

# Aware Biometrics Software

a complete set of development tools *for*  
standard-compliant, interoperable biometrics systems

Rob Mungovan  
Scott Hills

April 2005



 **A W A R E**




# ANSI/NIST Fingerprint Standard Update Workshop

April 26-29, 2005

*Suggestions for Changes That May Improve*

*ANSI/NIST-ITL 1-2000*

*Rob Mungovan, Aware, Inc.*



# (1) Use of UTF-8 in User Defined Fields

- We suggest this primarily as a way to better support international character sets
- UTF-8 is the 8 bit variant of Unicode
- Default variant is 16 bit
- Unicode is the common standard encoding scheme for international characters and text
- It supports most of the world's characters sets; including...
  - Ethiopic, Cherokee, Canadian Aboriginal Syllabics, Myanmar, Khmer, Mongolian and Braille.



# UTF-8 in User Defined Fields

- Current method in ANSI/NIST-ITL 1-2000 to support international character sets is as follows:
  - Described in sections 7.2.3 and 8.1.15
  - Field 1.015 “Directory of Character Sets”- a table of character sets used in the given transaction file
  - Example: index = 000, 7 bit ASCII is used, index = 002, 16 bit unicode is used
  - Within the larger file, the occurrence of the ASCII string 0x02 followed by an equal sign “=” is used to denote a new character set



# UTF-8 in User Defined Fields

- 0x02=002 This means that Unicode 16 bit data follows
  - This character set remains active until ASCII 0x03 is encountered or until the next ASCII information separator is encountered
  - This non-ASCII text must be base-64 encoded
- Problems with this scheme:
    - Overly complex
    - Not necessary
    - Base-64 encoding expands the data
    - Does anyone use it?



# Recommendation

- Use UTF-8 in place of 7 bit ASCII for all user-defined fields (not in type 1, type 4, type 10, etc).
- UTF-8 has been a common method for updating 7 bit ASCII systems to support international applications
- ASCII characters map directly to UTF-8
- This change would have no impact on existing systems or domains that do not use UTF-8.
- UTF-8 data can be read/written with existing parsers
- No syntactical changes are are required
- This scheme is in place already in certain locations



## (2) JPEG2000 support in Type 10 Records

- Field 10.11, (CGA)- Compression algorithm
  - Currently supports “NONE”, “JPEGB” (baseline JPEG) and “JPEGL” (lossless mode of JPEG)
- Recommendation:
  - We suggest the inclusion of two new entries
  - “JP2” for lossy JPEG2000 and “JP2L” for for lossless mode of JPEG2000



## (3) Software ID Field in Type 1 Record

- Recommendation
  - Use a new field, 1.016, to hold information related to the software that generated the file
  - This field could contain three strings
    - Organization name
    - Software application name
    - Version number of the software
- Background
  - This would be helpful when the source of invalid data must be identified
  - This ID concept is used by other software applications- including compression software





## (4) Formal Compression Algorithm Tables

- Record types 13, 14, 15, and 16 do not formally support any compression methods
  - The fields reserved for this information are the 11<sup>th</sup> field in each record respectively (13.011, etc)- CGA, “Compression Algorithm”
  - The only field entry officially supported is “NONE”
- Recommendation:
  - Add “JP2L”- JPEG2000 lossless for type 13 records
  - Add “WSQ” and “FPJP2” for 1000 ppi profile compliant compression: for type 14, 15, and 16 records



# Compression Algorithm Tables

- Background
  - This technique is used in the type 10 and type 8 records
  - Domains often neglect specification of this field for type 13-16 records, and incorrectly assume that it is covered by ANSI/NIST-ITL 1-2000



## (5) Type 4 CGA field Update

- Recommendation
  - Field 8 of the type 4 record should support binary 1 to signify WSQ compression is used
- Background
  - Currently binary 0 denotes no compression
  - If compression is used the Domain Registrar must maintain a list of codes for accepted compression techniques
  - FBI has standardized on binary 1 for WSQ, most domains follow this convention



Thank-you!





# Provide Standardized Method for Supporting XML Conversions

- Background:
  - Our customers sometimes wish to convert ANSI/NIST formatted data to an XML format- “NIST to XML”
  - We have supported this conversion in our software tools for several years now
  - Less frequently we are asked to support “XML to NIST”
  - This is more complicated because each Domain defines its own record and field requirements
  - Also, we have always provided the ability to read/write ANSI/NIST formatted data
    - “Use your preferred XML parser and write ANSI/NIST with our software” is what we suggested



# XML and ANSI/NIST-ITL 1-2000

- Most recently we have added XML to ANSI/NIST conversion support to our software tools
- Recommendation:
  - ANSI/NIST-ITL 1-2000 should not be converted to a pure XML format
  - The standard is very widely used and it very efficiently solves the problem it was designed to solve
    - Parsable, fully interchangeable, compact file that holds fingerprint images, facial images and biographic data
  - Guidelines or templates for converting between the standard and an XML schema could be provided in the new revision



# Domain Independent Schema

- Option
  - Provide a bridge to the XML world through a minimalist XML schema that mimics the structure of ANSI/NIST-ITL 1-2000
  - This schema elements have the format...
    - Transaction
    - Record
    - Field
    - Subfield
    - Item
  - Images can be exported as separate objects or base-64 encoded and embedded into the XML



# Domain Independent Schema

– Advantages:

- A single schema that can be used for any ANSI/NIST file
- Mirrors the structure of the ANSI/NIST file
- Users can take this “bridge schema” and use any number of available tools to parse or convert to another XML schema





# Example

```
<record type="2">
  <field number="2" name="T2_IDC">
    <subfield>
      <item>00</item>
    </subfield>
  </field>
  <field number="5" name="T2_RET">
    <subfield>
      <item>Y</item>
    </subfield>
  </field>
  <field number="6" name="T2_ATN">
    <subfield>
      <item>SA J Q SMITH, RM 4569</item>
    </subfield>
  </field>
  .
</record>
```



# Domain Specific Schema

- Option
  - Reflect domain specific requirements for TOTs, record types, and all type 2 record requirements in the element tags
  - Our solution:
    - Use an Aware devised schema that can support any ANSI/NIST domain
    - Use one of numerous conversion tools to convert between original schema and Aware domain specific schema
    - Our tools can read and convert to/from ANSI/NIST



# Domain Specific Schema

- Advantages
  - Element tag names identify field content making XML files more readable.



# Example

```
<Type2>
  <T2_IDC>
    <subfield>
      <item>00</item>
    </subfield>
  </T2_IDC>
  <T2_RET>
    <subfield>
      <item>Y</item>
    </subfield>
  </T2_RET>
  <T2_ATN>
    <subfield>
      <item>SA J Q SMITH, RM 4569</item>
    </subfield>
  .
  .
</Type2>
```



- Thank-you...