN 5140.

Location of penetration test
- Coordinate system used
- X and Y coordinates of the penetration test location
- Reference level (Z coordinate)
- Height of the selected fixed horizontal surface (= usually ground level) with respect to the reference level
- Description of pre-drilled/pre-excavated soil
- Groundwater level
- Unusual circumstances.

Measuring
- Date the penetration test was carried out
- Time when penetration test began
- Depth of first observation with respect to the selected fixed horizontal surface (for pre-testing, excavation or -drilling)
- Final depth of penetration test
- Stop criteria
- Zero values of all measurement elements before and after penetration test
- Minimal and maximum values per column.

Comments
- Mass and geometry of the probe apparatus, including anchoring
- Method for processing interruptions
- Correction method for zero drift
- Remarks.

2.1.2.1 Analysis

- Calculation formula for calculated columns or references to the standard or a publication where this method of calculation is described

Reporting
- Printing format for the columns in the data block

2.1.4 File descriptive

- **GEF version number**
- **GEF-CPT-Report Procedure Code**
- Type of computer system.

2.2 Data block

The data block is composed of a variable number of columns, of which two columns are reserved for the two compulsory measurement values: the penetration length and the cone resistance. The other columns are available for the remaining direct measurement values and the derived results. A number of quantities are defined in Section 3.4.

A text field can be appended as the last column in the data block to add comments on a specific measurement scan, for example reference to a dissipation test.

3 CPT REPORT FILE IN GEF FORMAT

3.1 Code words used in the CPT Report

The starting point of the CPT Report file is that the measurement variables and text which it sets out have a fixed position. In accordance with the GEF definition, there is room for 1500 measurement variables and texts, where the first 128 variables and texts within the CPT Report definition are reserved for (future) “fixed” data (see Sections 3.2 and 3.3). Users can extend the list by using further free variables and texts (numbers 129 up to and including 1500).

The exact definition of the keywords is defined in “GEF, release 1.0.0” (see appendix 2).

Four types of variables are given in the list below, indicated between []:

- Character = a single readable symbol
- Text = readable text with a maximum of 256 characters, without commas or #s
- Number = an integer number
- Figure - a floating point number.

Items printed in **bold text** in the list below must **always** be present in a CPT Report file, while the other items are optional. The italic text provides brief information about the relevant keyword.

#GEFID	**=**	**1,0,0** ***release number GEF***
#COLUMN	**=**	**[number]** ***the number of columns in the data block***
#COLUMNINFO	=	[number], [text], [text], [number] *column number, dimension, quantity, quantity number. See table in section 3.4*
#COLUMNINFO	**=**	**1, m, penetration length, 1**
#COLUMNINFO	**=**	**2, MPa, Cone value, 2**
#COLUMNMINMAX	=	1, [figure], [figure]
#COLUMNMINMAX	=	2, [figure], [figure]
#COLUMNMINMAX	=	3, …, … etc.
#COLUMNSEPARATOR	=	[character] *text dividing columns (default = space)*
#COLUMNTEXT	=	[number], [text] *text on or off* 0 = off 1 = on
#COLUMNVOID	=	1, [figure] *definition of "no value"*
#COLUMNVOID	=	2, [figure]
#COLUMNVOID	=	3, ... etc.
#COMPANYID	**=**	**[text], [text], [integer]** ***executing company***
#DATAFORMAT	=	ASCII (this is the compulsory data format)
#FILEDATE	**=**	**[number], [number], [number]** ***yyyy, mm, dd***
#FILEOWNER	**=**	**[text]**
#LASTSCAN	**=**	**[number]** ***number of measurement scans in data block***
#MEASUREMENTTEXT	=	1, [text], client
#MEASUREMENTTEXT	=	2, [text], project name
#MEASUREMENTTEXT	=	3, [text], name of location

#MEASUREMENTTEXT	=	4, [text], cone type and serial number
#MEASUREMENTTEXT	=	5, [text], Mass and geometry of probe apparatus\, incl. anchoring
#MEASUREMENTTEXT	=	6, [text], according to standard *NEN 5140 incl. Class*\, *NEN 3680,….*
#MEASUREMENTTEXT	=	7, [text], own coordinate system
#MEASUREMENTTEXT	=	8, [text], own reference level
#MEASUREMENTTEXT	=	9, [text], fixed horizontal level (= usually ground level)
#MEASUREMENTTEXT	=	10, [text], orientation of biaxial inclination measurement (N-direction)
#MEASUREMENTTEXT	=	11, [text],unusual circumstances
#MEASUREMENTTEXT	=	20, [text], correction method for zero drift
#MEASUREMENTTEXT	=	21, [text], processing method for interruptions
#MEASUREMENTTEXT	=	22, [text], remarks
#MEASUREMENTTEXT	=	23, [text], remarks
#MEASUREMENTTEXT	=	30, [text], calculation formula for column …
#MEASUREMENTTEXT	=	31, [text], calculation formula for column …
#MEASUREMENTVAR	=	1, [figure], mm^2, Nom. surface area of cone tip
#MEASUREMENTVAR	=	2, [figure], mm^2, Nom. surface area of friction casing
#MEASUREMENTVAR	=	3, [figure], -, Net surface area quotient of cone tip
#MEASUREMENTVAR	=	4, [figure], -, Net surface area quotient of friction casing
#MEASUREMENTVAR	=	5, [figure], mm, cone distance to centre of friction casing
#MEASUREMENTVAR	=	6, [number], -, friction present
#MEASUREMENTVAR	=	7, [number], -, PPT u1 present
#MEASUREMENTVAR	=	8, [number], -, PPT u2 present
#MEASUREMENTVAR	=	9, [number], -, PPT u3 present
#MEASUREMENTVAR	=	10, [number], -, inclination measurement present
#MEASUREMENTVAR	=	11, [number], -, use of back-flow compensator
#MEASUREMENTVAR	=	12, [number], -, type of penetration test
#MEASUREMENTVAR	=	13, [figure], m, pre-excavated depth
#MEASUREMENTVAR	=	14, [figure], m, groundwater level
#MEASUREMENTVAR	=	15, [figure], m, water depth (for offshore activities works)
#MEASUREMENTVAR	=	16, [figure], m, end depth of penetration test
#MEASUREMENTVAR	=	17, [number], -, Stop criteria
#MEASUREMENTVAR	=	20, [figure], MPa, zero measurement of cone before penetration test
#MEASUREMENTVAR	=	21, [figure], MPa, zero measurement of cone after penetration test
#MEASUREMENTVAR	=	22, [figure], MPa, zero measurement friction before penetration test
#MEASUREMENTVAR	=	23, [figure], MPa, zero measurement friction after penetration test
#MEASUREMENTVAR	=	24, [figure], MPa, zero measurement PPT u1 before penetr. test
#MEASUREMENTVAR	=	25, [figure], MPa, zero measurement PPT u1 after penetr. test
#MEASUREMENTVAR	=	26, [figure], MPa, zero measurement PPT u2 before penetr. test
#MEASUREMENTVAR	=	27, [figure], MPa, zero measurement PPT u2 after penetr. test
#MEASUREMENTVAR	=	28, [figure], MPa, zero measurement PPT u3 before penetr. test
#MEASUREMENTVAR	=	29, [figure], MPa, zero measurement PPT u3 after penetr. test
#MEASUREMENTVAR	=	30, [figure], degrees, zero measurement inclination before penetr. test

#MEASUREMENTVAR	=	31, [figure], degrees, zero measurement inclination after penetr. test
#MEASUREMENTVAR	=	32, [figure], degrees, zero measurement inclination NS before penetr. test
#MEASUREMENTVAR	=	33, [figure], degrees, zero measurement inclination NS after penetr. test
#MEASUREMENTVAR	=	34, [figure], degrees, zero measurement inclination EW before penetr. test
#MEASUREMENTVAR	=	35, [figure], degrees, zero measurement inclination EW after penetr. test
#PROCEDURECODE	**=**	**GEF-CPT-Report, 1,0,0 *release of CPT-Report***
#PROJECTID	**=**	**[text], [text], [text] *order number***
#RECORDSEPARATOR	=	[character] *symbol at end of a measurement scan (default = CR/LF)*
#REPORTCODE	=	[text], [number], [number], [number]
#REPORTDATAFORMAT	=	[character] *print format per column, according to FORTRAN-definition* Iw = Integer, w positions long Fw.d = Floating pointing number, w positions long with d decimal positions Ew.d = Idem, but with exponent
#SPECIMENVAR	=	1, [figure], m, [text], *depth in m –mv and sample code according to NEN 5104 of the pre-drilled soil*
#SPECIMENVAR	=	n , [figure], m, [text], *depth in m –mv and sample code according to NEN 5104 of the pre-drilled soil*
#STARTDATE	=	[number], [number], [number] *yyyy, mm, dd*
#STARTTIME	=	[number], [number], [number] *hh, mm, ss*
#TESTID	**=**	**[text] *number of penetration test***

#XYID	=	[number], [figure], [figure] , [figure] , [figure] *coordinate system, X, Y, delta X, delta Y*
		The coordinate systems are linked to a country code so that the first numbers 0000-0009 are defined free or international.
		00000 = own coordinate system *see #MEASUREMENTTEXT = 7*
		00001 = Geographic Coordinate System
		01000 = SPCS
		31000 = RD: coordinate system = Cartesian, date= RD1918, projection method = stereographic
		31001 = UTM-3N: coordinate system = Cartesian, date = ED50, projection method = Mercator, central meridian = 3°OL
		31002 = UTM-9N: coordinate system = Cartesian, date = ED50, projection method = Mercator, central meridian = 9°OL
		32000 = Belgian Bessel: coordinate system = geographic, date = BD72, projection method = Belgian Lambert
		49000 = Gauss-Krüger: coordinate system = Cartesian, date = Potsdam, projection method = Transversal Mercator

#ZID	=	[number], [figure] , [figure] *height system, Z, delta Z*
		The height systems are linked to a country code. 000-0009 are defined free or international
		00000 = own reference level *see #MEASUREMENTTEXT = 8*
		00001= Low Low Water Spring

		31000= NAP
		32000= Ostend Level
		49000= Normal Null
#EOH	=	

In order to facilitate self-description of the keywords, two special keywords have been defined:

#STRUCTURETEXT	=	[keyword], [text], [text], ... *semantic definition of the key word*
#STRUCTURETYPE	=	[keyword], [text], [text], .… *syntactic definition of the keyword.*

3.2 Reserved MEASUREMENTVAR's

The list in section 3.1 assigns a number of MEASUREMENTVAR variables. In a GEF-CPT-Report these variables can only be used for the allotted quantities. The table below provides an overview of the numbers reserved for MEASUREMENTVARs.

Sequential number	[Default] value	Unit	Quantity
1	[1000]	mm^2	nom. surface area cone tip
2	[1500]	mm^2	nom. surface area friction sleeve
3		-	net surface area quotient of cone tip
4		-	net surface area quotient of friction sleeve
5	[100]	mm	distance of cone to centre of friction sleeve
6	0= no 1= yes	-	friction present
7	0= no 1= yes	-	PPT u1 present
8	0= no 1= yes	-	PPT u2 present
9	0= no 1= yes	-	PPT u3 present
10	0= no 1= yes	-	inclination measurement present
11	0= no 1= yes	-	use of back-flow compensator
12	0= electronic penetration test 1= mechanical discontinue 2= mechanical continue	-	type of cone penetration test
13		m	pre-excavated depth
14		m	groundwater level
15		m	water depth (for offshore activities)
16		m	end depth of penetration test
17	0= end depth reached 1= max. penetration force 2= cone value 3= max. friction value	-	stop criteria

Sequential number	[Default] value	Unit	Quantity
	4= max. PPT value 5= max. inclination value 6= obstacle 7= danger of buckling		
18-19			for future use
20		MPa	zero measurement cone before penetration test
21		MPa	zero measurement cone after penetration test
22		MPa	zero measurement friction before penetration test
23		MPa	zero measurement friction after penetration test
24		MPa	zero measurement PPT u1 before penetration test
25		MPa	zero measurement PPT u1 after penetration test
26		MPa	zero measurement PPT u2 before penetration test
27		MPa	zero measurement PPT u2 after penetration test
28		MPa	zero measurement PPT u3 before penetration test
29		MPa	zero measurement PPT u3 after penetration test
30		degrees	zero measurement inclination before penetration test
31		degrees	zero measurement inclination after penetration test
32		degrees	zero measurement inclination NS before penetration test
33		degrees	zero measurement inclination NS after penetration test
34		degrees	zero measurement inclination EW before penetration test
35		degrees	zero measurement inclination EW after penetration test
36-40			for future use
41		km	mileage

3.3 Reserved MEASUREMENTTEXT's

The list in section 3.1 assigns a number of MEASUREMENTTEXT variables. These variables can only be used in a GEF-CPT-Report for the allotted quantities. The table below provides an overview of the numbers reserved for MEASUREMENTTEXT.

Sequential number	Quantity
1	client
2	name of the project
3	name of the location
4	cone type and serial number
5	Mass and geometry of probe apparatus, including anchoring
6	applied standard, including class
7	own coordinate system
8	own reference level
9	fixed horizontal level (usually: ground level or flow bed)
10	orientation direction biaxial inclination measurement (N-direction)
11	unusual circumstances
12-19	for future use
20	correction method for zero drift
21	method for processing interruptions
22	remarks
23	remarks
24-29	for future use
30	calculation formula or reference for column number...
31	calculation formula or reference for column number...
32	calculation formula or reference for column number...
33	calculation formula or reference for column number...
34	calculation formula or reference for column number...
35	calculation formula or reference for column number...
36-40	for future us
41	highway, railway or dike code

3.4 Definition of columns in the data block

The following table lists 20 quantities numbers. Each number is unequivocally linked to a physical quantity: the number identifies which physical quantity is listed in a specific column.

Measurement value or calculated value	Unit	Quantity number
penetration length	**m**	**1**
measure cone resistance q_c	**MPa**	**2**
friction resistance	MPa	3
friction number	%	4
pore pressure u_1	MPa	5
pore pressure u_2	MPa	6
pore pressure u_3	MPa	7
inclination (resultant)	degrees	8
inclination N-S	degrees	9
inclination E-W	degrees	10
corrected depth, measured below the fixed horizontal surface	m	11
time	s	12
corrected cone resistance q_t	MPa	13
net cone resistance q_n	MPa	14
pore ratio B_q	-	15
cone resistance number Nm		16
weight per unit volume, γ	kN/m^3	17
in-situ, initial pore pressure u_o	MPa	18
total vertical soil pressure, σ_{v0}	MPa	19
effective vertical soil pressure, σ'_{v0}	MPa	20

It should be noted that the penetration length (= the uncorrected depth of the cone below the fixed horizontal surface) is the first quantity in the data block. The directly measured values of the other quantities are values taken at <u>the same time</u> as the cone resistance is observed.

4 EXAMPLES OF PENETRATION TESTS IN GEF-CPT-REPORT FORMAT

4.1 The minimum report of a cone penetration test in GEF-CPT-Report

```
#GEFID              =   1,0,0
#PROCEDURECODE      =   CPT-Report, 1,0,0
#COMPANYID          =   CPT bv, Sondeerburg, 31
#PROJECTID          =   CT, 380730
#FILEDATE           =   1998,02,18
#TESTID             =   C2-265
#FILEOWNER          =   W.A. van Buuren
#COLUMN             =   2
#LASTSCAN           =   1251
#COLUMNINFO         =   1, m, penetration length, 1
#COLUMNINFO         =   2, MPa, Cone, 2
#EOH                =
-0.12  0.205
-0.14  0.199
-0.16  0.219
-0.18  0.252
-0.20  0.298
-0.22  0.338
-0.24  0.437
…..
…..
-24.80  21.828
-24.82  22.057
-24.84  21.929
-24.86  21.433
-24.88  19.936
-24.90  18.599
-24.92  17.926
-24.94  18.004
-24.96  18.567
-24.98  19.387
-25.00  19.64
-25.02  19.747
-25.04  20.261
-25.06  21.368
-25.08  23.121
```

4.2 Extensive cone penetration test in GEF-CPT-Report format

#GEFID	=	1,0,0
#COLUMN	=	11
#COLUMNINFO	=	1, m, penetration length, 1
#COLUMNINFO	=	2, MPa, Cone value, 2
#COLUMNINFO	=	3, MPa, Friction, 3
#COLUMNINFO	=	4, MPa, Pore pressure u2, 6
#COLUMNINFO	=	5, degrees, Inclination NS, 9
#COLUMNINFO	=	6, degrees, Inclination EW, 10
#COLUMNINFO	=	7, m, corrected depth, 11
#COLUMNINFO	=	8, MPa, corrected cone value, 12
#COLUMNINFO	=	9, %, friction number, 4
#COLUMNINFO	=	10, s, time, 12
#COLUMNINFO	=	11, MPa, excess pore pressure, 15
#COLUMNMINMAX	=	1, -57.68, -1.52
#COLUMNMINMAX	=	2, 0.33, 61.12
#COLUMNMINMAX	=	3, 0.000, 1.324
#COLUMNMINMAX	=	4, -0.064, 2.060
#COLUMNMINMAX	=	5, 0.02, 9.25
#COLUMNMINMAX	=	6, .04, 8.41
#COLUMNMINMAX	=	7, -57.52, -1.52
#COLUMNMINMAX	=	8, -0.522, 60.892
#COLUMNMINMAX	=	9, -0.164, 119.189
#COLUMNMINMAX	=	10, 1, 3995
#COLUMNMINMAX	=	11, -0.557, 1.511
#COLUMNSEPARATOR	=	:
#COLUMNTEXT	=	1, on
#COLUMNVOID	=	1, -99999
#COLUMNVOID	=	2, -99999
#COLUMNVOID	=	3, -99999
#COLUMNVOID	=	4, -99999
#COLUMNVOID	=	5, -99999
#COLUMNVOID	=	6, -99999
#COLUMNVOID	=	7, -99999
#COLUMNVOID	=	8, -99999
#COLUMNVOID	=	9, -99999
#COLUMNVOID	=	10, -99999
#COLUMNVOID	=	11, -99999
#COMPANYID	=	CPT bv, Sondeerburg, 31
#DATAFORMAT	=	ASCII
#FILEDATE	=	1998, 02, 29
#FILEOWNER	=	W.A. van Buuren
#LASTSCAN	=	2808
#MEASUREMENTTEXT	=	1, NS, client
#MEASUREMENTTEXT	=	2, HSL-line, project name
#MEASUREMENTTEXT	=	3, Franeker, name of location

#MEASUREMENTTEXT = 4, C5W1-007, cone type and serial number
#MEASUREMENTTEXT = 5, Ballast wagon 18; 25 tons: no anchoring
#MEASUREMENTTEXT = 6, NEN 5140 class 1, applied standard
#MEASUREMENTTEXT = 7, Measured to the N corner of the lighthouses, coordinate system
#MEASUREMENTTEXT = 8, Measured to the upper side of electric fencing pole, Reference level
#MEASUREMENTTEXT = 9, ground level, fixed horizontal .level
#MEASUREMENTTEXT = 10, lighthouses, N-direction inclinometer
#MEASUREMENTTEXT = 11, Torrential rain, Unusual circumstances
#MEASUREMENTTEXT = 20, Zero drift of the cone elements remains within the norm therefore no correction, Correction method
#MEASUREMENTTEXT = 21, No special processing used, method of processing interruptions
#MEASUREMENTTEXT = 22, At 12.25 m a thin gravel layer, remarks
#MEASUREMENTTEXT = 30, The corrected cone value is corrected according to “Soil Characterisation by In Situ Tests” 1998-01-05 sect 2.16.
#MEASUREMENTTEXT = 31, Excess pore pressure dU = U – U_0 where U_0 is the groundwater level at that place
#MEASUREMENTVAR = 1, 1000, mm^2, Nom. surface are cone tip
#MEASUREMENTVAR = 2, 1500, mm^2, Nom. surface area friction element
#MEASUREMENTVAR = 3, 0.62, -, Net surface area quotient of the cone tip
#MEASUREMENTVAR = 4, 1.0, -, Net surface area quotient of the friction casing
#MEASUREMENTVAR = 5, 100, distance between cone and centre of friction casing
#MEASUREMENTVAR = 6, 1, -, Friction present
#MEASUREMENTVAR = 7, 0, -, PPT u1 present
#MEASUREMENTVAR = 8, 1, -, PPT u2 present
#MEASUREMENTVAR = 9, 0, -, PPT u3 present
#MEASUREMENTVAR = 10, 1, -, inclination measurement present
#MEASUREMENTVAR = 11, 1, -, use of back-flow compensator
#MEASUREMENTVAR = 12, 0, electronic cone penetration test
#MEASUREMENTVAR = 13, 1.50, m, pre-excavated depth
#MEASUREMENTVAR = 14, 0.10, m, groundwater level
#MEASUREMENTVAR = 16, 57.68, m, End depth of probe
#MEASUREMENTVAR = 17, 6, -, Stop criteria: obstacle
#MEASUREMENTVAR = 20, 0.000, MPa, zero measurement before penetration test
#MEASUREMENTVAR = 21, -0.001, MPa, zero measurement after penetration test,
#MEASUREMENTVAR = 22, 0.0000, MPa, zero measurement friction before penetration test
#MEASUREMENTVAR = 23, 0.0002, MPa, zero measurement friction after penetration test
#MEASUREMENTVAR = 26, 0.0000, MPa, zero measurement PPT u2 before penetr. test
#MEASUREMENTVAR = 27, 0.0003, MPa, zero measurement PPT u2 after penetr. test
#MEASUREMENTVAR = 32, 0.01, gr, zero measurement inclination NS before penetr. test
#MEASUREMENTVAR = 33, -.02, gr, zero measurement inclination NS after penetr. test
#MEASUREMENTVAR = 34, 0.01, gr, zero measurement inclination EW before penetr. test
#MEASUREMENTVAR = 35, 0.03, gr, zero measurement inclination EW after penetr. test
#PROCEDURECODE = GEF-CPT-Report, 1,0,0

```
#PROJECTID           = CT, 380731, 02
#RECORDSEPARATOR     = !
#REPORTDATAFORMAT    = F7.2 F7.3 F8.4 F8.4 F5.1 F5.1 F9.2 F9.3 F6.2 I6 F7.2
#SPECIMENVAR         = 1, 0.00, m, 15 = pure peat
#SPECIMENVAR         = 2, 0.40, m, 10 = sand clayish
#SPECIMENVAR         = 3, 0.80, m, 20 = gravel silty
#SPECIMENVAR         = 4, 1.30, m, 15 = pure peat
#STARTDATE           = 1998, 02, 01
#STARTTIME           = 10, 03, 59
#STRUCTURETEXT       = COLUMNINFO, column number in data block, unit, quantity, quantity-number
#STRUCTURETYPE       = COLUMNINFO, integer, string, string, integer
#TESTID              = C2-366
#XYID                = 0, 12.345, 6.7890, 0, 0
#ZID                 = 0, 40.03, 0
#EOH                 =
-1.52:0.382:0.0127:-0.0021:0.2:0.4:-1.52:0.384:3.32:1:-0.02:!
-1.54:0.382:0.0137:0.0045:0.2:0.4:-1.54:0.379:3.59:2:-0.01:!
-1.56:0.375:0.0151:0.0126:0.3:0.3:-1.56:0.366:4.03:3:0.00:!
-1.58:0.375:0.0164:0.0191:0.2:0.4:-1.58:0.361:4.37:4:0.00:example commentary text 1!
-1.60:0.414:0.0169:0.0051:0.3:0.4:-1.59:0.410:4.08:5:-0.01:!
.........
-57.54:21.648:0.3706:0.0407:8.6:5.2:-57.39:21.619:1.71:3987:-0.53:!
-57.56:18.683:0.369:0.1252:8.6:5.1:-57.41:18.593:1.98:3988:-0.45:!
-57.58:16.735:0.3643:0.2507:8.6:5.2:-57.43:16.554:2.18:3989:-0.33:!
-57.60:15.499:0.3722:0.4095:8.5:5.2:-57.45:15.204:2.40:3990:-0.17:example commentary text 2!
-57.62:16.239:0.386:0.5281:8.5:5.2:-57.47:15.859:2.38:4438:-0.05:!
-57.64:18.495:0.3914:0.5549:8.5:5.1:-57.48:18.095:2.12:4440:-0.02:!
```

5 WHAT IS VERIFIED BY *GEF VERIFICATION*

When the program ***GEF Verification*** is started an empty screen is shown. The program is operated by using the menu options or optionally by using the toolbar of which the buttons represent some of the common functions in the menus.

File→Open

This opens a GEF-file. A file selection box allows the user to browse directories for GEF-files. By default only files with the extension .GEF are shown, but this can be changed in the ‘show files of type’ box. To select a batch (=multiple files) the shift and control buttons must be used in combination with the mouse. After opening the verification is automatically performed and the verification log is shown on the screen.

File→Close

This closes GEF-files that have been opened. The screen will appear empty again.

File→Save logfile as…

The verification log can be saved in text format as a file with a filename that can be chosen by the user. Use the .TXT extension if you would like the file to open with Notepad.

File→Convert NENGEO Files…

This converts NENGEO-files to GEF-files. The file open box automatically lists files with a .SON extension. As in File:Open it is possible to change this extension and in the same way it is possible to select multiple NENGEO-files. After selection the files are automatically converted to GEF-files and saved in the same directory with the same name but with the extension .GEF instead of .SON. It is not possible to convert NENGEO files which contain multiple sessions. If a corrupt NENGEO file or a NENGEO file with multiple sessions is encountered the user is informed that the file can not be converted.

File→Exit

This exits ***GEF Verification***

View→Verification

The screen can only display one of the three views at a time:
Verification, File or Curve.
The verification view shows the log file with the verification results.

View→File

This option can only be selected if a single file is opened. The content of the GEF-file is shown in text format.

View→Curve

This option can only be selected if a single file is opened and the errors found in the verification are not related to errors in the data block. It shows graphs of the 4 most important quantities (if present) in a GEF-CPT-Report file as a function of depth.

View→Toolbar

Toggles between showing the toolbar or not.

View→Statusbar

Toggles between showing the statusbar or not.

Options

Toggle between:

- Only verifying the header of the GEF-file
- Verifying both the header and the data block of the GEF-file.

About

Displays the about box

5.1 Verification of GEF

A CPT-file in GEF-CPT-Report format consists of a header and a data block section.
GEF Verification verifies the following:

#GEFID in first line

If the first line does not contain the #GEFID keyword no further checks will be done. The file is considered not to be a GEF-file

Keyword identification

The header sequentially searches for keywords which must always start with a '#'-sign. The end of the keyword is indicated by the '='-sign. Between the keyword and the '='-sign, space is allowed.
An error is reported if after 1024 characters a '#'-sign has not been found.
An error is reported if a '#' is found, but after 1024 characters a'='-sign misses.
An error is reported if a keyword is unknown.

Checking the number of parameters

After the '='-sign a keyword has a certain number of parameters separated by comma's. Some parameters are obligatory, others are optional. An error is reported if the number of parameters is smaller than the minimum number of obligatory parameters or larger than the maximum number of parameters possible for the specific keyword.

Type-checking the parameters

An error is reported if a parameter has the wrong type. For example if a certain parameter is a string(text), but an integer(number) is expected, this will be reported

Repeated keywords

Some keywords can only be used once. Others can be used more often, but only once, in combination with a certain parameter (for example with a certain column). ***GEF Verification*** reports an error if keywords are encountered which have already been used or have already been used in combination with a certain parameter.

Warning for wrong version

GEF Verification 1.0 verifies GEF-CPT-Report 1.0.0. If a GEF-file has a different version the software reports a warning.

Obligatory keywords

The following keywords are obligatory keywords in GEF-CPT-Report:

- #GEFID
- #COLUMN
- #COLUMNINFO for each column
- #COMPANYID
- #FILEDATE
- #FILEOWNER
- #LASTSCAN
- #PROCEDURECODE
- #PROJECTID
- #TESTID
- #EOH.

If any of these keywords was not found after searching the entire file, an error is reported.
If the #COLUMN or #LASTSCAN keyword are, missing ***GEF Verification*** does not have enough information to start reading the data block. Consequently, the curve cannot be displayed.

Check the parameters of #COLUMNINFO

The keyword #COLUMNINFO assigns a certain parameter like cone resistance, friction ratio, pore water pressure, corrected depth, slope etc. to a specific column number in the GEF-file. Each parameter has its own unambiguous identification number.

An error is reported if a parameter is assigned to two different columns, since this would create an ambiguous situation when locating the correct column for a specific parameter.
An error is reported if parameter 1 ‘Length’ or parameter 2 ‘Cone Resistance’ have not been assigned to a column. The definition of the GEF-CPT-Report file format says these parameters *must* be present in the file.

Check data block

In the header it is optionally possible to specify a column separator and a record separator. It is also possible to allow the use of extra text in the data block if the #COLUMNTEXT keyword is used. ***GEF Verification*** reads the data block and *stops* if an error occurs. If this error occurs, the type of error is reported. If an error occurs when reading the data block, it is not possible to view any curves.

Minimum and maximum values of the columns

If the data block has been read, it is possible to calculate the minimum and maximum value of each column. These values can be compared to the minimum and maximum values stated by the (optional) keyword #COLUMNMINMAX in the headerblock and tested for. If the comparison gives different values, an error is reported.

APPENDIX 1:

GLOSSARY FOR STANDARDISING DIGITAL DATA OF CONE PENETRATION TESTS

The table below briefly describes all the concepts used.

Concept	Description
[integer]	integer number, no decimals allowed
[character]	a number, character or punctuation mark
[number]	integer number, no decimals allowed
[text]	numbers, characters, punctuation marks
apparatus	probe apparatus and data acquisition system to perform cone penetration test (pushing, measuring, registering and storing)
ASCII	American Standard Code Information Interchange
back-flow compensator	facility to compensate the probe length for upwards movements of the probe housing
calculated columns	column data stored in GEF and calculated externally, based on known calculation formulae where appropriate
calculation formula	formula for calculating new quantities, such as friction figure is the quotient of skin friction and cone resistance, using (different) measurement data possibly supplemented with extra parameters
calculation procedures	software component which converts measured data into calculated data
character	single readable character
class of the penetration	one of the three NEN 5140-defined (quality) classes test in accordance with NEN 5140
code words	words defined in GEF, also described as a string name
computer system	such as DOS, Unix, Apple
cone penetration test	measurement of the mechanical soil resistance using a cone over the specified depth of traject
Cone Penetration Test Report format	CPT-Report, using GEF
cone resistance q_c	the resistance needed during the penetration test to move the cone tip
cone tip	part of the penetration cone which defines the tip resistance q_c
column	collection of data defined using the same fixed position on a line of each individual registration with the depth/time
column information	definition of the content and description of a data column in the file
column number	the number of a column
compulsory measurement values	length and cone resistance (minimum content)
coordinate system	definition used for x,y, and sometimes z coordinates, such as the Geographical coordinate system: longitude and latitude
corrected cone resistance q_t	the cone resistance corrected for the difference in pore water pressure below and above the cone tip
corrected depth	depth of the penetration cone, measured beneath a fixed horizontal reference level, taking into account the deviation of the casing with respect to the vertical
country code	code of the country for defining the reference height level
CR/LF	control characters which indicate the end of a line
data block	part of the file where the measured and calculated data are stored in

Concept	Description
	accordance with the definition specified by the header block
data, calculated	data calculated using the known calculation formulae
data descriptive	data describing its own contents
data, fixed	static data
data format	definition of the data storage
date of penetration test	day on which a CPT was carried out, given as dd-mm-yyyy
dd-mm-yyyy	description of the day of the month with two characters, the month with two characters and the year with four characters
default value	value which applies if no other value is input
depth of first observation	depth below the selected horizontal (fixed) reference level used to register the data which can be considered as in-situ measurements (for example pre-test, -excavating or -drilling)
digital	in symbols, coded, used to store documents via computers
dimension	see unit
dissipation test	specifying the change in pore water pressure in the penetration cone in the time at a constant depth after the penetration cone is stationary
effective vertical soil $\sigma'_{v;0}$ stress	calculated effective vertical granular stress, being the total wet weight per surface area of the above-lying soil minus the prevailing pore water pressure
executing company	name of company performing the cone penetration test
excess pore pressure ratio DPPR	ratio number between measured excess pressure and the cone resistance q_c :DPPR = $(u - u_o)/q_c$
figure	decimal number
file	digital file
file descriptive	definition of a digital file including its definitions used for storage, such as definition of the foundation for the description (GEF version number, for example)
file tracing	possibility to trace who created the file
final depth	depth at which the cone penetration test stops
fixed position	defined place in GEF
format	the only word which in principle cannot be used in this project, in combination with other words intended as (agreed) structure of...
friction figure R_f	quotient of local friction and cone resistance
GEF definition	definition of the GEF format in written form
Geotechnical Exchange Format	agreed name of the exchange format
inclination (resultant not direction-oriented)	deviation from the vertical in degrees
inclination EW	deviation from the vertical in EW direction in degrees
inclination NS	deviation from the vertical in NS direction in degrees
groundwater level	level of the groundwater beneath the ground level
header block	part of the file which describes the stored data

height of the selected horizontal reference level (= usually ground level) beneath fixed reference level	height
identification number of penetration test	number of the cone penetration test for internal processing
in-situ, initial water pressure u_o	prevailing static pore pressure in the soil
interruptions	temporary interruption of one or more components of the penetration process
keywords	not used, see code words
scan	a collection of measurement values for one depth
maximum value	maximum registered measurement value of a parameter during a cone penetration test
measured cone resistance q_c	the resistance necessary to move the cone tip downwards during the cone penetration test
measurement elements	the part of the apparatus used to register the measurement values (e.g. the stretching gauge for the cone tip and not the probe rod)
measurement data	collection of measurement values
measurement procedures	pre-written series of actions to collect measurement data
measurement scan	in principle not used (see measurement data)
measurement texts	in principle not used (description uses code words, column information and columns)
measurement variables	in principle not used (description uses code words, column information and columns)
measurement value	a measured value
measurement	the collection of measured values (such as cone resistance)
minimum values	minimum registered measurement value during a cone penetration test
mm	description of the month of the year with two characters
NEN 3680	Dutch Standard for performing cone penetration tests (only valid for mechanical penetration tests)
NEN 5140	Dutch Standard for performing electronic cone penetration tests
net cone resistance q_n	cone resistance corrected for pore pressure and the effective vertical soil stress
net surface area quotient	ratio of the gross surface area to the net surface area above the cone tip in relation to the action of pore pressure
number	an integer number
original information sources	information about who created the file
order number	order number defined by the user processing the file (this can therefore change during the life-time of the file)
parameter	in principle not used, see quantity
remarks	circumstances worth mentioning during measurement which may influence the interpretation of the results, such as a power failure
point of time when penetration test starts	hour, minute, seconds since start of the cone penetration test or recording the zero measurements
pore pressure u_1	measured pore pressure in the cone tip
pore pressure u_2	measured pore pressure directly behind the cone tip

pore pressure u_3	measured pore pressure directly above the friction casing
pore pressure ratio B_q	ratio number between measured excess pore pressure and the cone resistance q_c corrected for the total soil pressure $B_q = (u-u_o)/(q_c-\bullet_{vo})$
print format	format of printed values
penetration cone	instrument used to measure the mechanical soil resistance
penetration data	data collected using the penetration cone
penetration length	distance traversed by the penetration cone in the soil
penetration location	location of the starting point of the cone penetration test
penetration report	report of the measurement results
procedure descriptive	providing a description of the procedure for the evaluation of data
procedures	see measurement procedures
processing	registration/display method (in particular, interruptions)
project information	description of project information which is relevant to file management and reporting
quantities	name of the data in one column
quantity number	number representing the type of data in one column
reference level	fixed level, for example NAP but not curbstone
release	version of the GEF definition
release number	number of the version of the GEF definition
report	form of data transfer to the client
results	in principle not used, see calculated data
results, deduced	in principle not used, see calculated data. The distinction between deduced and 'ordinary' results is formed by defining the relationship between measured and calculated data; empirical formulae are used for the deduced results.
separating character	defined character which indicates the division between two columns
friction	friction along the friction casing
friction casing	cylindrical part of the probe cone, directly above the cone tip, which measures the friction along the cone casing
standard	definition agreed and confirmed by standardisation institute for actions, qualitative or procedural
stop criteria	defined type of exceeding of one or more of the requirements of the cone penetration test
structure	defined structure
structure information	definition of the structure
text	readable text of 256 characters maximum, without commas or #s
text field	string
time	time passed since the start of the cone penetration test or recording the zero measurements
total vertical soil pressure, σ'_{v0}	calculated total wet weight per unit area of the soil above
type of penetration test	cone (resistance) possibly supplemented by local friction
unit	for example MPa, m or %
unusual circumstances	circumstances worth mentioning during the cone penetration test which may influence the interpretation of the result, such as an exceptional groundwater level
weight per unit volume, γ	weight of wet soil in weight per unit volume
water depth	depth of the bottom of the water with respect to the water surface

PPT filter (u_1, u_2, or u_3)	filter that only allows water to pass (or any other liquid) and protects the pore pressure transducer against contact with soil grains
X and Y coordinates	horizontal coordinates
yyyy	description of the year using four characters
Z coordinate	vertical coordinate
zero measurement	read-out of the constant value of a measuring instrument, without the instrument being affected
zero drift	change in the zero value during or after a test
zero values	measured value during the zero measurement

APPENDIX 2:

DESCRIPTION OF GEF LANGUAGE DEFINITION

2.1 Introduction

The objective of the geotechnical exchange format is to store measurement results in such a way that they can be analysed at a later time. Two types of information are needed when a measurement has been carried out and the measurement results are available: firstly, information about the circumstances under which the measurement was performed and how the measurement results are stored, and secondly, the measurement results themselves. This means that in addition to the measurement results themselves, mention must also be made of how a measurement is stored and what the figures in the file represent. As regards organisation, this includes the way in which the measurement is stored (such as binary), the number of columns, and the number of scans. For the interpretation, this can be, for instance, which quantity is listed in a column and in which units. For this purpose, the actual measurement results - the data - are preceded by a header which describes in a readable form (ASCII) how the measurement is composed.

The data are divided into columns and scans. One measured quantity is listed per column, for example during a specified period in the time or during a load. A scan is a snap-shot recording, which comprises a variety of registrations from measurement instruments, for example at a fixed point in time or load. The header has its own organisation. Information is profiled using fixed code words. A code word is recognised unequivocally. The sequence of most code words is not important. A code word is followed by information. How the information must be interpreted depends on the code word. The length of the header is variable. As the header is specified in ASCII, it is always readable, via an editor or a simple viewer, regardless of how the data is stored.

The geotechnical exchange format is a type of control language with words. Code words can be added, if required. The addition of words has consequences for the software which can import a GEF file: the ability to recognise new words must be added to the software. The information belonging to each code word has a fixed structure. If a program recognises a code word then it can import the information belonging to that code word, as the structure of a code word’s information is defined unequivocally.

For some years, GeoDelft has been using the geotechnical exchange format GEF for all its measurements. The formal description of GEF is laid down in a GeoDelft document, written by dr. H. den Adel and drs.P. Schaminée. This section originates from the document and has been revised to produce a first formal description of GEF.

2.2 Code words

A file in accordance with the geotechnical exchange format can be divided into two parts: the header and the measurement. The header describes how the registered data is organised in directly readable text. Code words are used for this purpose. A code word is preceded by a ‘#’ character. A ‘=’ character follows the code word. The code word is therefore between the ‘#’ and the ‘=’. Several examples of correct and incorrect code words are given in Table 4.1

Table 4.1 Examples of correct and incorrect code words

Code word	Evaluation
#ANALYSISCODE =	Correct
#ANALysisCODE =	Correct
# ANALYSISCODE =	Correct
#ANALYSIS CODE =	Incorrect, space between the words

Several conditions exist for code words and their information. A code word with accompanying information must always be on one line: no ‘end of line’ LF or CRLF may appear in the information. If the information following the ‘=’ characters consists of several parts, it is separated by commas. The end of the information is either specified by an end of line, or by the start of a new code word with a hash (#). A comma and a hash are therefore special characters. If these characters appear in an information field after a code word, it must be indicated that the literal meaning of the character is now required. The backslash is used for this purpose: \# or \,. This also makes the backslash a special character so that if a literal backslash is required in the text, this must also be preceded by a backslash.

There are currently 54 code words, but the flexibility of the geotechnical exchange format is such that new code words can be created if required. Expansion nonetheless has consequences for the applications capable of decoding the code words. These must be adjusted. Text and/or numbers must be placed after every code word, apart from the code word which indicates the end of the measurement description (#EOH=). Which information and how much depends on the code word itself.

Several conditions must be met if the measurement is to be meaningful: a minimum description of how the file is organised must be known. This leads to a division of code words into compulsory, non compulsory and conditionally compulsory code words. The last category consists of code words which are in fact compulsory, but for which a default value has been filled in. If the word is therefore not listed, the standard value is used. If the default value is not suitable as a result of another code word, the word must still be given. It is therefore compulsory under certain conditions. The compulsory code words are given in Table 4.2.

Table 4.2 Compulsory code words

Code word	Code word
#GEFID =	#COLUMN =
#FILEDATE =	#COLUMNINFO =
#PROJECTID =	#EOH =
#FILEOWNER =	

Table 4.2 forms the minimum subset of code words. A file which complies with the geotechnical exchange format consists of all these code words at least. If one or more of the compulsory code words are missing then the application should stop automatically processing the data. The measurement is incomplete. Optionally, the application can query the user so that the missing information can be found. The remaining code words are given in Table 4.3.
The code words are subdivided into four logical categories:

- File tracing
- File descriptive
- Data descriptive
- Procedure descriptive.

The first and second categories ensure the direct usefulness of the file. The third and fourth category state how the numbers in the header should be interpreted, what they represent and how (via which procedures) the data in the file was obtained.

File tracing code words allow the user to find out who created the file. It traces back to the organisation which set up the file and to the activities in the framework within which they were collected. This deals with readable data which can be interpreted without any knowledge about the type of measurement.

File descriptive code words give information about the file itself, how the file is organised, how it should be read, and the structure of the file components. This is also concerned with readable data which can be interpreted without any knowledge about the file.

Data descriptive code words state how the data is stored, what the numbers mean, and in which unit a quantity is listed.

Procedure descriptive code words state which procedures have been followed, such as how the measuring, sampling, analysing, archiving and reporting activities have been carried out, which additional information was received or was required, and which calculation method was used. Interpretation of the data belonging to these code words is only possible if the procedure used for the measurement or analysis is known.

In addition to these categories, a division can also be made into compulsory and non compulsory. The compulsory code words are give in Table 4.2. These are code words form the file tracing and file descriptive categories. The compulsory code words are the minimum requirement concerning the occurrence of code words with which a GEF file must comply. There are also non compulsory code words or conditionally compulsory code words.

Table 4.3 Non compulsory and conditionally compulsory code words

Codewoord	Codewoord	Codewoord
#ANALYSISCODE=	#COLUMNOFFSET=	#PROJECTNAME=
#ANALYSISTEXT=	#COLUMNSEPARATOR=	#REPORTCODE=
#ANALYSISVAR=	#COLUMNPOWERSUPPLY=	#REPORTTEXT=
#FILINGCODE=	#LASTSCAN=	#REPORTVAR=
#FILINGTEXT=	#MEASUREMENTCODE=	#RECORDSEPARATOR=
#FILINGVAR=	#MEASUREMENTTEXT=	#SCANFREQ=
#COMPANYID=	#MEASUREMENTVAR=	#SCANTIME=
#COMMENT=	#SPECIMENCODE=	#STARTDATE=
#DATAFORMAT=	#SPECIMENTEXT=	#STARTTIME=
#DATATYPE=	#SPECIMENVAR=	#STRUCTURETEXT=
#FIRSTSCAN=	#OBJECTID=	#STRUCTURETYPE=
#COLUMNAMPLIFIER=	#EQUIPMENT=	#OS=
#COLUMNVOID=	#PROCEDURECODE=	#LANGUAGE=
#COLUMNMINMAX=	#TESTID=	#TIMECOLUMN=
#COLUMNTEXT=	#XYID=	#
#REPORTDATAFORMAT=	#ZID=	

Conditionally compulsory code words are sometimes compulsory, depending on the occurrence of another code word. If, for example, binary data are recorded, not only must the fact that binary data are recorded be stated, but also the size in bytes of each measurement and in which sequence the bytes are written. This information is not necessary for ASCII values.

Parts of the information following code words which can be omitted are given between [] in the typescript. Abbreviations are used to characterise the information, as given in Table 4.4. 4Value represents a value of 4 bytes in size, sUnit states a string which describes the unit.

2.3 File tracing

2.3.1 #COMPANYID

#Companyid=sName,sNumber,iCountrycode

sName	Name of the company.
sNumber	A number by which the company can be identified in the country. For example, a VAT number in The Netherlands.
iCountrycode	Code for the country. International telephone codes are used. For The Netherlands: 31.

Example: #Companyid=GEODELFT, 8000.97.476.B.01,31.

2.3.2 #FILEOWNER

#Fileowner=sFileowner

sFileowner	Text stating the owner of the file. The file owner is responsible for the quality of the file. He/she is usually the same person who performed the measurement.
Example:	#Fileowner=Lambert Smidt.

2.3.3 #FILEDATE

#Filedate=iYyyy,iMm,iDd

iYyyy	Year, when the file was created.
iMm	Month, when the file was created.
iDd	Day, when the file was created.
Example:	#Filedate=1995,02,20.

Note. The default value is the current date.

2.3.4 #PROJECTID

#Projectid=sType,[sNumber,[sSub]]

sType	Order identification, Within GeoDelft, the type of order is given as: CO/SE/BO/SO.
sNumber	The order number. Within GeoDelft a 6-figure number.
sSub	The sub-project number. Within GeoDelft a 3-figure number.
Example:	#Projectid=CO, 342770, 624.

2.3.5 #PROJECTNAME

#Projectname=sProject

sProject	Text stating the relevant project. Maximum of 255 characters.
Example:	#Projectname= High Speed Line.

2.4 Data descriptive

2.4.1 #DATATYPE

#Datatype=sType

sType Text stating which type of numbers are stored.

Example: #Datatype=REAL8.

Note. The types currently recognised as legal data types are described in Table 4.4. All variables are of the same type. This code word is conditionally compulsory for binary storage (#DATAFORMAT=BINARY). The default value is a 4-byte floating point number.

Table 4.4 Types of variables used

Type	Description	Abbreviation	Number of bytes
byte	flag	b	1
int2	integer number	i	2
int4	integer number	1	4
real4	decimal number	4	4
real8	decimal number	8	8
string	text	s	variable

2.4.2 #DATAFORMAT

#Dataformat=sType

sType Text stating whether the measurement is stored as an ASCII or BINARY file.

Example: #Dataformat=Binary.

Note. Only ASCII and BINARY are permitted. Binary data storage can be imported and processed extremely rapidly. It is, however, system dependent. The data type default value is ASCII. Also see code word OS.

2.4.3 #FIRSTSCAN

#Firstscan=1First

1First The number of the first scan containing a meaningful value.

Example: #Firstscan=13.

2.4.4 #COLUMN

#Column=iColumn

iColumn	A number between 1 and 250. This gives the number of columns which are in the data block.
Example:	#Column=4.

Note. The data following the header must be written in #Column in the case of an ASCII file. Preferentially after the #Column columns follows a Line Feed (Unix) or a Carriage Return and a Line Feed (DOS). The default value is 0. For a binary file, it is assumed that #Column observations, consisting of 4-bytes floating point numbers, belonging to a scan, are stored. Also see Datatype.

2.4.5 #COLUMNAMPLIFIER

#Columnamplifier=iColumn,4Zero,4Amplifier[,1First[,1Last[, sCal]]]

iColumn	Number of the column to which the information relates. Maximum 250.
4Zero	The zero value of the transducer.
4Amplifier	The amplification factor for the transducer.
1First	The first scan from which the setting is applicable.
1Last	The last scan up to where the setting is applicable.
sIJk	The unit of the calibration factor.
Example:	#Columnamplifier=4, 0.26, 5.0, 123, 187, kPa/V.

Note. All figures from the relevant column are multiplied using the 4Amplifier and 4Zero is added: new=4Zero + 4Amplifier*old. Once the data are stored in a file, this code word disappears. The default value of lFirst is 1, and for 1Last is 2^{31}.

2.4.6 #COLUMNVOID

#ColumnVoid=iColumn, 8Value

iColumn	The number of the column to which the information relates. Maximum 250.
8Value	The figure which shows that the value does not contain suitable information.

Note. Instead of using a letter of a word, it was decided to define a figure to indicate that no suitable information is present at the position, so the possibility of using binary data blocks was not blocked.

2.4.7 #COLUMNINFO

#Columninfo =iColumn, sUnit, sQuantity[,iQnumber]

iColumn	Number of the column to which the information relates. Maximum 250.
sUnit	The unit in which the measurements are stored.
sQuantity	The quantity stored in the column.
iQNumber	The number identifying a quantity.

Example: #Columninfo=3, kPa, deviator pressure.

2.4.8 #COLUMNMINMAX

#Columnminmax=iColumn, 8Min, 8Max

iColumn	The column to which the information relates. Maximum 250.
8Min	Minimum value in the column.
8Max	Maximum value in the column.

Example: #Columnminmax=5, 0.17, 5.68.

2.4.9 #COLUMNOFFSET

#Columnoffset=iColumn, 1Scan[, 8Value]

iColumn	The column to which it relates. Maximum 250.
1Scan	The scan to which it relates.
8Value	The value which is ascribed to scan 1Scan in column iColumn.

Example: #Columnoffset=5, 135691, 1.0.

Note. Columnoffset is used to ascribe the physical value 8Value to scan 1Scan. This codeword is devised especially for relative measurements such as a displacement transducer. E.g. the codeword ascribes a value when the transducer makes contact with the object to be measured in scan 1Scan. In the example, this is scan 135691 in column 5 which has received the value of 1.0 The default value of 8Value is 0.0.

2.4.10 #COLUMNSEPARATOR

#Columnseparator=cChar

cChar	This is a character (to be printed in ASCII) which will be printed between each column, making it easier to import data with so-called spreadsheet programs or databases.

Example: #Columnseparator=,

2.4.11 #COLUMNPOWERSUPPLY

#Columnpowersupply=iColumn, 4Offset, 4Amplifier, iVcolumn[, sUnit]

iColumn	Number of the column with registrations which must still be converted Maximum 250.
4Value	Column offset.
4Amplifier	Amplification factor for the column.
iVcolumn	The column which states the supply voltage for the transducer mentioned in column iColumn.
sUnit	Unit of the calibration factor.

Example: #Columnpowersupply=3, 0.17, 5.0, 14, kPa/mV.

Note. This code word only occurs in an original measurement. It is used if the voltage given off by a transducer is dependent on the supply voltage to the transducers.

2.4.12 #COLUMNTEXT

#Columntext=iState

iState	A logical value, determining whether columntext do appear or not. 0 = no columntext. 1 = columntext will occur.

Example: #Columntext=1.

2.4.13 #LASTSCAN

#Lastscan=1Last

1Last	Number of the last meaningful scan.

Example: #Lastscan=888750.

Note. The default value is 2^{31}.

2.4.14 #RECORDSEPARATOR

#Recordseparator=cChar

cChar	The character used to finish a data record. A data record always consists of a fixed number of registrations (the value is given by the #column). It is the physical end of the scan. Also see ColumnVoid.

Example: #Recordseparator=;

2.4.15 #SCANFREQ

#Scanfreq=4Value, 1Start

4Value	The frequency used for measuring.
1Start	The number of the scan from where this frequency starts.

Example: #Scanfreq=27.0, 151745.

Note. The scan frequency is expressed in Hz (s^{-1}). The scan frequency is converted into a scan time in the GEF library.

2.4.16 #SCANTIME

#Scantime=4Value, 1Start

4Value	The time interval between two measurements.
1Start	The number of the scan from where this time interval starts.

Example: #Scantime=60.0, 151745.

Note. The time interval is expressed in s. In the example, a measurement is carried out each minute. For scanfreq and scantime, startdate and starttime are to be used. The measurement is thus placed in absolute form in time.

2.4.17 #STARTDATE

#Startdate=iYear, iMonth, iDay

iYear	The year in which the measurement started.
iMonth	The month in which the measurement started.
iDay	The day on which the measurement started.

Example: #Startdate=1995, 02, 06.

Note. Together with the start time, the start date is important for measurements as a functional of time.

2.4.18 #STARTTIME

#Starttime=iHour, iMin, 4Sec

iHour	The hour in which the measurement started.
iMin	The minute in which the measurement started.
4Sec	The second in which the measurement started.

Example: #Starttime=12, 35,24.75.

2.4.19 #TIMECOLUMN

#Timecolumn=iColumn[, iCode[, sUnit]]

iColumn	Number of the column giving the true measurement time.
iCode	A code for the unit which expresses time.
sUnit	Text stating the unit.

Example: #TIMECOLUMN=5,2.

Note. To calculate the time, the column which gives the time in respect of the starttime and startdate in sUnit has preference over a reconstructed time from scanfrequency and scantime. The default value for iCode is 1, namely seconds. The values are given in Table 4.5.

Table 4.5 Value of iCode and sUnit

iCode	sUnit
1	second
2	minute
3	hour
4	day
5	week

2.5 Procedure descriptive

2.5.1 #PROCEDURECODE

#Procedurecode=sCode, iRelease, iVersion, iUpdate[, sIsoref]

sCode	Text stating the code of the procedure.
iRelease	A number which gives the release of the procedure.
iVersion	A number which gives the version of the procedure.
iUpdate	A number which gives the update of the procedure.
sIsoref	Reference of ISO-9000 standard. Maximum 80 characters.

Example: #Procedurecode=GEF-CPT-Report, 1,0,0.

2.5.2 Analysis descriptive

2.5.2.1 #ANALYSISCODE

#Analysiscode=sCode, iRelease, iVersion, iUpdate[, sIsoref]

sCode	Text stating how the measurement data must be analysed.
iRelease	A number giving which release of the analysis program must be used to analyse the measurements.
iVersion	A number giving the version of the analysis program which must be used to analyse the measurements.

iUpdate	A number giving which update of the analysis program must be used to analyse the measurements.
sIsoref	Reference of ISO-9000 standard with which the analysis complies.

Example: #Analysiscode=MSAPRO, 2, 0, 5, ISO-01.

Note. Via this code word, a program can determine whether the data is suitable for analysis using this version of the program. This prevents a program designed to work out, for example, a dry critical density test from importing data from a compression test. In addition, measurements which still need to be analysed using an older version, can be processed using the most suitable method.

2.5.2.2 #ANALYSISTEXT

#Analysistext=1Number, sText[, sInformation

1Number	The number of the text variable. Maximum 1500.
sText	The text belonging to an analysis. Maximum 255 characters.
sInformation	Information about the text. Maximum 80 characters.

Example: #Analysistext=129, Cohesion calculated in accordance with NEN, comments.

2.5.2.3 #ANALYSISVAR

#Analysisvar=1Number, 4Value, sUnit, sQuantity

1Number	Number of the variable. Maximum 1500.
4Value	Value of the variable.
sUnit	Unit of the variable. Maximum 40 characters.
sQuantity	The quantity which the variable represents. Maximum 80 characters.

Example: #Analysisvar=129, 1.33,-,over-relax factor for creep.

2.5.3 Archive descriptive

2.5.3.1 #FILINGCODE

#Filingcode=sCode, iRelease, iVersion, iUpdate[, sIsoref]

sCode	Text which states how the measurement data must be archived.
iRelease	A number giving the release of the archive manager or database program which must be used to analyse the measurements.
iVersion	A number giving which version of the archive manager or database program must be used to archive the measurements.
iUpdate	A number giving which update of the archive manager or database program must be used to archive the measurement.
sIsoref	Reference of the ISO-9000 standard with which the archive management complies.

Example:	#Filingcode=B17, 3, 0, 1, NEN888888.

2.5.3.2 #FILINGTEXT

#Filingtext=1Number, sText[, sInformation]

1Number	The number of the text variable. Maximum 1500.
sText	The text belonging to an archive. Maximum 255 characters.
sInformation	Information about the text. Maximum 80 characters.
Example:	#Filingtext=129, Archiving in Oracle table PROJECT, tablename.

2.5.3.3 #FILINGVAR

#Filingvar=1Number, 4Value, sUnit, sQuantity

1Number	Number of the variable. Maximum 1500.
4Value	Value of the variable.
sUnit	Unit of the variable. Maximum 40 characters.
sQuantity	The quantity which the variable represents. Maximum 80 characters.
Example:	#Filingvar=133, 0.45, -,Location of the map.

2.5.3.4 #OBJECTID

#Objectid=1Value

1Value	Reference to a number in the archive where this GEF file belongs.
Example:	#OBJECTID=349123.

2.5.4 Measuring descriptive

2.5.4.1 #XYID

#XYid=iMap, 8X, 8Y[, 8epsX, 8epsY]

iMap	A number giving the type of coordinate system used.
8X	The x coordinate, measured in an East-West direction.
8Y	The y coordinate, measured in a North-South direction.
8epsX	Measurement accuracy of the X coordinate.
8epsY	Measurement accuracy of the Y coordinate.
Example:	#XYid=31000, 155015, 463701, 0.1, 0.1.

2.5.4.2 #ZID

#Locationid=iMap, 8Z[, 8epsZ]

iMap	A number giving the type of coordinate system used.

8Z	The z coordinate, the height or depth.
8epsZ	Measurement accuracy of the Z coordinate.

Example: #Zid=31001, -15.75, 0.001.

The default value of 8Z is 0.0.

2.5.4.3 #MEASUREMENTCODE

#Measurementcode=sCode, iRelease, iVersion, iUpdate[, sIsoref]

sCode	Text stating how the data were collected.
iRelease	A number giving the release of the test procedure.
iVersion	A number giving the version of the test procedure.
iUpdate	A number giving the update of the test procedure.
sIsoref	Reference of the ISO-9000 standard with which the test procedure complies.

Example: #Measurementcode=Cu, 3, 1, 0, 300-1-CU.

2.5.4.4 #MEASUREMENTTEXT

#Measurementtext=1Number, sText[, sInformation]

1Number	The number of the text variable. Maximum 1500.
sText	The text belonging to the test implementation. Maximum 255 characters.
sInformation	Information about the text. Maximum 80 characters.

Example: #Measurementtext=129, Temperature measured in accordance with NEN, temp. measurement.

2.5.4.5 #MEASUREMENTVAR

#Measurementvar=1Number, 4Value, sUnit, sQuantity

1Number	Sequence number of the variable. Maximum 1500.
4Value	Value of the variable.
sUnit	Unit in which the variable is expressed. Maximum 40 characters.
sQuantity	The quantity which the variable represents. Maximum 80 characters.

Example: #Measurementvar=133, 17.1, °C, temperature outside.

Note. Measurementvar is used to store one-time measurements which form part of a test, such as temperature outside.

2.5.4.6 #EQUIPMENT

#Equipment=sText

sText	Text stating the name of the equipment.

Example: #Equipment=TS200.

2.5.4.7 #TESTID

#Testid=sNumber

sNumber	Text stating which test from a series it concerns.

Example: #Testid=T13B.

2.5.5 Sample descriptive

2.5.5.1 #SPECIMENCODE

#Specimencode=sCode, iRelease, iVersion, iUpdate[, sIsoref]

sCode	Text stating how the samples must be collected. Maximum 80 characters.
iRelease	A number giving the release of the sampling.
iVersion	A number giving the version of the sampling.
iUpdate	A number giving the update of the sampling.
sIsoref	Reference of the ISO-9000 standard with which the sample complies. Maximum 80 characters.

Example: #Specimencode=MN, 3, 1, 0, 300-1-Mons.

2.5.5.2 #SPECIMENTEXT

#Specimentext=1Number, sText[, sInformation]

1Number	The number of the text variable. Maximum 1500.
sText	The text belonging to the sample. Maximum 255 characters.
sInformation	Information about the text. Maximum 80 characters.

Example: #Specimentext=129, Sample fell apart during transportation, general comment.

2.5.5.3 #SPECIMENVAR

#Specimenvar=1Number, 4Value, sUnit, sQuantity

1Number	Sequence number of the variable. Maximum 1500.
4Value	Value of the variable.
sUnit	Unit in which the variable is expressed. Maximum 40 characters.
sQuantity	The quantity which the variable represents. Maximum 80 characters.

Example: #Specimenvar=17, 0.37, -, porosity.

Note. Specimenvar seems very similar to Measurementvar as far as structure and use are concerned. The difference is that the information is specifically related to a sample with Specimenvar, while Measurementvar is concerned with quantities which are specifically related to measuring and measurement apparatus where appropriate.

2.5.6 Report descriptive

2.5.6.1 #REPORTDATAFORMAT

#Reportdataformat=sLayout

sLayout	The layout of the values to be printed sLayout consists of a series of words, separated by blanks. The number of words must be equal to the number of columns, as started in #COLUMN=. Each word consists of one letter and one or more digits. This word determines how a value is printed. The convention of the FORTRAN format identifier is used: In Integer, n digits long. Fw.d Floating point number, w digits long, d decimals. Ew.d Exponential representation, w digits long,, d decimals.
Example:	#reportdataformat=F7.3 F7.3 E11.4 I6.

2.5.6.2 REPORTCODE

#Reportcode=sCode, iRelease, iVersion, iUpdate[, sIsoref]

sCode	Text stating how the report is compiled. Maximum 80 characters.
iRelease	A number giving the release of the report.
iVersion	A number giving the version of the report.
iUpdate	A number giving the update of the report.
sIsoref	Reference of the ISO-9000 standard with which the report complies.

Maximum 80 characters.

Example:	#Reportcode=Standard, 3, 1, 0, 300-1-SDS.

2.5.6.3 #REPORTTEXT

#Reporttext=1Number, sText[, sInformation]

1Number	The number of the text variable. Maximum 1500.
sText	The text belonging to the report. Maximum 255 characters.
sInformation	Information about the text. Maximum 80 characters.

Example: #Reporttext=129, Section 5 is confidential, general comment.

2.5.6.4 #REPORTVAR

#Reportvar=1Number, 4Value, sUnit, sQuantity

1Number	Sequence number of the variable. Maximum 1500.
4Value	Value of the variable.
sUnit	Unit in which the variable is expressed. Maximum 40 characters.
sQuantity	The quantity which the variable represents. Maximum 80 characters

Example: #Reportvar=17, 66, -, number of pages.

2.5.6.5 #LANGUAGE

#Language=sLanguage

sLanguage	String which states the language. Maximum 2 characters.
sQuantity	The quantity which the variable represents. Maximum 80 characters.

Example: #LANGUAGE=UK.

Note. This code word can influence the program which imports and analyses a GEF file. Default value is ‘NL’.

2.6 File descriptive

2.6.1 #COMMENT

#Comment=sText

sText	Optional random text, not necessary for the measurement, rather an annotation but as a marginal note. A maximum of 1500 comment lines can be stored. Maximum 255 characters.
Example:	#Comment=Repeat of the test of 1995-06-25.

2.6.2 #EOH

#Eoh=[sMeaningless]

sMeaningless	Text which consists of some commentary.
Example:	#Eoh=.

Note. This code word has no data behind the '=' character which needs to be used. It is optional whether anything is entered. In addition, this code word is completed using a Line Feed (Unix) or the Carriage Return and the Line Feed (DOS). When using binary data, no text is allowed.

2.6.3 #GEFID

#GEFID=iRelease, iVersion, iUpdate

iRelease	Release number of the geotechnical exchange format of the file.
iVersion	Version number of the geotechnical exchange format of the file.
iUpdate	Update number of the geotechnical exchange format.
Example:	#GEFID=1,0,0.

Note. This code word must be the first one to appear in the file. It must be written in capital letters. Using the first line, operation systems like Unix are able to deduce the type of file and which program or printer is needed to control the file.

2.6.4 #OS

#Os=sType

sType	Text stating for which operating system the data are stored. Maximum 10 characters.
Example:	#OS=UNIX.

Note. This code word is only important for binary data, Values other than DOS or UNIX are not permitted. DOS relates to little endian words, UNIX to big endian words.

2.6.5 #STRUCTURETEXT

#Structuretext=sCodeword, sParameter1[,sParameter2[,sParameter3[,,]]]

sCodeword	The code word for which the parameters are described.
sParameter1	The first parameter belonging to the code word.
sParameter [i]	The i-th parameter belonging to the code word, i goes from 1 up to the number of parameters.

Example: #Structuretext=GEFID, Releasenumber, Versionnumber, Updatenumber.

2.6.6 #STRUCTURETYPE

#Structuretype=sCodeword, sType1[,sType2[,sType3[,,]]]

sCodeword	The code word for which data types of the parameters are described.
sParameter1	The data type of the first parameter belonging to the code word.
sParameter [i]	The data type of the i-th parameter belonging to the code word, i goes from 1 up to the number of parameters.

Example: #Structuretype=GEFID, int2, int,2, int2.

APPENDIX 3:

DESIGN OF THE *GEF VERIFICATION* TOOL

GEF Verification was developed using Microsoft Visual C++ 5.0 which makes use of Microsoft Foundation Classes to easily develop windows applications that run on Windows 95 or Windows NT.

A lot of the reading and a significant amount of verifying was done using GEFLIB, a library developed by GeoDelft and NITG-TNO together. GEFLIB is an important part of ***GEF Verification***. The implementation of this library will be described in section 5.

GEF Verification was created using the Visual C++ application wizard for a Single Document Interface (SDI) application.

The source of the ***GEF Verification*** application consists of several C++ classes. These are listed below.

3.1 The classes of GEF Verification

CAboutDlg	The about box (created by the application wizard)
CData block	Concerns access to the data block
CDrawCurveView	Draws the curve view
CDrawFileView	Draws the file view
CDrawVerificationView	Draws the verification view
CGEFVerificationApp	The application itself (created by the application wizard)
CGEFVerificationDoc	The document(created by the application wizard)
CGEFVerificationView	The main view(created by the application wizard)
CGrondmech	This class links GEFLIB into the application
CMainFrame	The mainframe(created by the application wizard)
CNengeo	Concerns the reading and conversion of NENGEO-files
CVerification	Performs the verification
struct HEADER	A C-structure with the header information (GEFLIB)

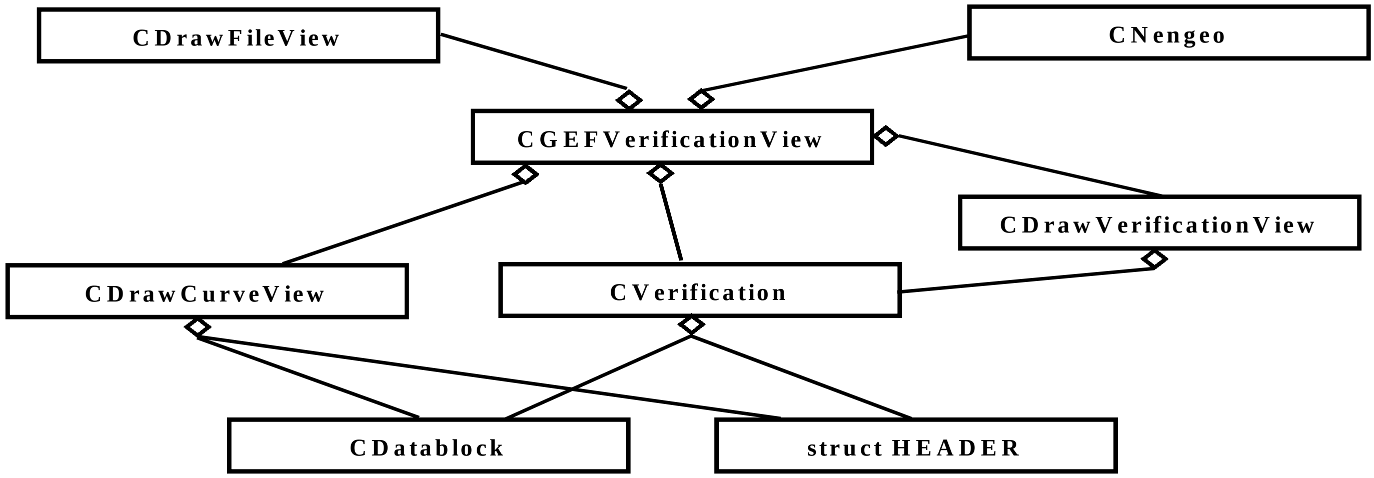
The classes created by the application wizard were not edited and will not be discussed here. These classes are the basis of the ‘standard’ windows application.

The only exception is CGEFVerificationView. This class forms the basis of the functional part of the application as can be seen in the chart displaying the object relationships between the classes. CGEFVerificationView is the class form which all menu items are handled:

- Convert NENGEO files	triggers CNENGEO
- Changing the view	triggers CDrawCurveView, CDrawFileView or CDrawVerificationView
- Open file	triggers CVerification and after that CDrawVerificationView

The link to GEFLIB is made via the empty class CGrondmech, where the files of GEFLIB are included and thereby linked to ***GEF Verification***. The GEFLIB subroutines are used in CVerification to read a GEF-file and write information to the logfile.

The object relationships between the most important classes:


CDrawFileView
CNengeo
CGEFVerificationView
CDrawVerificationView
CDrawCurveView
CVerification
CDatablock
struct HEADER

APPENDIX 4:

IMPLEMENTATION OF *GEF VERIFICATION*

This section is intended to document the member variables and member functions of the most important classes.

4.1 CData block

HEADER *my_header	*the HEADER structure of the file concerned*
char *cstyle_path_string	*string containing the location of the file*
char **q	*pointers to access the data block directly*
double **p	

int GetColNumOfQuantityID(int quant_id) — *returns the column in which to find a certain quantity, if not found return -1*

double DetermineMax(int column) — *determine the maximum value in a column*

double DetermineMin(int column) — *determine the minimum value in a column*

void Initialise(HEADER* par1, double** par2, char** par3, char* par4)
set the members variables

int GetNumScans() — *returns the number of scans using the header*

double GetData(int column, int line) — *returns a date element*

int GetNumColumns() — *returns the number of columns in the header*

4.2 CDrawCurveView

HEADER* TheHeader	*the HEADER structure of the file concerned*
CData block* TheData block	*the data block of the file concerned*
int xticknum	*the number of tickmarks on the x-axis*
int xtickspacing	*the spacing between tickmarks on the x-axis*
int xaxisxoffsetbegin	*the spacing in x-direction before the x-axis starts*
int yticknum	*the number of tickmarks on th y-axis*
int ytickspacing	*the spacing between tickmarks on the y-axis*
int yaxisyoffsetbegin	*the spacing in y-direction before the y-axis starts*
int scroll_y	*the y-size of the entire graph*
int scroll_x	*the x-size of the entire graph*
int ytickstart	*the value of the first tickmark in y-direction*
int ytickstop	*the value of the last tickmark in y-direction*

CSize GetScrollSizes() — *returns the size of the graph*

int CorrectOutlier(double& temp_val, double start, double end)
corrects x-values not to exceed the bounds of the graph

void DrawAxisY(CDC* pDC, int yaxisxoffset, int ytickoffset, int ytickwidth)
draws the y-axis

void DrawAxisX(CDC* pDC, int xaxisyoffset, int xtickoffset, int xtickwidth, double start, double end, int skip, int decimals, CString TheCaption, COLORREF TheColor)
draws the x-axis; the start and end can be specified, so can the number of decimals, the number of values displayed. the caption and the color of the line

void DrawCurve(CDC* pDC, int col_y, int col_x, COLORREF color, double start, double end)	
	draws the curve; the x-column, y-column, color, starting x-value and ending x-value (see x-axis) can be specified
void Initialise(HEADER* hp, CData block* dbp)	
	initialises the class
void DrawGrid(CDC* pDC)	*draws the grid*

4.3 CDrawFileView

int scroll_y	*the y-size of the entire graph*
int scroll_x	*the x-size of the entire graph*
CString TheFileName	*the filename of the file to display*
CSize GetScrollSizes()	*returns the size of the view*
void Draw(CDC* pDC)	*displays the view*
void Initialise(CString f_name)	*initialises the class*

4.4 CDrawVerificationView

CVerification* The Verification	*the verification class*
int scroll_y	*the y-size of the entire graph*
int scroll_x	*the x-size of the entire graph*
CSize GetScrollSizes()	*returns the size of the view*
void DrawLog(CDC* pDC)	*draws the view (displays the log)*
void Initialise(CVerification* vp)	*initialises the class*

4.5 CGEFVerificationView

CDrawVerificationView TheDrawVerificationView	
	the CDrawVerificationView member
CDrawFileView TheDrawFileView	*the CDrawFileView member*
CDrawCurveView TheDrawCurveView	*the CDrawCurveView member*
int verification_mode	*only header or header and data block*
CVerification TheVerification	*the CVerification member*
CString filename_string	*the name of the file concerned*
CString path_string	*the directory path of he file concerned*
int file_mode	*single file, batch mode or nothing selected*
int current_view	*the current view*
OnClose()	*closes a file*
OnDraw(CDC* pDC)	*draws the current view*
OnFileOpen()	*opens file(s) and automatically triggers the verification*
OnImport()	*converts NENGEO files to GEF-files*
OnViewCurve()	*switches the view*

OnViewFile()	
OnViewVerification()	
On.............()	*all other handling of the menu items is done in this class*

4.6 CNengeo

int max_num_records	*the number of columns*
double* data	*the NENGEO data block*
int num_scans	*the number of scans*
CString sond_header_text[10]	*the text at the beginning of the NENGEO file*
int num_header_lines	*the number of header lines*
double maaiveld	*the reference level*
char org_name[100]	*the original name of the CPT session*
int num_sond	*the number of CPT session in this file*
CString file_header_text[10]	*the text in the header of the CPT session*
char file_name[100]	*the filename*

int ConvertToCPTReport()	*converts the NENGEO data to create a GEF-file with the same name but extension .GEF*
int PrintToFile(CString fn)	*prints the NENGEO data to a file*
int GetNumRecords(CString line)	*gets the number of records in a line*
int Read()	*reads the NENGEO file into memory*

4.7 CVerification

HEADER my_header	*the HEADER structure of the file concerned*
CData block TheData block	*the data block*
char cstyle_path_string[100]	*string containing the location of the file*
char** q	*pointers to access the data block directly*
double** p	

int CheckVersionGEFID()	*verifies and warns if an other version was used*
int CheckMinMax()	*checks for minimum and maximum*
int CheckMinimumKeywords()	*checks if the obligatory keywords have been used*
void VerifyFile(CString path, int verification_mode)	*verifies a single file*
void VerifyBatch(CStringArray& files, int verification_mode)	*verifies a batch of files*
void LogFileInfo(CString path)	*writes information about the file to the logfile*
void Verify(CString path, int verification_mode)	*verifies a file*

APPENDIX 5:

IMPLEMENTATION AND USE OF GEFLIB

This section will describe the implementation and the use of GEFLIB.
GEFLIB was written in plain C and consists of 3 files:

- struct.h — a header file that contains the GEF data structure and a few ‘defines’
- geflib.h — a header file containing the declarations of the functions in geflib.c
- geflib.c — the source code of the library

The functions of GEFLIB can be grouped into 3 sections:

- high-level functions to read and write GEF-files
- low-level functions (documentation can be found in the comments in the source code)
- low level functions added specifically for verification purposes

Geflib.c contains a main( ) function in which the high-level functions are demonstrated. In combination with another main( ) function it will be necessary to delete this demonstration main( ) function.

To create a conversion program from format X to GEF one should create a GEF HEADER structure and fill it with the appropriate information extracted from format X. The HEADER structure has an element vlaggen which indicates if a certain part of the structure has been used by setting the flag concerned. To initialise these flags call init_vlaggen(HEADER* kop). After filling the HEADER structure and setting the flags the user can create an array of doubles and fill it with the data extracted from format X. If necessary an array of strings can be created and filled with column texts. If all header information and the data block are in memory, the only thing to do is call pr_sff(HEADER *kop, double *p, char **q, FILE *fuit, int dataformat) and a syntactically correct GEF-file will be created.

To create a conversion program from GEF to format X one should create a GEF HEADER structure, a pointer to the data and a pointer to the column texts. If verification of the GEF-file is desired remember to set the write_log_status, header_status and data block_status to one and the escape_status to zero and to call clear_log( ) and init_vlaggen(HEADER* kop).

This means adding the following code before rd_sff:

```
clear_log();
write_log_status(SET, 1);
header_status(SET, 1);
data block_status(SET, 1);
escape_status(SET, 0);
```

The data will be read into memory and verified by calling rd_sff(char *bestand, HEADER* kop, double **p, char ***q). The data is now in memory. The user should extract the data from the header by directly accessing the HEADER structure. The data block can be accessed using the high-level functions:
geef_waarde(HEADER *kop, double *data, int kolom, long skan) and geef_kolomtekst(kop, tekst, scan). With this information it is possible to create a file in format X.

High level functions to read and write GEF-files

void rd_sff(char *bestand, HEADER* kop, double **p, char ***q)
reads and verifies syntax of a GEF-file

void pr_sff(HEADER *kop, double *p, char **q, FILE *fuit, int dataformat)
writes GEF data in memory to a file

int geef_aantal_kolommen(HEADER* kop) *returns the number of columns*

long geef_aantal_scans(HEADER* kop) *returns the number of scans*

double geef_waarde(HEADER *kop, double *data, int kolom, long skan)
returns the value of a specific data element

char* geef_eenheid(HEADER *kop, int kolom)
returns the unit of the data in a column

char* geef_grootheid(HEADER *kop, int kolom)
returns the quantityof the data in a column

double* geef_kolom_adres(HEADER *kop, double *data, int kolom)
returns the adress of a column

char* geef_kolomtekst(kop, tekst, scan) *returns the column text of a specific scan*

int fr_sff(kop, data, tekst) *frees the memory that was allocated for the data block and several allocated items in the header*

void init_vlaggen(HEADER* kop) *initialises the flags of the header*

Low level functions added specifically for verification purposes

The following two functions have to do with the logfile. The location of the logfile can be set in struct.h by altering the define LOGFILENAME. The effect setting the bold or tab flag in write_log is that certain text is added before the line is added to the file. This text can also be found in struct.h after defines BOLDSTRING, NOTBOLDSTRING, TABSTRING

int clear_log() *clears the log file*

int write_log(const char* text, int bold, int tab)
adds a line to the logfile

The following functions record or return the status of a certain variable. They act as global variables. The first parameter determines weather the is returned or set. The first parameter can have value SET (=1) or value GET(=0)

int write_log_status(int get_set, int value) *subroutine in which the status of writing to the logfile is recorded:*
- one if write_log should write to file
- zero if write_log should do nothing

int escape_status(int get_set, int value) *subroutine in which the status of ‘whether or not to escape to the main program’ is recorded:*
- one if an escape should take place
- zero if not

int header_status(int get_set, int value) *subroutine in which the status of reading the header is recorded:*
- one if there are no errors
- zero if an error occurred
- minus 1 if a serious error occurred

int data block_status(int get_set, int value)

- *one if there are no errors*
- *zero if an error occurred*
- *minus 1 if a serious error occurred*

The following functions verify the parameters after a keyword:

int verify_par(void* waarde, int type, char* keyword, int num)

verifies the type of a parameter, detects the absence of a required parameter

int verify_too_much(void* wat, char* keyw, int num)

verifies if the number of parameters after a keyword is not to large