Extended DAS DTD proposal

Zhihong Zhang[1], Yuying Tian, Jing Zhao, Chengdong Zhang, Tian Xue, J. Dana Eckart[2]
Virginia Bioinformatics Institute, Blacksburg, VA 24061

Introduction

DAS/1 is designed as a simple protocol for browsing features annotated on some `entry point' sequence (typically an assembled chromosome). Currently DAS/1 is able to handle DNA and its annotation (Feature). As we use DAS more often, we may begin to ask some questions: Can DAS be made to include RNA or Protein Sequence so it can support protein/RNA structure analysis? Is there a better way to handle nested features? In cases where feature updating is necessary, such as deletion, modification and merging, will it be efficient and easy? By examining DAS/1, we have found that there are some scenarios that can't be handled or can't be handled efficiently, such as Feature hierarchy. We propose some ideas to enhance DAS/1 by extending or modifying the concepts of DAS Sequence, Group, Feature. There are five subjects in this document:

1. The combination of DAS **GFF**, DAS **SEQUENCE**, **DASEP** and **DASDSN**
2. Extend DAS **SEQUENCE**
3. DAS **GROUP**
4. DAS **FEATURE**
5. DAS **NOTE**

An additional issue that should be kept in mind is how to extend DAS/1 to handle genome/chromosome level data and gaps (un-sequenced regions). Based on the design of DAS/1, one DAS document (**SEQUENCE** or **SEGMENT**) is usually used to handle a single sequence. When scientists browse or search sequence/annotation data, they usually want to begin from the whole genome level, then to chromosome and sequence level. The genome level or chromosome level data gives the scientist an overview of the organism. It is very hard to store and retrieve genome level data using DAS/1, it would be helpful if

[1] All authors contribute equally to this proposal
[2] Contacts: Dr. Dana Eckart, dana@vbi.vt.edu

DAS/1 can be extended to store genome and chromosome level of data, such as the number of chromosomes, cytogenetic banding patterns, and genetic linkage maps for each chromosome. As for gaps, although the actual DNA sequence is not available, other data associated with the gap region may still be available, such as the approximate size of the gap, genetic maps for the region. For this case, it might be useful to treat gaps as special segments.

Notational conventions:

- *ITALIC* in the text indicates XML attributes.
- **BOLD** in the text indicates XML elements.

1. The combination of DAS GFF, DAS SEQUENCE, DASEP and DASDSN

----- Problem for current DTD

==========

In DAS/1, there are four independent pieces for **GFF**, **SEQUENCE**, **DASEP** (entry point) and **DASDSN**. There is no common root element to combine these individual XML documents. In order to retrieve a sequence and its annotation, two separate XML documents will be returned from the server, one document for sequence, the other document is for annotation information. Since sequence and annotation are in separate documents, there should be some rules for mapping sequence to segment. If you want to move the data (sequence and annotation) from one place to another, we have to transfer two documents instead of one.

---- Proposed Solution

==========

We propose to add a new root element named **DAS** and put **DASSEQUENCE** , **GFF**, **DASEP** and **DASDSN** under the same root element **DAS**.

---- Proposed DTD

==========

```
<!ELEMENT DAS (DASSEQUENCE?, GFF?, DASEP?, DASDSN?, DESCRIPTION?)>
<ELEMENT DESCRIPTION (#PCDATA)>
```

---- Example

==========

```
<DAS>
      <DASSEQUENCE>
            <!- - DNA sequence is added here  -->
      </DASSEQUENCE>
      <GFF>
         <!- - DAS Segment and features information are added here  -->
      </GFF>
      <DASEP>
         <!- Entry point information -->
      </DASEP>
      <DASDSN>
         <!- DSN information -->
      </DASDSN>
      <DESCRIPTION>
          This is mouse chromosome 1 short arm sequence and its
          alignment to human DNA sequence
      </DESCRIPTION>
</DAS>
```

---- Rationale

==========

The advantage of combining sequence and annotation is to simplify the mapping from sequences to annotations (segments). Also, if the DNA sequence, annotation, DSN and entry point information can all be stored in a single XML document, then this will increase data portability and integrity. If we want to export/import the data, one XML document will be enough to store all the required information (such as sequence, annotation, entry point). Also we can only use one XML parser to parse any kind of DAS document. Furthermore, it is still possible to separate these pieces when so desired.

2. Extend DNA SEQUENCE

----Problem for Current DTD

==========

DAS/1 defines **DASDNA** as a required top-level element. This tag includes one and only one DNA **SEQUENCE** element. Due to this limitation, it is hard to have multiple sequences including DNA, RNA and Protein defined in a single DAS document. This makes it difficult to directly relate protein structural and functional information, such as structure alignment, to genomic data. In some cases, we might want to include additional sequence information, such as the source of the organism, or we have no **SEQUENCE** information. To adapt more possible situations to a distributed annotation system, we propose extending **DASSEQUENCE**.

----Proposed Solution

==========

We proposed that **DASSEQUENCE** allow a single DAS document to have zero or more sequences with a sequence type of DNA, RNA or PROTEIN.

----Proposed DTD

==========

```
<!ELEMENT DAS (DASSEQUENCE?, GFF?, DASEP?, DASDSN?)>
<!ELEMENT DASSEQUENCE (SEQUENCE+)>
<!ELEMENT SEQUENCE (#PCDATA)>
<!ATTLIST SEQUENCE id CDATA #REQUIRED>
<!ATTLIST SEQUENCE start CDATA #REQUIRED>
<!ATTLIST SEQUENCE stop CDATA #REQUIRED>
<!ATTLIST SEQUENCE version CDATA #REQUIRED>
<!ATTLIST SEQUENCE type (DNA|RNA|PROTEIN) #REQUIRED>
<!ATTLIST SEQUENCE organism CDATA #IMPLIED>
<!ELEMENT GFF (SEGMENT+, GROUP*)>
<!ELEMENT SEGMENT (FEATURE*, NOTE*)>
<!ATTLIST SEGMENT  id CDATA #REQUIRED>
```

```
<!ATTLIST SEGMENT  refid CDATA #REQUIRED>
<!ATTLIST SEGMENT  start CDATA #REQUIRED>
<!ATTLIST SEGMENT  stop CDATA #REQUIRED>
<!ATTLIST SEGMENT  version CDATA #IMPLIED>
<!ATTLIST SEGMENT  organism CDATA #IMPLIED>
```

----Example

==========

```
<DAS>
  <DASSEQUENCE >
    <SEQUENCE id="BAB16S" start="1" stop="1430" version="2.10"
              type="DNA" organism="brucella">
        TCAACTTGAGAGTTTGATCCTGGCTCAGAACGAACGCTGGCGGCAGGCTTAACACATGCAACTC
        GAGCGCCCGCAAGGGTGAGCGGCAGACGGGTGAGTAACGCGTGGGAACGTACCATTTGCTACGG
        AATAACTCAGGGAAACTTGTGCTAATACCGTATGTGCTTGGGGGAAAGATTTATCGGCAAATGA
        TCGGCCCGCGTTGGATTAGCTAGTTGGTGGGGTAAAGGCTCACCAAGGCGACGATCCATAGCTG
        GTCTGAGAGGATGATCAGCCACACTGGGACTGAGACACGGCCTAGACTCCTACGGGAGGCAGCA
        GTGGGGAATATTGGACAATGGGCGCAAGCCTGATCCAGCCATGCCGCGTGAGTGATGAAGGCCC
        TAGGGTTGTAAAGCTCTTTCACCGGAGAAGATAATGACGGTAACCCGAGAAGAAGCCCCGGCTA
        ACTTCGTGCCAGCAGCCGCGGTAATACGAAGGGGGCNAGCGTTGTTCGGATTTACTGGGCGTAA
        AGCGCACGTAGGCGGACTTTTAAGTCAGGGGTGAAATCCCGGGGCTCAACCCCGGAACTGCCTT
        TGATACTGGAAGTCTTGAGTATGGAAGAGGTGAGTGGAATTCCGAGTGTAGAGGTGAAATTCGT
        AGATATTCGGAGGAACACCAGTGGCGAAGGCGGCTCACTGGACCATTACTGACGCTGAGGTGCG
        AAAGCGTGGGGAGCAAACAGGATTAGATACCCTGGTAGTCCACGCCGTAAACGATGAATGTTAG
        CCGTCGGGGTGTTTACACTTCGGTGGCGCACGTAACGCATTAAACATTCCGCCTGGGGAGTACG
        GTCGCAAGATTAAAACTCAAAGGAATTGACGGGGGCCCGCACAAGCGGTGGAGCATGTGGTTTA
        ATTCGAAGCAACGCGCAGAACCTTACCAGCCCTTGACATCCCGGTCGCGGTTAGTGGAGACACT
        ATGGTTCAGTTAGGCTGGACCGGAGACAGGTGCTGCATGGCTGTCGTCAGCTCGTGTCGTGAGA
        TGTTGGGTTAAGTCCCGCAACGAGCGCAACCCTCGCCCTTAGTTGCCAGCATTCAGTTGGGCAC
        TCTAAGGGGACTGCCGGTGATAAGCCGAGAGGAAGGTGGGGATGACGTCAAGTCCTCATGGCCC
        TTACGGGCTGGGCTACACACGTGCTACAATGGTGGTGACAGTGGGCAGCGAGCACGCGAGTGTG
        AGCTAATCTCCAAAAGCCATCTCAGTTCGGATTGCACTCTGCAACTCGAGTGCATGAAGTTGGA
        ATCGCTAGTAATCGCGGATCAGCATGCCGCGGTGAATACGTTCCCGGGCCTTGTACACACCGCC
        CGTCACACCATGGGAGTTGGTTTTACCCGAAGGCGCTGTGCTAACCGCAAGGAGGCAAACGACC
        ACGGTAGGGTCAGCGACCGGGG
    </SEQUENCE>
```

```
  <SEQUENCE id="Q9RGH9" start="1" stop="267" version="2.10"
            type="PROTEIN" organism="brucella">
      XPKLEEGVEGLVHVSEMDWTNKNIHPSKVVQVGDEVEVQVLDIDEERRRISLGIKQCKSNPWED
      FSSQFNKGDRISGSIKSITDFGIFIGLDGGIDGLVHLSDISWNEVGEEAVRRFKKGDELETVIL
      SVDPERERISLGIKQLEDDPFSNYASLHEKGSIVRGTVKEVDAKGAVISLGDDIEGILKASEIS
      RDRVEDARNVLKEGEEVEAKIISIDRKSRVISLSVKSKDVDDEKDAMKELRKQEVESAGPTTIG
      DLIRAQMENQG
  </SEQUENCE>
</DASSEQUENCE>

<GFF>
  <SEGMENT id="SEG11" refid="Q9RGH9" start="1" stop="267"
            organism="brucella">
     <FEATURE id="PF00575">
        <TYPE>Domain</TYPE>
        <START>1</START>
        <END>228</END>
        <LINK href="http://www.sanger.ac.uk/getacc?PF00575"/>
     </FEATURE>
     <FEATURE id="SM00316">
        <TYPE>Domain</TYPE>
        <START>3</START>
        <END>228</END>
        <LINK href="http://www.smart.org/query?ACC=SM00316"/>
     </FEATURE>
  </SEGMENT>

  <SEGMENT id="SEG12" refid="BAB16S" start="1" stop="1430"
            organism="brucella">
     <FEATURE id="1">
        <TYPE>ORF</TYPE>
        <START>388</START>
        <END>543</END>
        <ORIENTATION>+</ORIENTATION>
     </FEATURE>
     <FEATURE id="2">
        <TYPE>ORF</TYPE>
        <START>643</START>
```

```
            <END>765</END>
            <ORIENTATION>+</ORIENTATION>
         </FEATURE>
         <FEATURE id="3">
            <TYPE>ORF</TYPE>
            <START>1316</START>
            <END>1444</END>
            <ORIENTATION>+</ORIENTATION>
         </FEATURE>
      </SEGMENT>
   </GFF>
</DAS>
```

----Rationale

==========

The proposed **DASSEQUENCE** is more extensible and flexible than the original DAS/1. For example, we can use DAS to relate protein structural and functional information to the genome. Both protein and DNA sequence data for the same organism can be in a single DAS document. The tradeoff of this **DASSEQUENCE** proposal is that **SEGMENT** information is very hard to map to **SEQUENCE**. To solve this problem, we will add the implied attribute *REFID* to the **SEGMENT** element to refer to **SEQUENCE** id. The example shown includes both DNA and Protein sequences from *Brucella* and uses different prediction programs to predict both sequences and presents feature results. We also add *ORGANISM* attribute information in **SEGMENT** element in cases where no sequence information is included in the DAS document.

3. DAS GROUP

---- Problems with existing DAS

==========

In DAS/1, the **GROUP** element was defined inside the **FEATURE** element. This has caused the following two problems:

a) No support for multi-level grouping.

The single-level grouping mechanism only supports one-level of feature hierarchy. While in biology, multi-level feature grouping is often desired.

b) Group data redundancy.

For example, if feature F1 and feature F2 belong to the same group G, the same information about G will be repeated in both F1 and F2.

---- Proposed Solution

==========

The major motivation for extending DAS grouping is to provide support for multi-level features. We approach this goal in two ways: one is to allow nested features (Please see "Feature hierarchy" section for details), the other is to move group out of feature and allow groups of groups.

Feature nesting works nicely to describe location-oriented relationships among features. For example, feature A located at (1,100), feature B located at (10,20), feature C located at (60,80), then we can consider B and C as sub-features of A in most cases. However, besides "physically containing", there are still many other possible intrinsic relationships among features. For example, three promoter features can be grouped together; four genes that separately encode the same type of enzymes, say, protein kinases, could be grouped together; a gene and its regulation sequences could be grouped together to suggest one gene expression regulation pathway. In all these situations, the container of related features is not a super feature by itself, due to the lack of meaningful physical location of the container. In situations like these, feature nesting will fail to properly represent this type of non-location-oriented feature hierarchy. To solve this problem, we propose "GROUP" should handle all other possible feature relationships other than "physically containing", such as "function similarity".

Furthermore, multi-level grouping may be desirable under many circumstances. In the last example, several of these gene regulation pathway groups can be further grouped together to describe a common mechanism of gene regulation. Therefore group of groups (plus some features or not) is also allowed.

Note that the user might want to group features in a variety of ways, and the DTD does not pose any limitations on how to make the group decision. If the user wants to group "physically containing" type of features, it's still legal in terms of the grammar, just be aware that this may lead to data redundancy, if this relationship has already been described in the feature hierarchy.

---- Proposed DTD

==========

```
<!ELEMENT GROUP (PARTICIPANT+, NOTE*)>
<!ATTLIST GROUP id CDATA #REQUIRED >
<!ATTLIST GROUP type CDATA #IMPLIED >
<!ELEMENT PARTICIPANT (NOTE*) >
<!ATTLIST PARTICIPANT refid CDATA #REQUIRED >
<!ATTLIST PARTICIPANT type (group | feature) #REQUIRED >
```

The **GROUP** element is put at the same level as **SEGMENT** to support grouping across segments.
The *ID* (required) attribute makes it possible to group several groups into one super group.
The *TYPE* attribute suggests the category of relationship among participating group members. For example, a group of several promoter features is a group of "function" type; a group of several repeat region features is a group of "sequence" type.

<**NOTE**> (optional, zero or more)
The **NOTE** element can provide human-readable information about the group, such as description of the group as a whole, reasons to include some specific feature in this group, etc.

<**PARTICIPANT**> (required, one or more)
Each group can have one or more participants. In the **PARTICIPANT** element, the *REFID* attributes provides the reference ID for the members of the group. As a participating group member could be a feature, or another group, the required attribute *TYPE* indicates which is the case.

---- Example

==========

```
<GFF>
      <SEGMENT id=“seg1” refid=”1” start=”1” end =”1000” >
            <FEATURE id="seg1.gene1.exon1"/>
                  <TYPE id=”exon”>exon</TYPE>
                  <METHOD id=”id”>
                        <DESCRIPTION>decription</DESCRIPTION>
                        <SCORE>1</SCORE>
                  </METHOD>
                  <START>200</START>
                  <END>400</END>
                  <ORIENTATION>+</ORIENTATION>
                  <PHASE>-</PHASE>
            </FEATURE>
            <FEATURE id="seg1.gene2.exon1"/>
                  <TYPE id=”exon”>exon</TYPE>

                  <METHOD id=”id”>
                        <DESCRIPTION>decription</DESCRIPTION>
                        <SCORE>1</SCORE>
                  </METHOD>
                  <START>400</START>
                  <END>600</END>
                  <ORIENTATION>+</ORIENTATION>
                  <PHASE>-</PHASE>
            </FEATURE>
      </SEGMENT>
      <SEGMENT id=“seg2” refid=”1” start=”3000” end=”5000”
               version=”1.0”>
            <FEATURE id="seg2.feat1"/>
                  <TYPE id=”regulation signal”></TYPE>
                  <METHOD id=”id”>
                        <DESCRIPTION>decription</DESCRIPTION>
                        <SCORE>10</SCORE>
                  </METHOD>
```

```
            <START>3100</START>
            <END>3150</END>
            <ORIENTATION>+</ORIENTATION>
            <PHASE>-</PHASE>
        </FEATURE>
        <FEATURE id="seg2.exon">
            <TYPE id=”exon”>exon</TYPE>
            <METHOD id=”id”>
                <DESCRIPTION>decription</DESCRIPTION>
                <SCORE>1</SCORE>
            </METHOD>
            <START>3200</START>
            <END>3400</END>
            <ORIENTATION>+</ORIENTATION>
            <PHASE>-</PHASE>
        </FEATURE>
    </SEGMENT>

    <GROUP id=“group1” type=”expression regulation”>
        <NOTE type=”curated”>
            <TEXT>A group of exons that may be regulated by ***
            mechanism”</TEXT>
        </NOTE>
        <PARTICIPANT refid=”seg1.gene1.exon1” type=”feature”>
        <PARTICIPANT refid=”seg1.gene2.exon1” type=”feature”>
        <PARTICIPANT refid=”seg2.exon” type=”feature”>

    </GROUP>

    <GROUP id=“supergroup”, type=”expression regulation”>
        <NOTE type=”curated”>
        <TEXT>regulation signal and exons it may affect”</TEXT>
        </NOTE>
        <PARTICIPANT refid=”seg2.feat1” type=”feature”>
        <PARTICIPANT refid=”group1” type=”group”>
    </GROUP>
</GFF>
```

---- Rational

==========

The advantage of putting **GROUP** outside segment is to avoid group data redundancy when multiple segments exist and there are groups across segments. In the above example, group1 has members from both seg1 and seg2, putting **GROUP** inside segment would require the same information about group1 in both seg1 and seg2. The disadvantage, however, is the loss of DAS segment integrity and portability. When we export seg1, we also need to export group1. On the other hand, if we put **GROUP** inside segment, each segment contains its own **GROUP** information, import/export a segment will become clean and simple.

The group participant *REFID* can be either **FEATURE** *ID* or **GROUP** *ID*, allowing both groups and features to participate in a group. In the above example, group1 and a feature of segment2 are grouped further into a super-group.

In the above sample, group1 contains three different exons. In the real world, member features may not always be of the same types. Therefore, grouping can do much more than feature type filtering.

The group hierarchy is supported using *ID* reference. Group will be a flat structure in the DTD, and contains reference to Ids of participating member groups or features. The reason for not adopting nested groups is to avoid data redundancy. For example, in supergroup-A contains group1 and group2, supergroup-B contains group1 and group3, nesting group will lead to repeat group1 information in two places: supergroup-A and supergroup-B.

4. DAS Feature

4.1 Feature Hierarchy

----Problems for Current DTD

==========

In DAS/1, features are defined at a single level. All features are listed under **SEGMENT**. This causes two problems.

The first problem involves searching. For example, if we want to find all ORFs of a specific feature (gene), we have to search through the whole document. If we need to find ORFs for several features, we have to go through the document again and again. When the document is big, this process is very time consuming.

The second problem is how to group the features. For DAS/1, the only way to describe the relationship between features is to use **GROUP**. However, when relationships are complicated, **GROUP** maybe unable to represent all relationships clearly. When we have multiple levels of related features, describing their relationships by **GROUP** is complicated and confusing. Figure 1 shows a set of multi-level relationships among features.

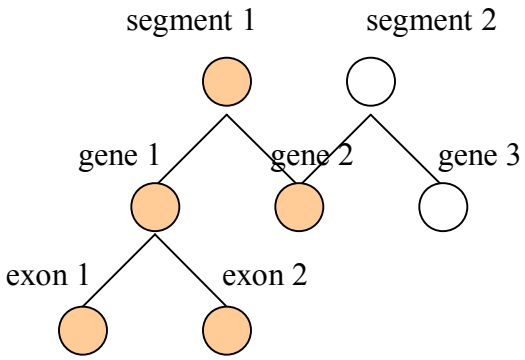


Figure 1. A set of multi-level relationships for features.

For these multi-level relationships, the DAS/1 has to define many groups in order to describe the relationships. For example, we can group segment1, gene1, gene2, exon1, and exon2 together. However, this group does not indicate that exon1 and exon2 are sub-features of gene1, not gene2. We have to make another group including gene1, exon1, and exon2. The more levels we have, the more groups we need to define. And at some point, it becomes very confusing.

----Proposed Solution

==========

We propose a nested feature hierarchy model for DAS/2. Hierarchical structure will be determined by the combination of biological concept and physical location of features. A feature could include sub-features whose start and end positions are located inside the range of its parent feature location, and at the same time, compositional or functional information of the sub-feature are related to the parent feature. Here we can combine the DAS/2 core ontology for feature type hierarchy, which was proposed by Thomas Down in RFC 4. The nested feature hierarchical model will automatically establish the relationship among features. For example, Feature A is located in the range of Feature B, and Feature A is biologically related to Feature B. In DAS/2, Feature A is nested under Feature B.

Due to the complexity of biological concepts, there are exceptions to the consistency of physical location and biological relationship of some features. This happens often for regulation factors. For example, a promoter (Feature C), which is physically located in a gene (Feature A) could have nothing to do with this gene but functionally regulate another gene (Feature B) on a different location. In this case, we use **GROUP** to indicate that Feature C is a sub-feature of Feature B.

----Proposed Feature DTD

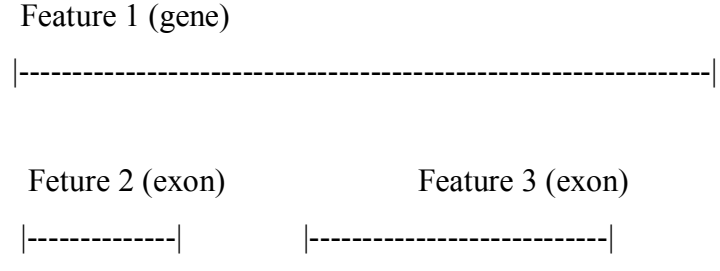
==========

```
<!ELEMENT FEATURE (TYPE, METHOD+, START, END, ORIENTATION?, PHASE?,
                   NOTE*, LINK*, FEATURE*)>
<!ATTLIST FEATURE id CDATA #REQUIRED>
```

```
<!ATTLIST FEATURE label CDATA #IMPLIED>
<!ATTLIST FEATURE version CDATA #IMPLIED>
<!ELEMENT TYPE (#PCDATA)>
<!ATTLIST TYPE id CDATA #REQUIRED>
<!ATTLIST TYPE category CDATA #IMPLIED>
<!ATTLIST TYPE reference CDATA "no">
<!ATTLIST TYPE subparts CDATA "no">
<!ELEMENT METHOD (DESCRIPTION, SCORE*)>
<!ATTLIST METHOD id CDATA #IMPLIED>
<!ELEMENT DESCRIPTION (#PCDATA)>
<!ELEMENT START (#PCDATA)>
<!ELEMENT END (#PCDATA)>
<!--  the following attributes are used to handle fuzzy location for
features -->
<!ATTLIST START isfuzzy (YES|NO) #IMPLIED>
<!ATTLIST START min CDATA #IMPLIED>
<!ATTLIST START max CDATA #IMPLIED>
<!ATTLIST END isfuzzy (YES|NO) #IMPLIED>
<!ATTLIST END min CDATA #IMPLIED>
<!ATTLIST END max CDATA #IMPLIED>
<!ELEMENT SCORE (#PCDATA)>
<!ATTLIST SCORE type CDATA #REQUIRED>
<!ELEMENT ORIENTATION (#PCDATA)>
<!ELEMENT PHASE (#PCDATA)>
<!ELEMENT LINK (#PCDATA)>
<!ATTLIST LINK href CDATA #REQUIRED>
```

----Example

==========



1. Example using current DAS/1.dtd

```
<FEATURE id="cTel33B" label="cTel33B">
    <TYPE id="Gene" category="expression" reference="yes">Gene</TYPE>
    <METHOD id="Genomic_canonical">Genomic_canonical</METHOD>
    <START>1</START>
    <END>2679</END>
    <SCORE>-</SCORE>
    <ORIENTATION>+</ORIENTATION>
    <PHASE>0</PHASE>
    <GROUP id="cTel33B">
        <LINK href="http://www.wormbase.org/db/cTel33B</LINK>
    </GROUP>
 </FEATURE>

<FEATURE id="cTel33B.1" label="cTel33B_exon1">
    <TYPE id="exon" category="expression" reference="yes">exon</TYPE>
    <METHOD id="Genomic_canonical">Genomic_canonical</METHOD>
    <START>120</START>
    <END>679</END>
    <SCORE>-</SCORE>
    <ORIENTATION>+</ORIENTATION>
    <PHASE>0</PHASE>
    <GROUP id="cTel33B">
        <LINK href="http://www.wormbase.org/db/get?cTel33B</LINK>
    </GROUP>
 </FEATURE>

<FEATURE id="cTel33B.2" label="cTel33B_exon2">
    <TYPE id="exon" category="expression" reference="yes">exon</TYPE>
    <METHOD id="Genomic_canonical">Genomic_canonical</METHOD>
    <START>1000</START>
    <END>2279</END>
    <SCORE>-</SCORE>
    <ORIENTATION>+</ORIENTATION>
    <PHASE>0</PHASE>
    <GROUP id="cTel33B">
        <LINK href="http://www.wormbase.org/db/get?cTel33B</LINK>
    </GROUP>
 </FEATURE>
```

2. Example using our proposed dtd

```
<FEATURE id="cTel33B" label="cTel33B">
    <TYPE id="Gene" category="expression" reference="yes">Gene</TYPE>
    <METHOD id="Glimmer">
         <DESCRIPTION>Glimmer2.0</DESCRIPTION>
         <SCORE type="Confidence" >0.33</SCORE>
    </METHOD>
    <START>1</START>
    <END>2679</END>
    <ORIENTATION>+</ORIENTATION>
    <PHASE>0</PHASE>

    <FEATURE id="cTel33B.1" label="cTel33B_exon1">
      <TYPE id="exon" category="expression" reference="yes">exon</TYPE>
      <METHOD id="Glimmer">
          <DESCRIPTION>Glimmer2.0</DESCRIPTION>
          <SCORE type="Confidence" >.98</SCORE>
      </METHOD>
      <START>120</START>
      <END>679</END>
      <ORIENTATION>+</ORIENTATION>
      <PHASE>0</PHASE>
    </FEATURE>

    <FEATURE id="cTel33B.2" label="cTel33B_exon2">
      <TYPE id="exon" category="expression" reference="yes">exon</TYPE>
      <METHOD id="Glimmer">
          <DESCRIPTION>Glimmer2.0</DESCRIPTION>
          <SCORE type="Confidence" >.58</SCORE>
      </METHOD>
      <START>1000</START>
      <END>2279</END>
      <ORIENTATION>+</ORIENTATION>
      <PHASE>0</PHASE>
    </FEATURE>
 </FEATURE>
```

In DAS/1, all features are listed at the same level. In our proposal, feature2 and feature3 are nested under feature1. Because feature2 and feature3 are exons for the gene and they are physically located in feature1. If feature2 is biologically related to more than one feature, feature2 will be repeated inside of other features.

----Rational

==========

The advantage of a nested feature hierarchy is to provide stronger XPath query capability. With the development of XML database technology, DAS documents can be saved and retrieved from the database in XML format. In a nested feature hierarchy model, parent and child features form a tree structure, as shown in Figure 1. This makes it quick and easy to retrieve all sub-features of a specific feature in the genome through XQueries. Once a feature is found, all of its child features could be retrieved immediately.

The nested feature hierarchy also resolves the problem of using too many groups. The hierarchy already defines the grouping clearly. **GROUP** is needed only for special cases. Using the example in Figure 1, exon2 is located in gene1, but its protein product is a structural component of gene2 protein product. For this case, the exon2 feature should not be nested under gene1 or gene2. The relationship between gene2 and exon2 is defined in a **GROUP**.

4.2 Feature METHOD and SCORE Multiplicity

----Problem for Current DTD

==========

DAS/1 defines **METHOD** and **SCORE** as required single sub-elements within **FEATURE**. There is the case that two different programs generate the same feature information for a sequence. Listing the same feature information multiple times will be redundant. It is also hard to compare the prediction results using different methods on the same sequence.

----Proposed Solution

==========

Based on our hierarchical **FEATURE** and **NOTE** proposals, we proposed to combine features when two different methods generate the same feature information for a sequence. **METHOD** is used to indicate the corresponding method, while each **METHOD** has **SCORE** as a sub-element for quality control.

----Proposed DTD

==========

See section 4.1 for proposed Feature DTD

----Example

==========

```
<FEATURE id="cTel33B">
    <TYPE id="Gene" category="expression" reference="yes">Gene</TYPE>
    <METHOD id="glimmer2.10">
         <DESCRIPTION>Glimmer2.10 for gene prediction</DESCRIPTION>
         <SCORE type="Confidence">93.2</SCORE>
    </METHOD>
    <METHOD id="genemark">
         <DESCRIPTION>Genemark for gene prediction</DESCRIPTION>
         <SCORE type="Confidence">99.6</SCORE>
    </METHOD>
    <START>1</START>
    <END>2679</END>
    <ORIENTATION>+</ORIENTATION>
    <PHASE>0</PHASE>

    <FEATURE id="cTel33B.1">
      <TYPE id="exon" category="expression" reference="yes">exon</TYPE>
      <METHOD id="glimmer2.10">
         <DESCRIPTION>Glimmer2.10 for gene prediction</DESCRIPTION>
         <SCORE type="Confidence">93.2</SCORE>
      </METHOD>
```

```
      <METHOD id="genemark">
         <DESCRIPTION>Genemark for gene prediction</DESCRIPTION>
         <SCORE type="Confidence">99.6</SCORE>
      </METHOD>
      <START>120</START>
      <END>679</END>
      <ORIENTATION>+</ORIENTATION>
      <PHASE>0</PHASE>
    </FEATURE>

    <FEATURE id="cTel33B.2">
      <TYPE id="exon" category="expression" reference="yes">exon</TYPE>
      <METHOD id="glimmer2.10">
         <DESCRIPTION>Glimmer2.10 for gene prediction</DESCRIPTION>
         <SCORE type="Confidence">93.2</SCORE>
         <SCORE type="I/Ac">82</SCORE>
         <SCORE type="Do/T">66</SCORE>
      </METHOD>
      <METHOD id="genemark">
         <DESCRIPTION>Genemark for gene prediction</DESCRIPTION>
         <SCORE type="Confidence">99.6</SCORE>
      </METHOD>
      <START>1000</START>
      <END>2279</END>
      <ORIENTATION>+</ORIENTATION>
      <PHASE>0</PHASE>
    </FEATURE>
 </FEATURE>
```

----Rationale

==========

This example clearly demonstrates feature information from two different gene prediction programs. **METHOD** refers to the program applied to predict this feature. **SCORE** is used to evaluate the method. In addition, **METHOD** is allowed to have multiple scores. **SCORE** attribute *TYPE* is used to distinguish scores. The above example provides three types of scores. Confidence-type scores refer to the likelihood of a feature being a

gene/exon. The I/AC-type score refers to initiation signal or acceptor splice site score. The Do/T-type score refers to donor splice site or termination signal score.
The last two scores are closely related to a particular method. At this stage, it is difficult to define a scope of score types due to the variety of methods that could be used. The main advantage of this proposal for **METHOD** and **SCORE** multiplicity is that we can directly and clearly compare the feature information obtained from different programs and at the same time, reduce the feature redundancy.

5. DAS NOTE

----- Problem for current DTD

==========

There is information that does not fit into the sub-element or attributes of features when we retrieve gene annotation data from genbank. For instance, there is some information about "qualifier" and "value". This information was put into the "Note" of feature based on DAS/1, but is difficult to be utilized because **NOTE** is not well structured.
Features can be recognized using many different methods, such as experimental, computation, literature. Information about a gene's functional or protein structure is put into the "Note" of the feature.
Information concerning a feature gradually increases. For example, features were created by computation methods, and later verified by experimental methods. In addition, comments about the computation methods within this context might be included. These data should also be placed into the "Note" of the feature.

---- Proposed Solution

==========

We propose to define "Note" as an element which has itself as a sub-element, a text element and an attribute type. A tree-structured "Note" for a feature can be extended infinitively and unambiguously.

---- Proposed Feature DTD

==========

```
<!ELEMENT NOTE (NOTE*, TEXT)>
<!ELEMENT TEXT (#PCDATA)>
<!ATTLIST NOTE type  #IMPLIED >
<!-- we recommend user to chose following types: (experimental |
     computational | literature | curated | function | structure |
     naming | link | other)
-->
```

---- Example

==========

Note is composed of attributes of *TYPE*, zero or more sub-**NOTE** element(s) and a required **TEXT** element. Several **NOTE** types are recommended in this proposal, but customized types can be used. <**NOTE**>(zero or more per **NOTE**) is used for noting information about its parent **NOTE**.

The following **FEATURE** data is about a discovered gene. At the beginning, there was one gene that was predicted by the GLIMMERM gene prediction software. This gene was further supported by GENESCAN predictions. Meanwhile, a gene was cloned from the EST library and found to encode a DNA binding protein. Based on the gene mapping result, and the gene start and end position, we verified these two genes are actually the same gene.

```
<FEATURE id="cTel33B">
   <TYPE id="Gene" category="expression" reference="yes">Gene</TYPE>
   <METHOD id="glimmerm">
        <DESCRIPTION>GLIMMERM for gene prediction</DESCRIPTION>
        <SCORE type="Confidence">93.2</SCORE>
   </METHOD>
   <START>1</START>
   <END>1000</END>
   <ORIENTATION>+</ORIENTATION>
   <PHASE>PHASE</PHASE>
   <NOTE type="curated">
       <NOTE type="computational">
```

```
            <NOTE type="computational">
                <TEXT>verifying the GENESCAN predicted gene with various
                   parameters. the verified gene has only two CDS and
                   one large intron
                </TEXT>
             </NOTE>
             <TEXT>computed gene using GLIMMERM, it contains three CDS
                   and one intron
             </TEXT>
          </NOTE>
          <NOTE type="experimental">
             <NOTE type="experimental">
                <TEXT>extended experiment demonstrates that it protein
                      is DNA binding protein, and regulate the down
                      stream genes transcription.
                 </TEXT>
             </NOTE>
             <TEXT>by cloning from EST library confirmed the gene. it
                   produce large molecular protein
             </TEXT>
          </NOTE>
    </NOTE>
</FEATURE>
```

-----Rational

==========

NOTE is a sub-element of **FEATURE**, **GROUP**, **SEGMENT** and contains information about its parent. Any additional unanticipated types of information about the parent can be added into the **NOTE** element. For example, a segment was aligned with three contig sequences. These three contig DNAs were cloned into BAC and sequenced. Such information can be put into the segment's **NOTE** element giving user extra information about the segment resource.

NOTE types are useful for describing feature's diversity because many kinds of features, such as regulators, genes, CDS etc, can be obtained by computational and/or

experimental methods. Many **NOTE** elements are structured as a tree. For example, based on a DNA sequence, a set of ORFs is generated by the GLIMMER gene prediction software. Each predicted (ORF) feature has a note whose type value is "computational". For certain ORFs, a user wants to verify them by cloning. For these **FEATURE**'s notes, each note will have a child note whose type value as "experimental". If a specific ORF is cloned and expressed and its product function is clarified, additional information about this feature will be added into the note tree.

Structured tree **NOTE**s allow convenient retrieval of a specific **NOTE** and all corresponding child **NOTE**s by simply querying the desired **NOTE** element instead of traversing all **NOTE** elements. It also simplifies removing **NOTE**s. There is a note-tree as described above that contains several sub-note trees about computational, experimental and curated information about the feature. We can simply remove the computational sub-note if we know that some predicted features (ORF) by GLIMMER were obviously inappropriate due to insufficient training.