SERIES OF TECHNICAL PRESENTATIONS RELATING TO THE DIGITAL DOCUMENT INTERCHANGE FORMAT

BABBAGE AUDITORIUM, SPIT BROOK ROAD FACILITY

JULY 9, 10, 11

Wednesday July 9

9:30  Review of DDIF status
      Review/update of agenda

10:00  Introduction to DDIF  Bob Travis

DDIF has been under development at Digital for over a year, and forms the basis for two current document processing development efforts. This introductory presentation will describe DDIF's syntax and semantics.

11:30  lunch break

13:00  DDIF/DDIS VMS RTL Access  Bill Laurune

Two utility packages will be described. The first package provides low level access to DDIS TLV-encoded data. The subroutines will provide basic encoding functions (such as length field construction) and syntax checking based on dynamically loaded syntax tables.

The second routine package is DDIF specific, and provides the service of maintaining the current context of DDIF segment attributes.

The DDIS access routine package is scheduled for submission to the VAX/VMS Run-Time Library in V5.0. The DDIF access routines are not currently scheduled for RTL submission, but will be available for by DEC applications.

14:20  coffee break

14:40  VPSPLUS/DECPAGE logical access  Bruce Taylor

The Logical Access Layer (LAL) is planned as a functional interface to DDIF documents. It provides primitives for data editing, hierarchical navigation, searching and attribute resolution. For each open document, the LAL maintains a tree representing structure and content. If the document is large, not all of this data is represented explicitly at a given time; but from the caller's point of view, the entire document appears to be "in memory."
This talk will describe a high-level design for the LAL and its internal data structures, together with two related components: the Cache Manager, which controls paging of the data structure; and the Physical Access Layer, which converts between the internal format and DDIF.

16:00 DDIS Character Sets  
Tom Hastings

DDIS is the basic data syntax upon which DDIF is based. It defines the encoding of all binary and character items. Fundamental to any DDIF application are the text character sets; DDIS defines both 8- and 16-bit character set encodings. This session will describe the on-going process to define these character sets, and the pending issues surrounding them.

17:00 end of day

Thursday July 10

9:30 Status of standards in computer graphics & documents  
Dr. J. Schönhut et al.

Under the auspices of an external research grant, Fraunhofer-Gesellshaft and the Technical University at Darmstadt researchers have been doing some investigations into the application of DDIF to existing and future graphics applications. All of today's sessions are related to various aspects of that research, and will be presented by Dr. Jürgen Schönhut and some of his associates from the Fraunhofer-Gesellshaft.

10:30 Compatibility of graphics stds: PHIGS, GKS, etc.  
Schönhut et al.

11:30 lunch break

13:00 PDC: A device independent graphics interface to DDIF/LAL  
Paul Wong

14:00 Interfaces and data formats for transfer and communications in computer graphics systems  
Schönhut et al.

15:00 coffee break

15:20 text models in computer graphics, Videotex and documents  
Schönhut et al.

16:00 Comparison of DDIF, SGML, ODA/ODIF  
Schönhut et al.

17:00 end of day
Friday July 11

9:30  Role of DDIF in the Compound Document Processing Strategy  Jim Kapadia

10:30  SLIDEX: an example for integration of text and graphics  Schönhut et al.

11:30  end of seminar
CURRENT DDIF STATUS:
- "24-OCT" version used by WPSPLUS and SARAH prototypes
- "14-JAN" version now in use in SARAH field test
- Several improvements made since then; targeted for Summer release
- Formal Review Process being established in SSG and DS
Introduction to DDIF

Bob Travis

July 9, 1986
What is DDIF?

DDIF is a storage and interchage format for compound documents in revisable form. The primary purpose of DDIF is to serve as a medium of exchange for revisable documents between Digital compound document processors. But many document creating applications will simply store user files in DDIF in order to greatly simplify the inter­change process, and also because DDIF has been designed to be an efficient representa­tion for revisable compound documents.

It will be a formal DEC standard after it has been successfully implemented and released in at least one product.
What is a Compound Document?

A compound document is a unified collection of data that may be edited, formatted, or otherwise processed as a document. A compound document is likely to contain a number of integrated components, including proportionally spaced text in various renditions and styles, positioning and formatting parameters, abstract synthetic graphics, and scanned images. In the future, other forms of data representation may be tightly integrated with these document components.
DDIF's Goals:

- Completeness
  - high quality text and graphics
  - structure
  - suitability for supporting application integration

- Ease of Processing
  - clear syntax
  - usable on small systems

- Ease of Maintenance
  - self-describing syntax
  - provision for upward migration and version tracking

- Common Encoding for Interchange, Mail
o DDIS domain
Key Points:

- Based on DDIS
- Hierarchical Structure
- Generic Elements
- Text, Graphics, Image, Foreign' data
- Links to external content and attribute info
Review of DDIS:
- TLV encoding
- Formal meta-syntax for Domains
- Defines basic data elements, and constructors
- Based on international standards (X.409, ANS-1)
DDIF DOCUMENT ::= 
Descriptor 
identifies this as DDIF, also what 
version 
Header 
Title, author, dates, fo. s, external 
files, 
page layouts, print parameters, etc. 
Content 
Hierarchical segment structure, with 
content and attributes
Segment:

BeginSegment element

Contains all segment attributes

Series of content primitives and/or nested segments

Provides the actual content of the segment

EndSegment element

No added information; just terminates the segment
Content Primitives:

**Text strings — 8- and 16-bit character sets**
- Directives — Line, page, block, tab, layout
- Graphics — Multipoint, arc, spline
- Image — Bi-tonal, grayscale multispectral
- Macro-reference — Includes standard content, transformed
- Other — PLP, Domain, Private
Text:

LATIN1_STRING — the DEC-standard multi-national set

TEXT_8 — first byte of string selects 8-bit set

TEXT_16 — first byte-pair selects 16-bit set
Directives:

   Hard/soft text — integer value, selects Page, Line, etc

   Page Layout — integer value, selects new page layout
Graphics:

Multipoint — sequence of points, draw pattern, draw/fill/mark/reg/close flags

Arc — center, radius, eccentricity, start/extent, rotation, draw/fill/pie/close flags

B_Spline — sequence of points, order, draw/fill/close flags

Image:

— sequence of planes, with coding attributes, pixels
— rich variety of coding schemes, component ordering
— compatible with CCITT FAX for btonal
Other content:

Macro Reference — with transformation

PLP — included 'pictures' in existing protocols: ANSI, GKS, etc

Domain, Private — application data, not directly rendered
Coding Example:

ARC_DEF ::= [APPLICATION 11] IMPLICIT
  SEQUENCE {
  Arc_Center_X [0] IMPLICIT MEASURE,
  Arc_Center_Y [1] IMPLICIT MEASURE,
  Arc_Radius_X [2] IMPLICIT MEASURE,
  Arc_Radius_Delta_Y [3] IMPLICIT
    MEASURE DEFAULT 0,
  Arc_Start [4] IMPLICIT ANG' E DEFAULT
    0,
  Arc_End [5] IMPLICIT ANGL E DEFAULT
    21600,
  Arc_Rotation [6] IMPLICIT ANGLE
    DEFAULT 0,
  Arc_Flags [7] IMPLICIT INTEGER {
    Draw_Arc(1), Fill_Arc(2), Pie_Arc(4),
    Close_Arc(8) } DEFAULT 1 }
Begin Segment element:

ID — for reference from other segments

User Label — for application reference via UI

Generic Reference — for indirect attribute specification

Local Attributes — for direct attribute specification

Element Definitions — supply source for generic attrs and macros

Structure Description — supplies rules for segment construction
Generic Referencing:

[Parent segment
   Element definitions: X, Y

[...
[Child segment
   Generic reference = X ]

...]
]
Attribute Inheritance:

[Parent segment
  Local attributes: text font1, colors 1/0, other (initial)

[Child 1
  Local attributes: font2]

[Child 2
  nolocal attributes]
Attributes:

Semantic Tags — indicate paragraph, chapter, etc
Frame — clipping rectangle, relocation, transformation
Color Map — in RGB
Text — font, color, margins, tabs
Character — orientation, alignment
Line — thickness, color, end, pattern
Fill — pattern, color
Marker — selection, color
Conversion — format of section numbers etc.
Computed Content — numbers, cross-reference, external
Page Layout — selects page layout
Language — selects linguistic support
Image — type, orientation, etc
Page Layouts:

- in Header
- indexed from content and/or attributes
- sequence of BLOCKS
- each block either predefined content or poured from content
- filled in sequence
- each page layout can link to successor
Transformations:
- Sequence of Transformation Primitive

Transformation Primitives:
- translation, rotation, skew, scaling
Recently Added:
- Image content and attributes
- Hierarchical generic/content definitions

Work in Progress:
- Improved layout for annotation and footnotes
- Table specification, including link to data/spreadsheet
- Review for use in engineering drawing domain
Bill Lahurune

DD/5 Access Routines
<table>
<thead>
<tr>
<th>CLASS</th>
<th>INTERFACE (examples)</th>
<th>EXAMPLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Processor</td>
<td>Interapplication Protocol</td>
<td>DDIF Routines 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DDIF Routines 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Encoding Stream</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DDIS Routines 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PUT</td>
</tr>
<tr>
<td>Physical Storage</td>
<td></td>
<td>I/O Routines 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RMS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Files 11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DECnet</td>
</tr>
</tbody>
</table>
Figure Drawing Convention: Memory locations are shown with increasing addresses running right to left & top to bottom. If this data were to be transmitted over a communication line, the order of transmission is low-address bit/byte first.

```
+------+
| TYPE  | : A |
+------+
+-------+
| LENGTH (L1) : B | Data Element 1 |
+-------+
| VALUE  : C |
+-------+

+-------+
| TYPE  | : C+L1 |
+-------+
+-------+
| LENGTH (L2) : D | Data Element 2 |
+-------+
| VALUE  : E |
+-------+
```

TYPE Field  - variable length (1 .. 4 bytes)
LENGTH Field - variable length (1 .. 5 bytes)
VALUE Field  - variable length (0 .. 2**32-1 bytes for primitive & counted constructor, unlimited for uncounted constructor)
1-BYTE TYPE FIELD

<table>
<thead>
<tr>
<th>7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+-----------------</td>
</tr>
<tr>
<td>: A</td>
</tr>
<tr>
<td>+-----------------</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td>v</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td>v</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td>v</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td>v</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td>v</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td>v</td>
</tr>
<tr>
<td>__________</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Class Bits - TYPE <7:6>

- 00 = Universal
- 01 = Application wide
- 10 = Context-specific
- 11 = Private (Customer/OEM)

Form Bit - TYPE <5>

- 0 = Primitive
- 1 = Constructor

ID Code Bits - TYPE <4:0>

- 00000 .. 11110 = ID Code
- 11111 = ID Code is encoded in one or more extension bytes

FIGURE 2a
EXTENSION OF TYPE FIELD

1. 1 Byte:

```
+---------+
| m s b    |
+---------+
7 6 5 4 3 2 1 0
```

ID Code range
0 .. 30

```
/   |   ID Code
| Form
| bit
Class
bits
```

2. 2 Byte:

```
+---------+
| m s b    |
+---------+
7 6 5 4 3 2 1 0
```

ID Code range
31 .. 127

```
/   |   ID Code
| Extension
| bit
| A+1
```

The ID Code shall be encoded in the fewest possible bytes.

FIGURE 2b
EXTENSION OF TYPE FIELD (continued)

3. 3 Byte:  

```
7 6 5 4 3 2 1 0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| c | c | f | l | l | l | l | l | : A
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
 m   l
 s   s
 b   b
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| l | | | | | | | : A+1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| l | | | | | | | : A+2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-

ID Code range
128 .. 2**14-1
```

4. 4 Byte:  

```
7 6 5 4 3 2 1 0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| c | c | f | l | l | l | l | l | : A
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
 m   l
 s   s
 b   b
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| l | | | | | | | : A+1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| l | | | | | | | : A+2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| 0 | | | | | | | : A+3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-

ID Code range
2**14 .. 2**21-1
```

ID Code (Note: bit 7 of extension bytes is not included in ID Code)

The ID Code shall be encoded in the fewest possible bytes.

FIGURE 2c
LENGTH FIELD

1. Short form: 7 6 5 4 3 2 1 0 
   Length range 0 .. 127
   Length

2. Long form: 7 6 5 4 3 2 1 0 
   Length range 128 .. 2**32-1
   Length

3. Indefinite form: 7 6 5 4 3 2 1 0 
   Length

May (but need not be) used for constructors. Shall not be used for primitives.

FIGURE 3
CONSTRUCTORS

(Example: assume Type & Length fields are 1 byte long)

1. Uncounted Constructor

```
+++++++
| Indef | T  | : A
+++++++

| V1    | L1 | T1 | : B
+++++++
| V2    | L2 | T2 | : B+L1+2
+++++++
|       |    |    | ...
|       |    |    | ...
| Vm    | Lm | Tm | : B+L1+...+Lm+2m
+++++++
```

/ end-of-constructor element

2. Counted Constructor

```
+++++++
| L    | T  | : A
+++++++

| V1    | L1 | T1 | : B
+++++++
| V2    | L2 | T2 | : B+L1+2
+++++++
|       |    |    | ...
|       |    |    | ...
| Vn    | Ln | Tn | : B+L1+...+Ln+2n
+++++++
```

where \( L = L1+L2+...+Ln+2n \)

FIGURE 4
**UNIVERSAL ID CODES**  
**(BUILT-IN TYPES)**

<table>
<thead>
<tr>
<th>ID Code</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>End_of_Constructor</td>
</tr>
<tr>
<td>1</td>
<td>Boolean</td>
</tr>
<tr>
<td>2</td>
<td>Integer</td>
</tr>
<tr>
<td>3</td>
<td>Bit_String</td>
</tr>
<tr>
<td>4</td>
<td>Octet_String</td>
</tr>
<tr>
<td>5</td>
<td>Null</td>
</tr>
<tr>
<td>6</td>
<td>F_Float</td>
</tr>
<tr>
<td>7</td>
<td>G_Float</td>
</tr>
<tr>
<td>8</td>
<td>D_Float</td>
</tr>
<tr>
<td>9</td>
<td>H_Float</td>
</tr>
<tr>
<td>10-15</td>
<td>RESERVED FOR FUTURE STANDARDIZATION</td>
</tr>
<tr>
<td>16</td>
<td>Sequence</td>
</tr>
<tr>
<td>17</td>
<td>Set</td>
</tr>
</tbody>
</table>

* Although not a data type, the End_of_Constructor data element consumes an ID code.

**FIGURE 5**
DDIS ACCESS ROUTINES

1. Sponsored by Display Systems Software

2. Used in GPU (Sarah) file I/O system

3. Suitable for RTL usage
   A. Shareable Code
   B. I/O independent
   C. Syntax Independent
   D. Written in Bliss-32

4. Advantages of RTL support (VMS 5.0)
   A. Allows customers to read & write DDIS
   B. Reduces size of applications
   C. Provides extensive error checking
   D. Extended testing

5. Issues:
   How do we supply syntax to customer?
   DDIS must be very stable
DDIS FEATURES

1. Any constructor may be counted or indefinite
2. Full support of SET
3. Support of all DDIS syntax, including ANY
4. Strict syntax enforcement
5. Default Values are TBD

RESTRICTION:

1. Tag Length: 4 Bytes or Less
2. Maximum value length: $2^{32} - 1$ bytes
3. Max. Stream Length: currently $2^{32} - 1$ bytes
CONTEXT BLOCK MANAGEMENT

DDIS$ALLOCATE_CONTEXT

1. Allocates context block, its buffers & stacks
2. Initializes internal queue headers
3. Associates Parse Tables
4. Parameters: Context block return
   Get memory routine address
   Free memory routine address
   I/O routine address
   Condition handler address (established)
   Parse table addresses...

DDIS$DEALLOCATE_CONTEXT

0. Parameters: Context block address
1. Deallocates context block & substructures
2. Application should check parse state
**DDIS Read Routines**

1. **DDIS$GET_TAG**
   
   A. Parameters: context block address
   B. Skips value if pending
   C. Tag is read into context block
   D. Tag is located in the parse table (or error)
   E. Length is decoded and placed in context block
   F. Returns end of counted constructor as EOC tag

2. **DDIS$GET VALUE**
   
   A. Parameters: context block [, buffer]
   B. Uses context block buffer by default
   C. Built-in buffer may be reallocated
   D. Error if no value is available (e.g. EOC)

3. **DDIS$SKIP_TO_EOC**
   
   A. Parameters: context block
   B. Proceeds to end of current constructor
   C. No parse table checking - can skip private data &c
WRITE ROUTINES

1. DDIS$PUT_TLV

A. Parameters: context block [,tag] [,length] [,value]
B. Tag is checked against syntax
C. Length is encoded
D. Value is written (integers are compressed)
E. Constructor is counted if length is passed
F. Counted constructors must end on a tag

2. DDIS$PUT_EOC

A. Parameters: context block, count
B. Writes a number of EOCs to the stream
C. A convenience routine, NYI

2. DDIS$GET_INTLEN

A. Parameters: Maximum length, address
B. Returns length of compressed integer
1. SEQUENCE

A. Listed tags must be in designated order
B. Tags must be present unless OPTIONAL
C. No other tags can be present

2. SET

A. Listed tags may be in any order
B. But no tag may be repeated
C. Tags must be present unless OPTIONAL

3. CHOICE

A. Only one may be selected
B. May be nested in SET or SEQUENCE
C. Forms a constructor if tagged

4. DDIS routines parse TLV stream

A. Use a dynamically loaded parse table
B. Maintains unique context in table (supports dispatch)
C. Discourages creation of invalid files, structures
D. Catches logic errors (during execution)
1. Tree structure reflects syntax
2. Contains implicit types
3. Each tag has an unique entry
4. Flags for OPTIONAL, CHOICE, etc
5. Common definition, common nodes
6. Tree structure is PIC & shareable

.MACRO TAG tag
  .LONG tag
  .ENDM TAG

.MACRO DESCENDANTS location
  .WORD location
  .ENDM DESCENDANTS

.MACRO TYPE type
  .BYTE type
  .ENDM TYPE

.MACRO FLAGS flags
  .BYTE flags
  .ENDM FLAGS
PART OF DDIF TABLE

DDIF DDIS TABLE::

LABEL1 :  TAG DDIF$C_DDIF ; 0
DESCENDANTS LABEL2-LABEL1
TYPE ddis$c_sequence
FLAGS 0

END OF CONSTRUCTOR ; 1

LABEL2 :  TAG DDIF$Cocument_Descripto
DESCENDANTS LABEL30-LABEL2
TYPE ddis$c_sequence
FLAGS 0

LABEL3 :  TAG DDIF$Cocument_Profile
DESCENDANTS LABEL35-LABEL3
TYPE ddis$c_sequence
FLAGS 0

LABEL4 :  TAG DDIF$Cocument_Header
DESCENDANTS LABEL83-LABEL4
TYPE ddis$c_sequence
FLAGS 0

LABEL5 :  TAG DDIF$Cocument_Content
DESCENDANTS LABEL103S-LABEL5
TYPE ddis$c_sequence
FLAGS 0

END OF CONSTRUCTOR ; 6
DEVELOPMENT ENVIRONMENT

What’s needed:

A. BNF Compiler, table builder
B. Symbols for DDIS tags: EOC, INTEGER, etc
C. Symbols for application tags: DDIF, etc
D. File Analyzer, Dump Facility
E. Run-time Debugging Tools

What we have so far:

A. PTU: Prototype BNF Compiler, table builder
B. SDL file of Symbols for DDIS tags
C. PTU creates Symbols for application tags
D. TAN: File Analyzer, Dump Facility
E. DDIS module with audit trail
F. ..and the debugger works with this stuff

may need a complete compiler...
PARSE TABLE UTILITY

1. PTU compiles BNF into Macro-32 declarations
2. Human readable output, for development (.mar)
3. Tables can be linked to application (.obj)
3. Or tables can be loaded dynamically (.exe)
4. PTU provides symbol declarations (require files)
TLV ANALYZER (TAN)

1. Dumps a DDIS file in MACRO-32
2. Output can be re-assembled, linked
3. Displays tags by name (PTU-generated)
4. Parse tables can be loaded
5. DDIF Parse table is built in

; This TAN X3.7 output was generated on 27-MAR-1986 08:11:04.
; from the file DDIF$:[EXAMPLES]SIMPLE_PARAGRAPHS.DDIF;4

; 306 symbols read from DDIF$:[PRGSRC]DDIF_ENTRY_SYMBOLS.T

;Based on built-in parse table for DDIF
;
.PSECT ddis,rd,nowrt,byte,shr
;
C0: .BYTE  'B00111111 : Entry = 0 (DDIF$C_DDIF)
     .BYTE  'B10000000 : Continued tag
     .BYTE  'B00000001 : Continued tag
     .BYTE  'X80 : Indefinite Length

C1: .BYTE  'B00110000 : Entry = 2 (DDIF$C_DOCUMENT_DESC
     .BYTE  'X80 : Indefinite Length

P0: .BYTE  'B10000000 : Entry = 18 (DDIF$C_MAJOR_VERSION
     .BYTE  'X00 : Length
     ; Value (INTEGER) = virtual zero
Logical Access Layer

Introduction

- purpose of LAL
- related components
  - KODDIF
  - Physical Access Layer (PAL)
  - Cache Manager

Logical Access Layer

- internal data-structure
- general purpose operations
- application specific operations

Physical Access Layer

- relevant KODDIF features
- action routines
- conversion between internal data structure and DDIF constructs

Cache Manager

- writing from internal data structure
- pruning the " " "
- reading into " " "

Current design vs. Gold prototype

Conclusions

presented by Bruce Taylor
M/S ZK02-1/N20
E-Net MAGIC::TAYLOR

LAL 9-July-1986
Internal data structure

A tree for each open DDIF document

Interior nodes (complex objects)
- correspond to DDIF segments
- have attributes, children
- format is application independent

Leaf nodes (primitive objects)
- correspond to DDIF content elements or low-level segments
- may encode rendition information
- formats are application specific
General purpose operations

Object naming
Attribute resolution
Data editing
Search for attributes
Navigation
Secondary index creation
Document level operations
Object naming scheme

Object variables: used by an application to access objects

Similar to pointers

• repositioned during navigation

Unlike pointers

• retain connection with objects no longer in-memory
• don’t allow direct manipulation of data structure

Managed by LAL functions:

• create_object_variable
• destroy_object_variable
Attribute resolution

Terms:

• specific attributes
• effective attributes
• attribute resolution

Computing effective attributes

• specific attributes
• generic references
• defaults
• inheritance

Problem: relative attributes

• relative margins
• transformations
General purpose data editing

Create complex object
Delete complex object
Cut, paste

Other operations, concerning

- segment id’s
- external references
- data type boundaries
- access rights to objects
Navigation

Three kinds of navigation:

- tree navigation
- navigation by unique segment id
- navigation by secondary index

Secondary index:

- stores page breaks, line breaks, etc.
- a set of pointers into the stored document
- created, managed by application
- LAL provides two services:
  - make_primary_key
  - position_using_primary_key
Application specific operations

General categories:

• create primitive object with content
• search for primitive content
• get copy of primitive content
• modify primitive content
• specialized operations on structure and content
Application specific operations: examples

Text Editor
- apply_rendition_change
- split, merge

Graphics Editors
- transform_2d_object

Attribute Editor
- specify_attributes

Untyped operations
KODDIF

Two major components:

Access method

- document-level services
- positioning of "streams" at content elements
- reading, insertion, update, deletion at a stream

KODDIS with DDIF parse table

- input is driven by DDIF syntax
- output is controlled by PAL
KODDIS

Encodes, decodes DDIS TLV's (primitives and constructors)

Uses two tables:

- parse table
- action table

Tables are generated for an application by Koala DDIS Utility (KDU)

Input: KODDIS checks syntax, invokes appropriate action routine when an element is recognized

Output: KODDIS generates DDIS strings, while checking syntax. Provides calls to

- start a constructor
- end a constructor
- output a primitive value
Physical Access Layer

Input ("action") routines convert DDIF content elements to tree nodes

- invoked by KODDIS
- correspond to DDIF constructors or primitives
  - allocate a tree node
  - assign a value to a field
  - link completed node into tree structure

Output routines convert tree nodes to DDIF

- call each other
- at lowest level, call KODDIS
  - begin a constructor
  - output a primitive element
  - end a constructor
Cache Manager

Writing from internal data structure

- occurs before document is closed or checkpointed
- may occur before internal d.s. is pruned
- objects are written out using PAL
- only changed objects are written out

Pruning internal data structure

- occurs when the d.s. is getting too large
- least recently used tree nodes are written out, deallocated from tree
- some information is saved to facilitate later retrieval
Reading into internal data structure occurs

- when existing document is opened
- after roll-back to a previous document state
- when an application tries to access an object not in-memory
Current design vs. Gold prototype

In the prototype, "LAL" corresponded to Physical Access Layer

- each DDIF content element was read into a temporary d.s.
- LAL extracted useful information and added to editor's d.s.

"Document Manager" corresponded to LAL

- provided a view of a section of the stored document
- document manager, editor shared this data structure
- editor had to distinguish between "in memory", "on disk" data and compute effective attributes
Current design

- avoids an extra conversion
- encapsulates application-independent functions
  - attribute handling
  - common data editing operations
- eliminates "in memory", "on disk" distinction
- provides a cleaner interface to document
Editing Components

TEXT

GRAPHICS

ETC.

LOGICAL ACCESS LAYER

CACHE MANAGER

PHYSICAL ACCESS LAYER

KODDIF

DDIF
DDIS & DDIF

Character Sets

9 - July - 86

T. Kashihs
<table>
<thead>
<tr>
<th>Code</th>
<th>Character Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RESERVED</td>
</tr>
<tr>
<td>1</td>
<td>DEC MCS (has only 1 character)</td>
</tr>
<tr>
<td>2</td>
<td>Latin-2 (Eastern Europe)</td>
</tr>
<tr>
<td>3</td>
<td>Latin-3 (Southern Europe)</td>
</tr>
<tr>
<td>4</td>
<td>Latin-4 (Northern Europe)</td>
</tr>
<tr>
<td>5</td>
<td>Greek</td>
</tr>
<tr>
<td>6</td>
<td>Cyrillic</td>
</tr>
<tr>
<td>7</td>
<td>Hebrew</td>
</tr>
<tr>
<td>8</td>
<td>Arabic</td>
</tr>
<tr>
<td>9</td>
<td>Technical character set</td>
</tr>
<tr>
<td>10</td>
<td>Publishing character set</td>
</tr>
<tr>
<td>11</td>
<td>Output Rendering character set</td>
</tr>
<tr>
<td>12</td>
<td>Special Graphics (Line Drawing) char set</td>
</tr>
<tr>
<td>13-127</td>
<td>RESERVED FOR FUTURE STANDARDIZATION</td>
</tr>
<tr>
<td>128-254</td>
<td>Reserved for use by DEC customers/OEMs</td>
</tr>
<tr>
<td>255</td>
<td>RESERVED FOR FUTURE EXTENSIONS</td>
</tr>
</tbody>
</table>

FIGURE 6
### SUMMARY OF DDIS CHARACTER SETS

<table>
<thead>
<tr>
<th>A.2</th>
<th>ISO LATIN-1 CHARACTER SET USED IN LATIN1_STRING</th>
<th>A-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.3</td>
<td>TEXT8_STRING - ADDITIONAL 8-BIT CHARACTER SETS</td>
<td>A-2</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>A.3.1</td>
<td>DDIS-DEC Multinational (DEC MCS)</td>
<td>A-21</td>
</tr>
<tr>
<td>A.3.2</td>
<td>DDIS ISO Latin-2 Character Set (Eastern Europe)</td>
<td>A-29</td>
</tr>
<tr>
<td>A.3.3</td>
<td>DDIS ISO Latin-3 Character Set (Southern Europe)</td>
<td>A-33</td>
</tr>
<tr>
<td>A.3.4</td>
<td>DDIS ISO Latin-4 Character Set (Northern Europe)</td>
<td>A-37</td>
</tr>
<tr>
<td>A.3.5</td>
<td>DDIS ISO Greek Character Set</td>
<td>A-41</td>
</tr>
<tr>
<td>A.3.6</td>
<td>DDIS ISO Cyrillic Character Set</td>
<td>A-44</td>
</tr>
<tr>
<td>A.3.7</td>
<td>DDIS DEC Hebrew Character Set</td>
<td>A-47</td>
</tr>
<tr>
<td>A.3.8</td>
<td>DDIS ISO Arabic Character Set</td>
<td>A-51</td>
</tr>
<tr>
<td>A.3.9</td>
<td>DDIS DEC Technical Character Set</td>
<td>A-55</td>
</tr>
<tr>
<td>A.3.10</td>
<td>DDIS DEC Publishing Character Set</td>
<td>A-58</td>
</tr>
<tr>
<td>A.3.11</td>
<td>DDIS DEC Output Rendering Character Set</td>
<td>A-62</td>
</tr>
<tr>
<td>A.3.12</td>
<td>DDIS DEC Special Graphics (VT100 Line Drawing) Character Set</td>
<td>A-67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A.4</th>
<th>TEXT16_STRING - 16-BIT CHARACTER SETS</th>
<th>A-70</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>A.4.3</td>
<td>DDIS JIS Japanese Character Set</td>
<td>A-72</td>
</tr>
<tr>
<td>A.4.4</td>
<td>DDIS GB Chinese Character Set</td>
<td>A-93</td>
</tr>
</tbody>
</table>
SUMMARY OF DDIS CHARACTER SETS

A.2 ISO LATIN-1 CHARACTER SET USED IN LATIN1_STRING .... A-8
     ISO 8859/1 Latin Alphabet Nr 1 - approved 1986
A.3 TEXT8_STRING - ADDITIONAL 8-BIT CHARACTER SETS .... A-21
     0 Reserved
A.3.1 1 DDIS DEC Multinational (DEC MCS) ............... A-21
     DEC Std 169 DEC Multinational - approved May 1982
A.3.2 2 DDIS ISO Latin-2 Character Set (Eastern Europe) .... A-29
     ISO 8859/2 Latin Alphabet Nr 2 - approved 1986
A.3.3 3 DDIS ISO Latin-3 Character Set (Southern Europe) .... A-33
     ISO dp 8859/3 Latin Alphabet Nr 3 - draft
A.3.4 4 DDIS ISO Latin-4 Character Set (Northern Europe) .... A-37
     ISO dp 8859/4 Latin Alphabet Nr 4 - draft
A.3.5 5 DDIS ISO Greek Character Set .................... A-41
     ISO dp 6937/7 - draft
A.3.6 6 DDIS ISO Cyrillic Character Set .................. A-44
     ISO DIS 6937/8 - draft
A.3.7 7 DDIS DEC Hebrew Character Set .................. A-47
     Based on Hebrew 7-bit set
A.3.8 8 DDIS ISO Arabic Character Set .................. A-51
     ASMO Latin/Arabic 8-bit standard
A.3.9 9 DDIS DEC Technical Character Set ............. A-55
     DEC Technical Character Set spec (not ISO draft)
A.3.10 10 DDIS DEC Publishing Character Set ......... A-58
     DEC Publishing Character Set spec (not ISO draft)
A.3.11 11 DDIS DEC Output Rendering Character Set ...... A-62
     DEC spec (no ISO draft in progress yet)
A.3.12 12 DDIS DEC Special Graphics (VT100 Line Drawing) Character Set .... A-67
     DEC VT100
A.4 TEXT16_STRING - 16-BIT CHARACTER SETS .......... A-70
     0 Reserved
A.4.3 1 DDIS JIS Japanese Character Set ............. A-72
     JIS 6226-1983 (not 1978)
A.4.4 2 DDIS GB Chinese Character Set .............. A-93
     GB 2312-1980
<table>
<thead>
<tr>
<th>Latin 1</th>
<th>tag</th>
<th>length</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>+</td>
</tr>
<tr>
<td>Greek</td>
<td></td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Latin 1</td>
<td>9</td>
<td></td>
<td>resis</td>
</tr>
</tbody>
</table>
Principle of Uniqueness

Each character appears in only one DD1S set.

If DD1S set was drawn from a standard, duplicate characters are omitted.

Example:
Latin-2 has entire left hand and many right hand characters duplicated. :: dropped.
TEXT16 has no Latin characters.

Issue - some say applications have to convert from DD1S to internal inorder to process. :: could hold duplicates.
Table 1. DEC Multinational Character Set

<table>
<thead>
<tr>
<th>ASCII Control Set</th>
<th>ASCII Graphic Character Set</th>
<th>Add'l Control Set</th>
<th>DEC Supplemental Graphic Set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes

Empty positions are reserved for future standardization.
## CODE TABLE

### ISO Latin Alphabet Nr 1
ISO 8859/1

<table>
<thead>
<tr>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td>08</td>
<td>09</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>00</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td>08</td>
<td>09</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>00</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td>08</td>
<td>09</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>00</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td>08</td>
<td>09</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>00</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td>08</td>
<td>09</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
</tbody>
</table>

- **SP** (00) 0   P   .
- **&** (02) 0   E   Q   e   q   Ñ   Ñ
- **1** (03) 0   1   A   Q   a   q   Á
- **!** (04) 0   1   A   Q   a   q   Ñ
- **"** (05) 0   1   A   Q   a   q   Ñ
- **#** (06) 0   1   A   Q   a   q   Ñ
- **$** (07) 0   1   A   Q   a   q   Ñ
- **%** (08) 0   1   A   Q   a   q   Ñ
- **&** (09) 0   1   A   Q   a   q   Ñ
- ***** (10) 0   1   A   Q   a   q   Ñ
- **+** (11) 0   1   A   Q   a   q   Ñ
- **,** (12) 0   1   A   Q   a   q   Ñ
- **-** (13) 0   1   A   Q   a   q   Ñ
- **.** (14) 0   1   A   Q   a   q   Ñ
- **?** (15) 0   1   A   Q   a   q   Ñ

**HYPHEN**

---

April 1986
### Table 3 - Primary and supplementary sets of graphic characters for text communication coding when represented by code combinations 2/1 to 7/14 and 10/1 to 15/14 of an 8-bit code

#### ISO 6937/2 Text Communication

<table>
<thead>
<tr>
<th>Code</th>
<th>Character</th>
<th>Code</th>
<th>Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000 00</td>
<td>0</td>
<td>a</td>
<td>P</td>
</tr>
<tr>
<td>0001 01</td>
<td>!</td>
<td>A</td>
<td>Q</td>
</tr>
<tr>
<td>0010 02</td>
<td>&quot;</td>
<td>B</td>
<td>R</td>
</tr>
<tr>
<td>0011 03</td>
<td>#</td>
<td>C</td>
<td>S</td>
</tr>
<tr>
<td>0100 04</td>
<td>§</td>
<td>D</td>
<td>T</td>
</tr>
<tr>
<td>0101 05</td>
<td>%</td>
<td>E</td>
<td>U</td>
</tr>
<tr>
<td>0110 06</td>
<td>&amp;</td>
<td>F</td>
<td>V</td>
</tr>
<tr>
<td>0111 07</td>
<td>'</td>
<td>G</td>
<td>W</td>
</tr>
<tr>
<td>1000 08</td>
<td>(</td>
<td>H</td>
<td>X</td>
</tr>
<tr>
<td>1001 09</td>
<td>)</td>
<td>I</td>
<td>Y</td>
</tr>
<tr>
<td>1010 10</td>
<td>*</td>
<td>J</td>
<td>Z</td>
</tr>
<tr>
<td>1011 11</td>
<td>+</td>
<td>K</td>
<td>k</td>
</tr>
<tr>
<td>1100 12</td>
<td>,</td>
<td>L</td>
<td>l</td>
</tr>
<tr>
<td>1101 13</td>
<td>-</td>
<td>M</td>
<td>j</td>
</tr>
<tr>
<td>1110 14</td>
<td>.</td>
<td>N</td>
<td>n</td>
</tr>
<tr>
<td>1111 15</td>
<td>/</td>
<td>O</td>
<td>o</td>
</tr>
</tbody>
</table>

*See 4.3.4*
DDIS & DDIF

ISO LATIN1 instead of DE MCS as fundamental character set

LATIN1_STRING (universal code 20)

need video fonts - 15 more glyphs
need UIS compose sequence input

March DDIS specifies ISO latin1, but Sarah Field test still DE MCS
ISO Latin 1 adds:

- NBSRN: NOBREAK SPACE
- BRR: BROKEN BAR
- DIA: DIAERESIS
- HY: HYPHEN
- REG: REGISTERED
- MAC: MACRON
- ACC: ACUTE ACCENT
- CED: CEDILLA
- THQ: THREE QUARTERS
- ETH: "O THORN
- YAC: Y WITH ACUTE ACCENT
- TIM: TIMES
- DIV: DIVIDE
- LOG: LOGICAL NOT

ISO Latin 1 drops:

- OE
- OYG

ISO Latin 1 changes:

- Y
DDIS Two Byte

2nd byte

1st byte

JIS 6226

with duplicates removed

2nd byte

1st byte

GB 2312

with duplicates removed
International 2, > 2 byte standards

ISO TC97/SC2/WG2 committee

only two byte scope

DEC wants greater than two byte for China, Japan

China > 50,000 characters

Japan 16,000 - 50,000

Taiwan

Hong Kong

Korea

etc.
Current ISO Two Byte Thinking:

```
<table>
<thead>
<tr>
<th></th>
<th>0 31</th>
<th>2nd byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>031</td>
<td>China</td>
<td>non-idea</td>
</tr>
<tr>
<td>1st 127 byte</td>
<td>idea</td>
<td></td>
</tr>
<tr>
<td>255</td>
<td></td>
<td>Idea</td>
</tr>
</tbody>
</table>
```

Non-idea
Status of Graphics and Document Standards

J. Schönhut
Disclaimer:

The information given does not represent any official position of ISO nor of its computer graphics working group ISO TC97/SC21/WG2. The author is chairing that working group; nevertheless this is purely the personal opinion and understanding of the author.
1. Overview and Introduction
2. Standards Making Bodies
3. Status of Standards Documents
4. Graphics Standards
   4.1 Graphical Kernel System
   4.2 Graphical Kernel System - Transition to Three Dimensions
   4.3 GKS Language Bindings
   4.4 Programmers Hierarchical Interactive Graphics Standard (PHIGS)
   4.5 Computer Graphics Metafile
   4.6 Computer Graphics Interface
   4.7 Conformance Testing of Computer Graphics Standards
   4.8 Formal Specification of Computer Graphics Standards
   4.9 Window/Terminal Management Standards
   4.10 Graphics in Documents
   4.11 Graphical Data Syntax / Videotex (ECMA/CEPT)
5. Towards a Reference Model for Computer Graphics Standards
1. Overview and Introduction

Computer Graphics Industry + 20 percent/year
- reduction of hardware prices
- increased complexity of applications
- main cost factor: software

Demand for standardization

not "industry standards"

standards by standards making bodies
2. Standards Making Bodies
Bodies involved in Graphics Standards

ISO - International Standardization Organization
ISO TC97/SC21/WG2 Computer Graphics

Member bodies: national standardization organizations
     other international organizations

DIN (Fed. Rep. of Germany)
ANSI (USA)
BSI (United Kingdom)
AFNOR (France)
NNI (Netherlands)
CSA (Canada)

IFIP (Interntl. Fed. for Information Processing)
ECMA (European Computer Manufacturers Assoc.)
Example: GKS
first developed by DIN NI 5.9 (now NI 21.2);
brought to ISO by DIN
international review by ISO TC97/SC21/WG2
result: ISO 7942

Operation Principle:
Consensus rather than mere decisions by voting
requires time (plus time for voting procedures)
normally ensures good quality

Other groups (esp. videotex):
ECMA
CEPT
time frame shorter -> earlier results
work based on work of ISO TC97/SC21/WG2
cooperation
3. Status of Standards Documents

Refs. to standards often unqualified
status of document important
status shows relative utility of document

ISO technical work structured in 3 levels
- TC (Technical Committee)
- SC (Sub-Committee)
- WG (Working Group)

e.g.

TC97 Information Processing
TC97/SC21 Open Systems (short: SC21)
TC97/SC21/WG2 Computer Graphics (short: WG2)
Procedural Steps
- New Work Item (NWI) approved by TC letter ballot commitment necessary
- NWI assigned to SC or ...
  WG does technical work
- Initial draft developed
- Working Document (visible to SC for information)
- DP Registration (SC registration ballot/SC meeting)
- DP ballot (SC ballot - technical comments)
- multiple DP's possible
- DIS Registration (out of DP ballot)
- DIS ballot (TC ballot - no technical changes)
- multiple DIS's possible
- Final Text for IS to ISO Central Secretariat Geneva
- Publication provided ISO Council accepts
- IS
4. Graphics Standards

Characteristics of standards
and standards under development

Status of projects
4.1 Graphical Kernel System

Graphical Kernel System (GKS) ISO 7942
functional standard
one or more workstation
passive and interactive graphics
two dimensional graphics
device independent attribute setting
one level of picture segments
storing, retrieving and interpreting picture
information via GKS Metafile (GKSM)
inquire functions
error handling concept
4.2 Graphical Kernel System - Extension to 3D

Extension of two dimensional GKS to three dimensions
upward compatible to GKS
GKS program runs unmodified under GKS-3D
one level of segmentation

Status: DIS 8805
4.3 GKS Language Bindings

GKS as functional specification needs embedding into programming languages by language bindings. Many languages not only ISO programming languages.

GKS Language Bindings for

FORTRAN
Pascal
Ada

by WG2 in close operation with WG of specific langua.
If no extension of language (e.g. FORTRAN subroutines) lead is with WG2

In case of integration into language lead is with WG dealing with language (e.g. BASIC)

Status: FORTRAN, Pascal, Ada DP 8651 parts 1, 2 and 3
        FORTRAN and Pascal going into DTS processing
        Ada goint to 2nd DP 8651/3
        BASIC currently dpANS
        C outside ISO, progress of a C binding closed
        monitored by WG2
4.4 Programmer's Hierarchical Interactive Graphics Standard (PHIGS)

response to need for
hierarchically structured pictures

GKS: one level of segmentation
PHIGS: hierarchical segment structures

editing of such structures
upward compatible to GKS wherever possible
most critical and difficult issue in actual graphics standardization work
4.5 Computer Graphics Metafile

GKS already contains functionality of GKSM formal status ambiguous not integral part of standard in ISO integral part of standard in e.g. DIN

Formal discrepancy: Need for International Standard for graphics metafile or set of metafiles
Computer Graphics Metafile (CGM) formerly known as VDM

Four part standard
functional specification
three parts data encodings
no segmentation in current version
possible, however not easy, use of CGM as a GKS for level 0 GKS
full GKS not met, but work is under way

Status: DIS 8632/1-4
4.6 Computer Graphics Interface

Interface to graphical devices

New Work Item Computer Graphics Interface (CGI)
formerly known as Virtual Device Interface

Major Issues:

Compatibility with CGM and GKS
Full support for GKS workstations
4.7 Conformance Testing of Computer Graphics Standards

After feasibility study request for New Work Item on Conformity Testing

If New Work Item accepted ISO Technical Report (TR)
4.8 Formal Specification of Computer Graphics Standards

Natural language descriptions:
sometimes difficult with correct
or intended interpretation
wish to formally specifying standards
one problem: poor readability of
formally specified standards
for non formal specs trained reader
possible solution: natural language text
along with a formal description,
in dubio pro formal specification

After feasibility study request for New Work Item on
Use of Formal Specification Techniques for
Computer Graphics Standards
If NWI is accepted, ISO Technical Report (TR)
4.9 Terminal/Window Management Standard

Fast expanding field: use of bit mapped displays overlapping window systems
Progress monitored by WG2
Possibility of standard Window Management investigated by ANSC X3H3

OSI context: virtual terminals virtual graphics terminals terminal management issues across all virtual terminals

Combination: window and terminal Management
4.10 Graphics in Documents

Graphics in Documents: growing importance
ISO TC97/SC18
WG2 experts participating
Proposals for integrating graphics in
ODA/ODIF
SGML
by WG2

Status of Document Standards:
ODA/ODIF     DIS 8613/1-6 (Integration into
            CCITT T.73 expected)
SGML         DIS 8879
4.11 Graphical Data Syntax/Videotex (ECMA/CEPT)

ECMA GDS Standard
based on GKS functionality
multi workstation interface

CEPT Videotex Standard
subset for output only from GDS
to form geometrical parts of
Videotex Standard.
5. Towards a Reference Model for Computer Graphics Standards

Relation of graphics standards projects among each other and to the outside world (OSI, Documents, Codings, Programming Languages)

Issue of compatibility
Reference Model (RM) for Computer Graphics:

Some questions:
- difference between Codings and Language Bindings
- Interfaces to be identified in the RM
- relation of Functional Standards (GKS, GKS-3D, PHIGS) to CGI (and CGM/GKSM)
- Structuring Concepts to be used in Graphics Standards (level vs. option sets, dimensionality, etc.)
- concepts for Attribute Binding.

Difficult, but badly needed

Current situation: everyone has a RM,
but each one is a bit different
Compatibility of Graphics Standards

J. Schönhut
Compatibility Problems

- GKS vs. PHIGS

- GKS vs. CGM
**PHI-GKS**

may be seen as:

1. a shell on top of PHIGS

2. a superset of GKS-3D and PHIGS

3. the current work item PHIGS (with some changes)

4. a GKS-3D extension with hierarchy and editing
PHI-GKS Summary

- level structure
- workstation control
- state diagram
- transformation pipeline
- hierarchical data structures
- attribute model
- primitives outside segments
- archiving
- deferral mode
- state lists and description tables
PHI-GKS

Level Structure

- levels as in GKS (output, input)
- dimensionality (2D/3D)
- additional output levels for hierarchy and editing
The state list is described by triple
\[ \{ \text{WS-S}, \text{SEG-S}, \text{ARC-S} \} \]
with the following state values:

- workstation state
  \( (\text{PHI-GKS\_OPEN}, \text{WSOP}, \text{WSAC}) \)

- segment state
  \( (\text{SEG\_CL}, \text{SEG\_OP}) \)

- archive state
  \( (\text{ARC\_CL}, \text{ARC\_OP}) \)
PHI-GKS

Workstation Control

- workstation independent segment structure vs. structure store

- WISS always active

- associate, insert, copy

- archival
PHI-GKS

Transformation Pipeline

- workstation dependent transformations
  - viewing transformation
  - workstation transformation

- workstation independent transformation
  - modelling transformation
    - global
    - local
  - normalization transformation
  - normalization clip
  - segment transformation
PHI-GKS

Hierarchy & Editing

- segments vs. structures
  - different data types
  - one data type

- segment attributes
  - segment header
  - structure elements

- 2D/3D - structure elements

- hierarchy of segments
  - execute segment
  - existence of (dummy) segments

- editing
PHI-GKS

Attribute Model

- individual/bundled & ASFs
- explicit: set modal attr. functions
- generation vs. traversal time binding
  - new attr. value: 'to be inherited' (tbi)
- PHI-GKS state list
- edit state list
- at segment creation: segment header from PHI-GKS state list
- binding:
  outside segments: use PHI-GKS state list values
  segment open: use edit state list
  at segment creation:
    values of Edit State List and PHI-GKS State List identical
- Traversal State List
PHI-GKS

- primitives outside segments
- metafile / archiving
- deferral / update states
  - deferral mode
  - implicit regeneration
- state lists
  - PHI-GKS State List
  - Workstation State List
  - Edit State List, in addition
  - Traversal State List (internal)
- description tables
structured

flat

PHIGKS

PHIGS

GKS/GKS-3D

2

1
structure  segment
Hierarchical Segments
< set polyline index ("to be inherited") >

create segment (a)
polyline
set polyline index 2
polyline
set polyline index ("to be inherited")
polyline
close segment

create segment (b)
set polyline index 1
execute (a) <1.>
set polyline index 3
execute (a) <2.>
close segment

post (b)
<table>
<thead>
<tr>
<th>function</th>
<th>Edit state list</th>
<th>PHI-GKS state list</th>
</tr>
</thead>
<tbody>
<tr>
<td>open (create)</td>
<td>copied from PHI-GKS state list</td>
<td>set values</td>
</tr>
<tr>
<td>set attribute</td>
<td>set values</td>
<td>set values</td>
</tr>
<tr>
<td>re-open</td>
<td>set from segment header + scan to edit position (set values)</td>
<td></td>
</tr>
</tbody>
</table>
PHI-GKS

Workstation Control

- workstation independent segment structure vs. structure store

- WISS always active

- associate, insert, copy

- archival
# Problems between GKS and CGM

<table>
<thead>
<tr>
<th>GKS</th>
<th>CGM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EXT</strong></td>
<td><strong>TEXT</strong></td>
</tr>
<tr>
<td>- point</td>
<td>- point</td>
</tr>
<tr>
<td>- string</td>
<td>- append flag</td>
</tr>
<tr>
<td><strong>ELL ARRAY</strong></td>
<td><strong>CELL ARRAY</strong></td>
</tr>
<tr>
<td>- corner points</td>
<td>- corner points</td>
</tr>
<tr>
<td>- nx, ny</td>
<td>- nx, ny</td>
</tr>
<tr>
<td>- colour array</td>
<td>- local colour precision</td>
</tr>
<tr>
<td><strong>ET PATTERN REPRESENTATION</strong></td>
<td><strong>PATTERN TABLE</strong></td>
</tr>
<tr>
<td>- WS identifier</td>
<td>- pattern index</td>
</tr>
<tr>
<td>- pattern index</td>
<td>- nx, ny</td>
</tr>
<tr>
<td>- nx, ny</td>
<td>- local colour precision</td>
</tr>
<tr>
<td>- colour array</td>
<td>- colour array</td>
</tr>
</tbody>
</table>

*Source: FhG AGD*
## Problems between GKS and CGM

<table>
<thead>
<tr>
<th>GKS</th>
<th>CGM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SET TEXT FONT AND PRECISION</strong></td>
<td>TEXT FONT INDEX</td>
</tr>
<tr>
<td></td>
<td>TEXT PRECISION</td>
</tr>
<tr>
<td><strong>SET FILL AREA INTERIOR STYLE</strong></td>
<td>HATCH INDEX</td>
</tr>
<tr>
<td></td>
<td>PATTERN INDEX</td>
</tr>
<tr>
<td><strong>SET ASPECT SOURCE FLAGS</strong></td>
<td>ASPECT SOURCE FLAGS</td>
</tr>
<tr>
<td>- 13 flags</td>
<td>- $n$ (ASF type; ASF value)</td>
</tr>
<tr>
<td><strong>GDP</strong></td>
<td>GDP</td>
</tr>
<tr>
<td>systematic use of a GDP identifier</td>
<td>no GDP identifier for standardized GDPS</td>
</tr>
</tbody>
</table>
PDC

Picture Description Call Interface
- A device independent graphics interface to DDIF/LAL

Paul H. Wong
10th July, 1986
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

Introduction
(0)

PRESENTER    PAUL H. WONG
             PIXEL::PWONG

GROUP        Document Processing Systems / Graphics

FORMER PROJECTS    VAX DECgraph / VAX DECide

CURRENT PROJECTS    PDC / CHARTER / XPRESS in WPSplus V3.0 (GOLD)
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

Agenda
(1) AGENDA
(2) WHY PDC?
(3) WHY TREE STRUCTURE?
(4) ORIGINAL PDC ARCHITECTURE
(5) CURRENT PDC/LAL ARCHITECTURE
(6) KEY FUNCTIONS OF PDC
(7) PDC FRONT END TO THE LAL
(8) PDC PRIMITIVES
(9) EXPERIENCE FROM THE GOLD PROTOTYPE
(10) QUESTIONS?
XPRESS and CHARTER

Needed a device- and operating environment-independent graphics interface for the VAX and the PC (Rainbow at that time).

Wanted to produce the same output file format for interchange purposes.

Wanted a hierarchical data structure for picture description:

- to have full tree structures like DECslide's linked list structure
- to use the tree structure for inheritance and generic references.
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

Why Tree Structure?

(3)

TYPICAL PDC TREE
**Picture Description Call Interface**

A device independent graphics interface to DDIF/LAL

Original PDC Architecture

1. APPLICATION
   - XPRESS, CHARTER
2. Hierarchy
   - HPDC
   - Tree Manipulation
3. Device Independence
   - IPDC
   - Th. Layer + GK / Basingstoke
4. LPDC
   - ReGIS / SIXEL
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

Current PDC/LAL Architecture (5)
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

Key Functions of PDC

KEY FUNCTIONS

Inquire / Manipulate Primitives

Inquire / Manipulate Attributes

Traverse the LAL tree, passing data and attribute information to output agents

Issue MS-Windows calls

Issue PostScript calls
FRONT END TO LAL
PDC currently provides a front end to LAL to:
- Walk (traverse) a sub-tree
- Copy a sub-tree
- Delete a sub-tree
- Move a sub-tree

VOLATILE STATUS
These functions form a logical layer so we can develop code independent of the LAL schedule. This layer might vanish. If that happens, the application will make calls directly to the LAL.
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

PDC Primitives
(8)

USAGE OF PRIMITIVES
PDC deals with Graphics Primitives in 4 ways:

- It processes them and puts them onto the LAL tree
- It edits them on the LAL tree
- It displays them in MS-Windows
- It translates them into PostScript code

LIST OF PRIMITIVES
- ARC
- CIRCLE
- ELLIPSE
- PIE
- ELLIPTICAL ARC
- ELLIPTICAL PIE
- MULTIPOINT
- SPLINE
IMPORT SARAH GRAPHICS

We successfully read in the Sarah "Helicopter" DDIF file with over 1000 line segments and were able to display it in PostScript and edit it with XPRESS.

Here are the steps we took:
- The Sarah prototype generated a DDIF file
- File format was translated to KODDIF format
- PDC used KODDIF to read the file into a PDC tree.

Note that the imported image was not exactly the same as the original Sarah one because the PDC prototype chose to ignore certain attributes in the DDIF file due to schedule constraints.

WRITING PORTABLE CODE

All the POC, XPRESS and HARTER code was written in C in a portable manner. The prototype ran on the VAX and the Rainbow with no major difficulties.

FILE SIZES

We did some file size polling for the prototype. Here's a comparison of relative file sizes:
- DECslide file - 100%
- PDC flat file - <50%
- ASCII DDIF file - >300%
- Binary DDIF file - <50%
Picture Description Call Interface
A device independent graphics interface to DDIF/LAL

Questions?

(10)
Interfaces and Data Formats for Transfer and Communication in Computer Graphics Systems

J. Encarnacao

J. Schönhut
Graphische und Geometrie-Normen
für Graphische Systeme
und ihre Anwendungen
Text Models

J. Schönhut
Text Models in Documents and Graphics

3 basic Models:

- Text & Graphics side by side
  "graphics" text & "text" text

- everything Text:
  Mosaic Graphics only

- everything Graphics:
  should simplify situation

BUT
<table>
<thead>
<tr>
<th>Attributes</th>
<th>DDIF</th>
<th>GKS</th>
<th>CEPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background (color)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Font</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rendition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tab Stops</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wrap Format</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quad Format</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Spacing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position (horiz./vert. * abs/rel)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Precision</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left Margin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right Margin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top Margin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bottom Margin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vertical Alignment</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horizontal Alignment</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Up Vector</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spacing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expansion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attributes (continued)</td>
<td>DDIF</td>
<td>GKS</td>
<td>CEPT</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------</td>
<td>-----</td>
<td>------</td>
</tr>
<tr>
<td>Underline</td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Size (normal, double height,</td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>double width, double)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flash</td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Conceal</td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Invert</td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Marked (for further action)</td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Protected</td>
<td></td>
<td></td>
<td>*</td>
</tr>
</tbody>
</table>
Textprocessing:

string --> break into resp. catenate multiple lines, hyphenation, blocks

a. 1D =====> b. 2D
processable

Graphics:

string --> exact positioning of strings, no modification

c. 2D or even 3D =====> d. 2D
processable

Currently there is

NO simple way of mapping b. to c. in general case except if breaking it down to single characters.
FhG-AGD

Functional and Data Interfaces
Presentation Graphics Layer on Top of GKS

Interface GKS - Presentation Graphics Package
- primitives
- data structures
- cluster/levels

Interface to the Environment
- operating system
- language binding
- interface to data handling utilities
- interface to methods handling utilities

Operator Interface
- passive/interactive
- dialog
- interaction techniques
GKS in a Network Environment

WSI of GKS opens possibility of using GKS in a network environment

GKS oriented communication protocol is based on services supplied by the T.70 transport protocol

DFN
Product Definition Data

- STEP
- PDES
- SET
- MAP Manufacturing Automation Protocol (General Motors)
- EDIF Electronic Design Interchange Format
Graphics in Documents

Two Standards:
- Standard General Markup Language SGML
- Office Document Architecture/Office Document Interchange Format ODA/ODIF

- logical and layout structure
- processible and image form
- integrate graphics in form of graphics metafiles
- compound documents
IGES

IGES includes three entity types:
- geometry (point, line, circle ...)
- dimensioning/annotation (label ...)
- structure (drawing, font and view relationships)

IGES history
- NBS standard
- ANSI standard (immature & age)
VDAFS

Geometric entities:
- point
- point set
- point vector set
- composite curve (include parametric splines)
- parametric spline surface
Videotex

- alpha-mosaic character graphics
- scanned image facsimile mode graphics
- geometrically encoded graphics

output subset of GDS as Videotex Standard

Geometric Encoding

NAPLPS
- not efficient encoding
- poor functionality
IGES

IGES file contains:
- Prolog Section
- Global Section
- Directory Entry Section
- Parameter Data Section
- Terminator Section

IGES file concept is strongly directed to the "transfer of drawings"
CGM

- descriptor elements
- control elements
- picture descriptor elements
- graphical elements
- attribute elements
- escape elements
- external elements

used as GKS Level 0 Metafi
Device & Wkst Interfaces

Workstation Interface (WSI)
- Separation of
  - GKS Kernel
  - GKS Workstation

CGI (ISO)

ECMA GDS
- GKS functionality
- adaption of the output subset by CEPT
Functional Standards

- GKS (ISO 7942)
- GKS-3D
- PHIGS
- PHI-GKS

Migration Issue

- from GKS to GKS-3D
- from GKS-3D to PHI-GKS
- from GKS-3D to PHIGS
Graphics Metafiles

 Ideally: formats for storing transmitting pictures device application independent

GKSM

GKSM contains:
- file/picture header
- end item
- control items
- output primitive items
- attribute items
- non graphical items (user items)

GKSM: input & output workstation
Data Interfaces

- GKS, PHIGS, PHI-GKS & Language Bindings
- CEPT Videotex (GDS subset)
- CGI (VDI)
- CGM (VDM)
- US Videotex Interface NAPLPS
- IGES, VDAFS, SET, STEP
- SGML & ODA/ODIF
- Presentation Graphics (based on GKS)

GKS Impact

- VLSI support of GKS implementations
- GKS based applications for LANs and WANs
Computer Graphics Standards

Milestones

1974          GSPC by ACM SIGGRAPH
1975          DIN
1976          SEILLAC I (Methodology in Computer Graphics)
Graphics System

- Operator (User)
- Graphics Support System (Services)
- Other User Interface Support System (Services)
- Application Functions
- Generic Action Routines
- Data Base
Graphics System

- Application
  - Control
  - Heuristics

- Model
  - Model Metafiles (Design Data)
  - Model Data Management

- Graphics
  - Graphics Viewing
  - Request Processing
  - Graphics Metafiles

- User (Operator of the Computer-Aided Environment based on a Graphics Support)
Comparison of DDIF, SGML and ODA/ODIF

J. Schönhut
The House of Nicolaus
Document Descriptor

Major_version 0 (?)
Minor_version 11 (?)
Application_id 2×15 (not registered)
Product_name "SlideX"

Document Profile

Title "DDIF Example"
Author "FhG-AGD Daun/Puchtler/Schoenhut"
Version "0.00"
Date "22-May-1986 09:23:23.00"
Fonts fontfile (?)
Document_Header

Page_LayoutDefs

Page_Desc_Name "Testpage"

Blocks

Block_Desc

Lower_Left 944,944 (2cm, 2cm)

Upper_Right 472,9440 (1cm, 20cm)

Block_Content

Begin_Segment

Segment_Id "Fhg Logo"

TEXT8_Content "FhG-AGD"

End_Segment

G-AGD DDIF Example page 2
Frame_Position
   Frame_X_Position
      X_Rel  2  (frame_x_center)
   Frame_Y_Position
      Y_Rel  0  (Frame_y_centered)
Top_Space  472  (1cm)
Quadrant  1  (Origin lower left) OPTIONAL
Frame_Transform
   Sx  378  (coordinates in range of)
   Sy  378  (0 to 10 scaled to full frame)
Color_Map
   1  0  0  0  0  (background black)
   1  1  1  1  1  (foreground white)
   1  1  0  0  0  (pure red for house)
Line_attributes
   Line_Color  2
Multipoint
   Points   3, 2, 7, 6, 5, 8, 3, 6, 7, 2, 3, 2, 3, 6, 7, 6, 7, 2
End_Segment
Begin_Segment
Local_Attributes
   Segment_Bindings
      Registered_Tag   19   (Single_Text_Line_tag)
Frame_Attributes
   Frame_Height   472 (1cm)
   Frame_Width    3780 (8cm)
   Frame_Position
      Frame_X_Position
         X_Rel   2   (frame_x_center)
      Frame_Y_Position
         Y_Rel   0   (Frame_y_centered)
   Top_Space    472 (1cm)
   Bottom_Space 944 (2cm)
EXT8_Content  "The House of Nicolaus"

EXT8_Content  "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB...B"

EXT8_Content  "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB...B"
ELEMENT P -- (#RCDATA) >

ELEMENT

GRAPHIC - 0 ((CGM | CGMREF), SUBTITLE?)

SIZEX NUMBER REQUIRED

SIZEY NUMBER REQUIRED

POSX (FLOAT | FIXED) FLOAT

POSY (FLOAT | FIXED) FLOAT

X0 NAME UNKNOWN

Y0 NAME UNKNOWN

X1 NAME UNKNOWN

Y1 NAME UNKNOWN

XALIGN (CENTER | LEFT | RIGHT) CENTER

YALIGN (CENTER | TOP | BOTTOM) CENTER

SCMODE (UNIFORM | NONUNIF) UNIFORM

ORIENT (0 | 90 | 180 | 270) 0

PICNUM NUMBER "1"

FRAME (FRAME | NOFRAME) NOFRAME >
<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEMENT CGM</td>
<td>- -</td>
<td>(#CDATA)</td>
</tr>
<tr>
<td>CODING</td>
<td>CLEAR</td>
<td>CHAR</td>
</tr>
<tr>
<td>LEMENT BANNER</td>
<td>- O</td>
<td>NULL</td>
</tr>
<tr>
<td>TEXT</td>
<td>(#CDATA)</td>
<td></td>
</tr>
<tr>
<td>LEMENT CGMREF</td>
<td>- O</td>
<td>NONE</td>
</tr>
<tr>
<td>FILE</td>
<td>CONREF</td>
<td>REQUIRED</td>
</tr>
<tr>
<td>CODING</td>
<td>CLEAR</td>
<td>CHAR</td>
</tr>
<tr>
<td>LEMENT SUBTITLE</td>
<td>- O</td>
<td>(#RCDATA)</td>
</tr>
</tbody>
</table>
<element>
  <element>
    <element>
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA...A
    </element>
  </element>
</element>

<element>
  <element>
    <element>
      GMF "NICOLAUS";
    </element>
    <element>
      VERSION 1;
    </element>
    <element>
      EM LIST "DRAWINGPLUS";
    </element>
    <element>
      CTY PE REAL
    </element>
    <element>
      GPIC OFF, 1, "BILD 1";
    </element>
    <element>
      GPIC B O D Y;
    </element>
    <element>
      CLR TABLE 1, (1, 0, 0);
    </element>
    <element>
      T LINE COLOR 1;
    </element>
  </element>
</element>
<table>
<thead>
<tr>
<th>ct Type</th>
<th>DOCUMENT LOGICAL ROOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct Identifier</td>
<td>1</td>
</tr>
<tr>
<td>Visible Name</td>
<td>&quot;Testpage&quot;</td>
</tr>
<tr>
<td>Coordinates</td>
<td>0,1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ct Type</th>
<th>BASIC LOGICAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct Identifier</td>
<td>1 0</td>
</tr>
<tr>
<td>Visible Name</td>
<td>&quot;FhG Logo&quot;</td>
</tr>
<tr>
<td>Portion</td>
<td>0</td>
</tr>
</tbody>
</table>

| Portion Identifier | 1 0 0 |
| Portion | "FhG-AGD" |
pection Type
pect Identifier 1 1
r Visible Name "Body"
ordinates 0, 1, 2
pection Type
pect Identifier 1 1 0
r Visible Name "Paragraph A"
presentation Style 2
ent Portion 0
ent Portion Identifier 1 1 0 0
ent "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA... A"

G-AGD  ODA Specific Logical Structure page 2
<table>
<thead>
<tr>
<th>Extent Type</th>
<th>COMPOSITE LOGICAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent Identifier</td>
<td>1 1 1</td>
</tr>
<tr>
<td>Extent Visible Name</td>
<td>&quot;Figure&quot;</td>
</tr>
<tr>
<td>Extent Coordinates</td>
<td>0, 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Extent Type</th>
<th>BASIC LOGICAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent Identifier</td>
<td>1 1 1 0</td>
</tr>
<tr>
<td>Extent Visible Name</td>
<td>&quot;Drawing&quot;</td>
</tr>
<tr>
<td>Extent Presentation Style</td>
<td>0</td>
</tr>
<tr>
<td>Extent Portion</td>
<td>0</td>
</tr>
<tr>
<td>Extent Portion Identifier</td>
<td>1 1 1 0 ^</td>
</tr>
<tr>
<td>Extent</td>
<td>/* sequence of geometric elements */</td>
</tr>
<tr>
<td>Struct Type</td>
<td>BASIC LOGICAL</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Struct Identifier</td>
<td>1111</td>
</tr>
<tr>
<td>Visible Name</td>
<td>&quot;Caption&quot;</td>
</tr>
<tr>
<td>Presentation Style</td>
<td>1</td>
</tr>
<tr>
<td>Portion</td>
<td>0</td>
</tr>
<tr>
<td>Portion Identifier</td>
<td>11110</td>
</tr>
<tr>
<td>Port</td>
<td>&quot;House of Nicolaus&quot;</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Object Type</td>
<td>BASIC LOGICAL</td>
</tr>
<tr>
<td>Object Identifier</td>
<td>112</td>
</tr>
<tr>
<td>Visible Name</td>
<td>&quot;Paragraph B&quot;</td>
</tr>
<tr>
<td>Presentation Style</td>
<td>2</td>
</tr>
<tr>
<td>Portion</td>
<td>0</td>
</tr>
<tr>
<td>Portion Identifier</td>
<td>1 1 2 0</td>
</tr>
<tr>
<td>Portion Identifier</td>
<td>&quot;BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB...B&quot;</td>
</tr>
</tbody>
</table>

G-AGD | ODA Specific Logical Structure | page 5
Presentation Style 0
Content Type Geometric Graphics

Presentation Style 1
Alignment CENTRED

Presentation Style 2
Alignment JUSTIFIED
COMPOUND DOCUMENT SYSTEMS

JULY 1986

Jim Kapadia

Workstation & Terminal SW Arch.
381-2326
ZKO2-3/Q08
ELUDOM::KAPadia
COMPOUND DOCUMENTS?

- DDIF/DDIS
- SARAH
- POSTSCRIPT
- DSRR/DOCUMENT
- WPS+
- ALLIS
- INTERLEAF
- ALL-IN-1
- FORMATTERS
- MAC TOOLS
COMPOUND DOCUMENTS

USER'S VIEW OF COMPOUND DOCUMENT SYSTEM:

• COMPOSITE DOCUMENT PROCESSING FUNCTION
  . TEXT, LINE GRAPHICS...
  . IMAGES

• OPTIONAL FUNCTIONS LIKE:
  . GRAPHING & CHARTING
  . SPREAD SHEETS
  . FORMS
  . DATA BASE QUERY
  ..... 
  ..... 

IN ESSENCE, AN INTEGRATED SET OF FUNCTIONS (applications)

• OPERATING ON MULTI-MODE DATA (text, graphics,..)

• PERFORMING IN A UNIFIED MANNER

• APPEARING TO BE ONE SYSTEM TO THE USER
WHAT IS A COMPOUND DOCUMENT?

• ORGANIZED COLLECTION OF DATA
  • TEXT, GRAPHICS, IMAGES,...
  • PLUS OTHER STRUCTURED DATA LIKE
    • CHARTS, GRAPHS, TABLES...

• FOR THE PURPOSE OF PRESENTATION TO USER

• CAPABLE OF BEING EFFICIENTLY PROCESSED
OPERATIONS ON COMPOUND DOCUMENTS

• RENDER - DISPLAY, PRINT, ...

• REVISE, EDIT, MODIFY, PROCESS, ...

• EXCHANGE, INTERCHANGE, ...

• TRANSMIT, RECEIVE, MAIL, COPY ...

• RETRIEVE & ACCESS

• .....
DEFINITIONS

- **REVISABLE FORM:**

  CD REPRESENTATION LENDING TO EFFICIENT MODIFICATION, REVISION, AND PROCESSING

  ABSTRACT RELATIONSHIPS NECESSARY FOR MODIFICATIONS ARE PRESERVED

- **FINAL FORM:**

  CD REPRESENTATION READY FOR ENDERING ON A DISPLAY DEVICE (VIDEO, PRINTER,..) FOR THE HUMAN

- **FORMATTING:**

  PROCESS BY WHICH COMPOUND DOCUMENT DATA IN A REVISABLE FORM IS TRANSFORMED INTO FINAL FORM SUITABLE FOR RENDERING FOR USER

- **DATA INTERCHANGE FORMAT:**

  REPRESENTATION OF DATA IN A FORM THAT LENDS ITSELF TO INTERCHANGE BETWEEN TWO PARTIES FOR THE PURPOSE OF INTERCHANGE AT A GIVEN LEVEL OF EXPECTATION
CHARACTERISTICS OF THE SYSTEM

• TRANSPARENT DATA INTERCHANGE BETWEEN FUNCTIONS
• UNIFIED & COHERENT USER INTERFACE TO THE SYSTEM
• SMOOTH USER TRANSITION BETWEEN FUNCTIONS
• SEAMLESS FUNCTIONALITY OF SYSTEM
• ABILITY TO ADD FUNCTIONS

ABOVE IMPLIES:

AN OPEN ENDED COMPOUND DOCUMENT SYSTEM ARCHITECTURE
SCENARIO TODAY

PRODUCTS LIKE ALLIS, SYMPHONY...  · TIP OF ICEBERG

- INFORMATION SHARING AMONG VARIOUS PEOPLE
- MULTIPLE FUNCTIONS CORRESPONDING TO NEEDS
- PRESENTATION OF MULTIMODE DATA TO USERS
## COMPOUND DOCUMENT SYSTEM
### ARCHITECTURE

<table>
<thead>
<tr>
<th>FUNCTIONAL COMPONENT</th>
<th>SOFTWARE/HARDWARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Processing</td>
<td>Creator SW</td>
</tr>
<tr>
<td></td>
<td>Editors - WYSIWYG, Batch...; Text, Graphics,</td>
</tr>
<tr>
<td></td>
<td>Spread Sheet Processors</td>
</tr>
<tr>
<td></td>
<td>Forms Processors</td>
</tr>
<tr>
<td></td>
<td>Chart Processors</td>
</tr>
<tr>
<td>Formatting</td>
<td>Print Formatters</td>
</tr>
<tr>
<td>Printing</td>
<td>Printers - Laser, Impact, ..</td>
</tr>
<tr>
<td>Display</td>
<td>Video - WSs, PCs, Terminals</td>
</tr>
<tr>
<td></td>
<td>Display System SW</td>
</tr>
<tr>
<td></td>
<td>. Graphics Subsystem</td>
</tr>
<tr>
<td></td>
<td>. Drivers</td>
</tr>
<tr>
<td></td>
<td>. PLP Translators</td>
</tr>
</tbody>
</table>

---

Kapadia 7/86
<table>
<thead>
<tr>
<th>FUNCTIONAL COMPONENT</th>
<th>SOFTWARE/HARDWARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Interchange</td>
<td>• Intra-DEC: DDIF/DDIS Format</td>
</tr>
<tr>
<td></td>
<td>Intra-vendor: IBM, Wang, ..</td>
</tr>
<tr>
<td></td>
<td>ODIF, DCA,.. Gateways</td>
</tr>
<tr>
<td>Document Distribution</td>
<td>• Networking Services</td>
</tr>
<tr>
<td></td>
<td>• Mail Services</td>
</tr>
<tr>
<td></td>
<td>• Intra-Cluster Services</td>
</tr>
<tr>
<td>Document File Storage</td>
<td>• DDIF/DDIS File Storage &amp; Access Services</td>
</tr>
<tr>
<td></td>
<td>• DDIF/DDIS File Access</td>
</tr>
<tr>
<td></td>
<td>• DDIF/DDIS File Storage</td>
</tr>
<tr>
<td>User Document Retrieval</td>
<td>• Document Access &amp; Retrieval System</td>
</tr>
<tr>
<td></td>
<td>• File Cabinets,..</td>
</tr>
<tr>
<td></td>
<td>• File Folders,..</td>
</tr>
<tr>
<td>System (OS) Support</td>
<td>• OS Utilities</td>
</tr>
<tr>
<td></td>
<td>• Copy, Mail, Type, Print,..</td>
</tr>
</tbody>
</table>
Compound Document System Architecture

- **DATA INTERCHANGE**
  - NETWORK
  - NETWORK MAIL SERVICES
  - ANOTHER SYSTEM RECEIVING/KMITEING

**DOC RETRIEVAL**

**DDIF/DDIS FILE ACCESS**

**INTERCHANGE FORMAT DDIF/DDIF**

**FORMAT XLATOR** ↔ **(ANOTHER) DOC PROCESOR**

**SYSTEM SUPPORT** ↔ **DOC PROCESSOR**

**DOC PROCESSOR** → **FORMATTING** → **FINAL FORM DOC**

**DISPLAY SYSTEM** → **PRINTING SYSTEM**
COMPOUND DOCUMENTS (WYSIWYG)
CREATION & PRINTING

[Diagram of process flow with various boxes labeled as follows:
- Creator Engine
- Video Formatting
- IN-Memory Data Storage
- Doc Processor
- DD/ODIS File Acc
- DD/ODIS Revisable Format
- Print Command
- Print Syngion
- Page-Level Layout (TOP-TO-BOTTOM)
- Transport (ETHER/Serial/Parallel)
- Final Form Xlation to FLF
- Printer (GO POST-SLIP)
]

digital

-13-
DOCUMENT PROCESSING/PRINTING (BATCH)
A LA RUNOFF, DOCUMENT

DDIF/DDNS
INTER-
CHANGE
FORMAT

MARKED-UP
DOCUMENT

CREATOR SW
Ct. EDT EDITOR

DISPLAY
(LOWER CAPABILITIES)

FORMATTER

FINAL
FORM
DOCUMENT

PREVIEWING
DEVICE

PRINTER

COMMAND
APPLICATION VIEW OF DATA

Application

-------------------------------

Compound Data Format

-------------------------------

Self Defined Data Encoding

-------------------------------

Data Storage Format

-------------------------------

Physical Data

DDIF Access Routines

DDIF

DDIS

FILES-11

RMS(VMS)

DDIS Access Routines
USER'S VIEW OF COMPOUND DATA

User

Final Form

Revisable Form

Raw Data from Domains

Presentation Level Protocol

Device Independent Final Form

Standard Data Interchange Format

Data Capture
EXAMPLE - COMPOUND DOCUMENT DOMAINS

DOCUMENT
Text, Graphics, Images

CHART/GRAPH

WYSIWYG editor
ala SARAH

TABLES

GRAPHER/CHARTER
ala DECgraph

FORM

SPREADSHEETS
ala DECcalac

FORMS PACKAGE
ala VAXforms

USER INTERFACE

digital
-18-

Kapadia 7/86
ISSUES

• ARCHITECTURE

• PRODUCT STRATEGY

• PRODUCT PLAN
ARCHITECTURE

- NEED AN OPEN SYSTEM ARCHITECTURE
- A MUST FOR THE LONG TERM
- FOUNDATION BLOCKS IN OR BEING PUT IN PLACE
- OPEN ARCHITECTURE - DEC'S EDGE OVER COMPETITION
- DIAF - ARCHITECTURE REVIEW PROCESS NEEDED
- RESOURCES - NEED PARTICIPATION & HELP
STATUS

- FORMING DIAF - MEMO/CHARTER OUT
- DATA INTERCHANGE ARCHITECTURE FORUM
- OVERALL REVIEW GROUP
- OVERVIEW, ARCHITECTURE & STRATEGY DEVELOPMENT GR
- OVERVIEW & TAXONOMY IN WORKS
- SYSTEM ARCHITECTURE OVERVIEW 'N WORKS
PRODUCT STRATEGY

- WORKING ON A SSG PROPOSAL

- DEC PRODUCTS:
  - SARAH *
  - WPS V3 *
  - DECGRAPH
  - DECSLIDE
  - VAXFORMS *
  - DECPAGE
  - RUNOFF
  - DSRR *
  - DOCUMENT *
  - TEAM DATA/RALLY
  - FMS
  - ALL-IN-1
  - MAIL (which one?)
GOOD NEWS:

- SEVERAL FUNCTIONS AVAILABLE - POTENTIAL'S THERE!

BAD NEWS:

- EACH PRODUCT UNTO ITSELF
- LIMITED OR NO DATA INTERCHANGE BETWEEN FUNCTIONS
- DIFFERENT USER INTERFACES
- ROUGH SEAMS SHOWING BETWEEN FUNCTIONS
- FUNCTIONALITY MATCHING WITH USER EXPECTATIONS

- WHO IS RESPONSIBLE FOR THE SYSTEM PRODUCT?
In Summary

- DDIF/DDIS - MAJOR ROLE IN COMPOUND DOC SYSTEMS

- WILL EVOLVE BEYOND CURRENT

- NEED YOUR HELP & CO-OPERATION IN DOING THIS
SlideX:
A Graphics Interpreter in Practical Use

Dr. Juergen Schoenhut
SlideX:
An Example for Integration of Text and Graphics

J. Schönhut
SlideX - ERLGRAPH - GKS MINT
Design - Principles

Portability of the System
  FORTRAN
  Machine Independence
    (trade off with Efficiency)
  Modular Structure
  Use of Standards (GKS)
  Building Blocks
  Machine Independence of Output
  Device Independence of Output
    via GKS

User Interface
  Simple
    more complex for complex Applications
SlideX - Structure

Layer Model with SlideX, ERLGRAPH and GKS

SlideX, ERLGRAPH Application Program

Erlangen Graphics System

GKS FORTRAN Language Bindg.

Graphical Kernel System

Operating System

Other Resources Graphics Devices
GKSMINT - Metafile Interpreter
  Menu Control
  Device Adaptation of Pictures

SlideX - ERLGRAPH
  Tool Kit System for Picture Construction
  Calls with few Parameters
  Many simple Calls
  Default Settings
  All Defaults can be changed

Primitives of GKS available
  Line, Marker, Text, Fill Area, Cell Array

Attributes of GKS available
  Color, Line Width, Line Type, Fill Style

Clipping
  User defined Rectangles
  Drawing Sheet

Blanking
  Convex Polygons (→ Fill Area)
Coordinate Systems SlideX - ERLGRAPH

User Coordinates UC 2D
(cm, hard clip limits)

Problem Coordinates PC 2D
(Units defined according to Problem,
Position in UC 2D arbitrary,
Clipping possible)

User Coordinates UC 3D
(Unit of the User Volume,
Format Filling Projection by
Viewing Transformation)

Problem Coordinates PC 3D
(Units defined according to Problem,
Position in UC 3D arbitrary,

Page Concept in SlideX additional
Page - all 4 Coord. Systems
+ Line Positioning

Picture - all 4 Coord. Systems
within Rectangle within Page
SlideX - ERLGRAPH Features

2D Base Software
Drawing Sheet Administration
  Definition, Excluded Areas
Lines + Attributes
Text + Attributes
Symbols + Attributes
Fill Areas + Attributes

Typewriter Mode

Geometric Figures

High Level 2D Software
Problem Coordinate Systems
  Cartesian, Polar,
  Linear, Logarithmic
Aaxes
Curves
only from Arrays
  Contour Lines
  Cell Arrays

SlideX Extensions
Business Graphics
Struktograms
Syntax Diagrams

3D Software
Projections
Problem Coordinates
  Axes
  Area Nets (Hidden Lines)
only from Arrays
  3D Contour Lines (-"-)

June 86, Schae produced by SlideX ERLGRAPH
Slidex State Diagram

- **No page open**
  - Macro definition, all commands not generating graphics output

- **Page open**
  - All commands except END, open picture, typewriter mode

- **Picture open**
  - All commands except typewriter mode, ENDPIC, PAGE and END

- **Structogram typewriter**
  - Structogram

- **End of program**
  - End
SlideX - Typewriter Mode

PAGESIZE, width, height
PAGNR, n
SETCOD, coding
PAGE(name)
COMMENT, text
LIST
END

LMARGIN, li
ADJUST(minfill, maxfill)
IN, n

TOP - SETTOP, dist
SETLINE, dist
SETTHI, height
DOUBLE - HALF
TYPE10(lines)
ASCII
CURSIV

SETPEN, index
BLACK RED GREEN BLUE

BOTTOM - SETBOTTOM, dist
NEGFL - SETLF, fac
SETDST, fac
LARGER - SMALLER
TYPE12(lines)
DIN
NOCURSIV

BACKGROUND, index
YELLOW MAGENTA CYAN

June 86, Schoe
produced by SlideX

July 86, Schoe
SlideX - Graphics Output

Picture Window

BLOCKPIC, breite, hoehe
FREEPIC, xlu, ylu, width, hight     ENDPIC

NOFRAME ONFRAME

ERLGRAPH Commands
eXcept Drawing Sheet Definition / FORTRAN Functions

Figures (solid filled, hatched, hollow)
BOXf, xlu, ylu, xsize, ysize, rad,       bound, fill
NGONf, n, xmid, ymid, rad, strt,      bound, fill
CIRCLEf, xmid, ymid, rad,      bound, fill
ARCf, xmid, ymid, rad, strt, ar      bound, fill
ELLIPSEf, xmid, ymid, a, b, axa,      bound, fill
ELLARCF, xmid, ymid, a, b, strt, ar, axa,    bound, fill

Connections ans Labels

ARROW, aimx, aimy
ARROWN, dx, dy
BRACES, x1, y1, x2, y2, aimx, aimy
BOOMERANG, x1, y1, aimx, aimy, (arrow)
BOOMERANG2, x1, y1, x2, y2, aimx, aimy, (arrow)
BOOMRADIUS, rad
TEXTCENTER, string
TXTEXT, string
Example for SlideX - Graphics

Within a Blockpic a Rectangle and a filled Rectangle with "rounded Corners" is drawn.

```
#box, 2, 1, 3, 4
#setch, 0.2, 135
#boxf, 4, 3, 5, 4, 0.5, 1, 1
```
SlideX - Variables

DEFINE, name
DEFINE, name, dim1, dim2
SET, name, expr
APPEND, name, liste
COPY, from, to, start, end, step
CATENATE, name, string1, string2
DUMP, expr

SlideX - Functions

INT (X)  MOD (X1, X2)  RND (X)  SGN (X)
DIV (X1, X2)  MOD (X1, X2)  RND (X)  SGN (X)
ABS (X)  SQRT (X)  MAX (X1, X2)  MIN (X1, X2)
EXP (X)  LN (X)  LOG (X)
ATAN (X)  COS (X)  SIN (X)
NOT (A)  AND (A1, A2)  OR (A1, A2)
CHAR (I)  ICHAR (C)

HIGH (O)  WIDE (O)  LINE (O)  LMAR (O)  RMAR (O)
X0 (O)  XSIZE (O)  Y0 (O)  YSIZE (O)
XACT (O)  YACT (O)  XTACT (O)  YTACT (O)
PAGE (O)  PEN (O)  THI (O)  CWI (O)  LF (O)
XEXT (O)  YEXT (O)
**SlideX - Business Graphics**

**Bar Charts**

```
BarChart, title, xlu, ylu, xsize, ysize          BarEnd
Bar, value                                        Bars, array, len
LegendItem, x, y, string1..8

Label(x,y)Ax, n, ticks                         (x,y)Label, string1..10
MonthXAx, minmonth, maxmonth
Integer(x,y)Ax, min, max, ticks
Linear(x,y)Ax, min, max
Log(x,y)Ax, min, max

BarVerti                                         BarHoriz
BarRow                                           BarColumn
BarEvict                                          CmCoord
BarCoord                                          BarIndex(index)
```

June 86, Schoe

produced by
SlideX - Struktograms

Sequenz

Alternative

left

right

While Loop

Repeat Loop

Case Selection

Break
SlideX - Business Graphics

General Commands

BusiReset

BusiTitle, topflag

FillSolid

ColorIndex, index

HatchIndex, index

FillHatch

ColorMax, number

Pie Charts

PieChart, total, title

Piece, value, string

Explode, offset

NoExplode

MinExplode, offset

PieCenter, x, y

PieRadius, r

PieAngle, degrees

PieceLabel, index

PieTextEll, a, b
SlideX - Macros

Macro Definition, Invocation

MAC, macname ENDM
ENTRY, macname RETURN
DELMAC, macname
macname, (parlist) CALL, macname(parlist)

Control Structures

IF, condition, command
IFNOT, condition, command
Block-IF, condition THEN cmds (ELSE cmds) ENDIF
Block-IFNOT, condition THEN cmds (ELSE cmds) ENDIF

FOR, varname, start, end, step cmds ENDFOR
REPEAT cmds UNTIL, condition
WHILE, condition cmds ENDFOR

CASE, caseindex, labellist
GOTO, name
LABEL, name