Proceedings of the 28th Annual Hawaii International Conference on System Sciences — 1995

YAMATO AND ASUKA: DNA Database Management System

Hajime Kitakami*, Tadasu Shin-I**, Kazuho Ikeo**, Yoshihiro Ugawa***,
Naruya Saitou**, Takashi Gojobori**, and Yoshio Tateno**

* Hiroshima City University
151-5 Ozuka, Numata-Chou
Asa-Minami-Ku
Hiroshima-Shi 731-31, Japan

** DNA Data Bank of Japan
National Institute of Genetics
1111 Yata, Mishima-Shi
Shizuoka-Ken 411, Japan

*** National Institute of
Agrobiological Resources
2-1-2 Kan-Non-Dai
Tsukuba-Shi
Ibaraki-Ken 305, Japan

Abstract

Recently, we newly developed a relational schema for effectively building, integrating, and searching the DNA database on the relational database management system, SYBASE, at DDBJ. The schema is named the "DDBJ-schema". The schema allowed us to implement a DNA database management system with two types of window interfaces, YAMATO and ASUKA, to easily build the DNA database. The first type with the DDBJ-schema is utilized by reviewers (DDBJ support staffs) for processing a single entry sent by an author. The second type with it is utilized by the Human Genome Project for processing multiple entries. Both also have a window interface similar to the flat file format. Finally, we developed two additional systems for both YAMATO and ASUKA. One is a restructuring tool to convert data with the GenBank-schema into those with the DDBJ-schema when we move AWB to the new system. Other is a tree search tool needed to repair the existing taxonomy database which is including many kinds of errors. A structured SQL-programming method is proposed to be used to implement these additional systems. The method is developed through the SQL expression and the control flow language (CFL) of SYBASE, and allowed us to implement the restructuring tool and the tree search tool.

1. Introduction

DNA (Deoxyribonucleic acid) is genetic information for all organisms except RNA viruses. DNA is a continuous series of nucleotides designated by A, T, G, and C. The DNA sequence data and its related information can be stored in the form of a DNA database which consists of nucleotide (DNA and RNA) sequences and related attributes which are bibliographic and biological information. Each nucleotide sequence data is represented as long strings of A's, T's, C's, and G's, along with associated biological annotation.

The DNA Data Bank of Japan (DDBJ) [1] is one of the International Nucleotide Sequence Data banks where the EMBL (European Molecular Biology Laboratories) Data Library [2,3] and NCBI (National Center for Biotechnology Information) /LANL (Los Alamos National Laboratories) [4,5,6,7] are the European and American representatives, respectively. The American DNA data bank is called GenBank. We have been collaborating with these two data banks in many areas through mutual exchanges of data over the international computer network. The mutual exchange of data on a daily basis is realized between DDBJ and the collaborative data banks using a flat-file format (DDBJ/EMBL/GenBank Formats). The DNA database of DDBJ was built in a flat-file system until 3 years ago. The flat-file system has been represented by data description rules or a grammar [8]. The flat-file system is not adequate for building, integrating, and searching a large-scale DNA database whose entries are continuously increasing in such an explosive manner as is occurring today. If we use the flat-file system for building the DNA database, the building is inefficient and it is difficult to manage it in a cooperative work. GenBank helped DDBJ to convert the flat-file data constructed by DDBJ staff, to a relational

format and to install the Annotator's Workbench, AWB. AWB is a tool for building the DNA database with the GenBank-schema [9] based on the relational database management system, SYBASE [10]. However, both integrating and searching are not carried out in the relational database at DDBJ. Moreover, it is difficult to manage the DNA database using AWB, since the GenBank-schema is a complex network structure [1].

This paper will present methodologies for solving those problems using the relational database system, SYBASE, in the UNIX environment. We newly developed a relational schema, DDBJ-schema, to effectively build, integrate, and search the DNA database on the relational database management system at DDBJ. The schema allowed us to implement a DNA database management system with two types of window interfaces, YAMATO and ASUKA, to easily build the DNA database. The first type with the DDBJ-schema is utilized by reviewers (DDBJ support staffs) for processing a single entry sent by an author. The second type with the DDBJ-schema is utilized by the Human Genome Project for processing multiple entries sent by itself. Both types also have a window interface similar to the flat file format. Finally, we developed two additional systems for both YAMATO and ASUKA. One is a restructuring tool [1] to convert data with the GenBank-schema into those with the DDBJ-schema, when we move AWB to the new system. Other is a tree search tool [11] needed to repair the existing taxonomy database which is including many kinds of errors. A structured SQL-programming method is proposed to be used to implement these additional systems. The method is developed through the SQL expression and the control flow language (CFL) of SYBASE, and allowed us to implement the restructuring tool and the tree search tool.

2. System Configuration

Figure 1 shows the configuration of the current system for building the DNA Database on the UNIX based computer at DDBJ. AWB and the GenBank-schema are software tools developed by LANL in GenBank. DDBJ's Reviewers check the form and content of a submitted entry sent by an author.

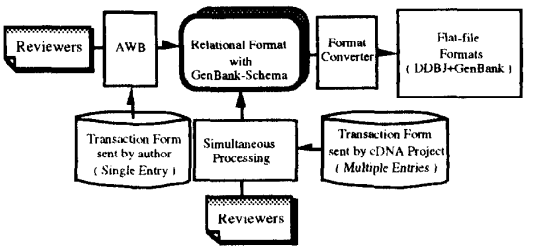


Figure 1. Current System

Reviewers extract the necessary information such as taxonomy, keywords, features etc., from the submitted entry and complete the entry form according to the internal (domestic) and international rules with these tools on the UNIX based computer. We developed a program to simultaneously process multiple entries at DDBJ, because AWB does not have a simultaneous processing function for a large amounts of DNA data. Since early spring 1993, we have been successfully able to process, using this program, many kinds of multiple entries sent by the Human Genome Project [12]. Figure 2 shows the transition for the number of entries to be processed.

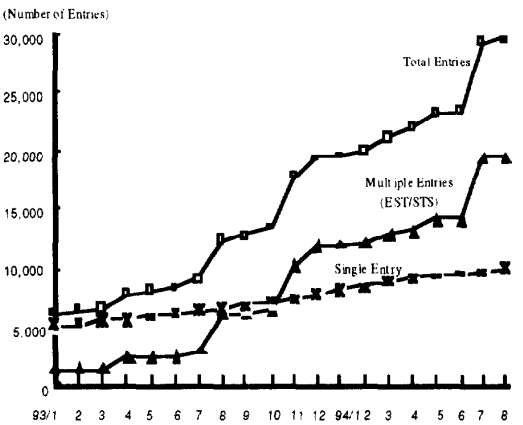


Figure 2. Transition for number of entries

Figure 3 shows the configuration of the newly developed system for solving the previously stated problems on the UNIX based computer at DDBJ.

Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

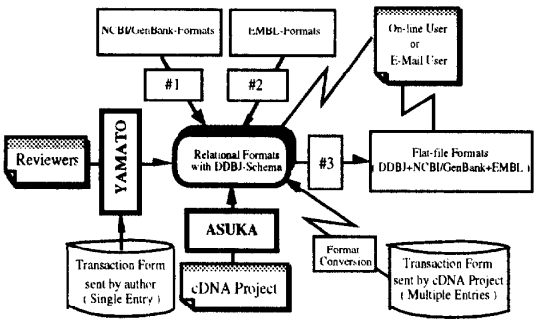


Figure 3. Newly Developed System

We developed the DDBJ-schema for storing the integrated DNA database. The DDBJ-schema allows us to implement two types of window interfaces, called YAMATO and ASUKA.

The first type with the DDBJ-schema is utilized by reviewers (DDBJ support staffs) for processing a single entry sent by an author. It has two kinds of interfaces, YAMATO/UNIX and YAMATO/MAC. The former is implemented in X-window and can be used on UNIX based workstations over computer networks. The latter is implemented in HyperCard and can be used on Macintosh computers over not only computer networks but also telephone networks. YAMATO/MAC is useful for homeworkers who are some of DDBJ-staffs and constructing the DNA database at their houses over telephone networks.

The second type with the DDBJ-schema is utilized by the Human Genome Project for processing multiple entries sent by itself. It has an interface, ASUKA, implemented in the DB-Library of SYBASE and can be used on workstations or character terminals over not only computer networks but also telephone networks. We also developed the restructuring mechanism which is an interface between the GenBank-schema of AWB and the DDBJ-schema. The restructuring will be needed to convert data with GenBank-schema into those with DDBJ-schema, when we move AWB to the new system.

The integrated DNA database needs to receive DNA data with the NCBI/GenBank-flat file of NCBI and EMBL-flat file of the EMBL Data Library. In addition, we need to convert the relational format with the DDBJ-schema into the GenBank flat file format and vice versa. For this reason, we developed conversion programs #1, #2, #3. If we can change AWB into the new system, on-line users and E-mail users will be able to access the integrated DNA database with the relational format over the computer network. The DDBJ provides users with useful software systems to analyze the DNA database with the flat-file format but the software systems do not have any interface with the relational format. The interface would be developed in future at DDBJ.

3. DDBJ-Schema

The DDBJ-schema shown in Appendix-1 is more easily managed than the GenBank-schema. The GenBank-schema [1] has 60 tables with the complex network structure and its structure is different from the grammatical structure of the flat-file format. This makes it difficult for the DDBJ-staff to manage the GenBank-schema. We successfully developed the DDBJ-schema with three parts.

Sixteen tables in the first part are connected horizontally by an accession number, "acc_num", and include all the contents of the flat-file formats. These tables are similar to the grammatical structure of the flat-file format. We can provide these tables for integrating and searching the DNA database.

The DDBJ-schema also includes dictionaries which store biological information such as taxonomy and feature key, a list of persons, and a list of addresses in the second part. We use the dictionaries to automatically fill in the blanks of the 16 tables.

In the third part, the DDBJ-schema includes system control tables which are shown in the lowest part of Appendix-1. The system control tables are used for data restructuring, data transportation, and data flow management.

4. Window Interfaces

We developed two types of window interface to build the 16 tables in the DDBJ-schema.

The first type has two interfaces, YAMATO/UNIX and YAMATO/MAC. The system configuration of YAMATO is shown in Figure 4. YAMATO/UNIX and YAMATO/MAC are utilized by reviewers for

Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

constructing the DNA database over the computer network. YAMATO/MAC is utilized by reviewers for processing the single entry over not only computer networks but also telephone networks.

The second type has an interface, ASUKA, implemented in the DB-Library of SYBASE and can be used on either of both workstation and character terminal over computer networks and telephone networks. It is utilized by the Human Genome Project for processing multiple entries. Both types also have a window interface similar to the flat file format.

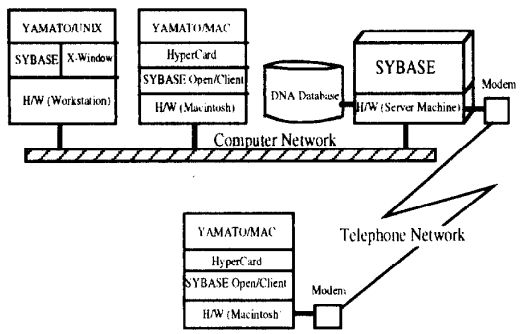


Figure 4. System Configuration of YAMATO/UNIX and YAMATO/MAC

We show an example of YAMATO/UNIX in Appendix-2. The window shows the range of data from "locus_line" to "source_line" in the flat file format. Other lines can be displayed by clicking a mouse button after users move the pointer to either "REFERENCE" or "FEATURE".

For example, if we click a mouse button after moving the pointer to "FEATURE", we can open the FEATURE window shown in Appendix-2. The window shows the range of data from "feature_line" to "sequence_line" in the flat-file format. Thus the window is similar to the grammatical structure of the flat-file format. The DDBJ-schema allowed us to implement easily the window interface.

Two typical windows of YAMATO/MAC are shown in Figure 5 and 6. They are available on Macintosh computers and similar to YAMATO/UNIX shown in Appendix-2. The first window of YAMATO/UNIX is divided into two parts, MAIN and SOURCE windows, in the YAMATO/MAC interface, since HyperCard does not support such long column length as X-Window.

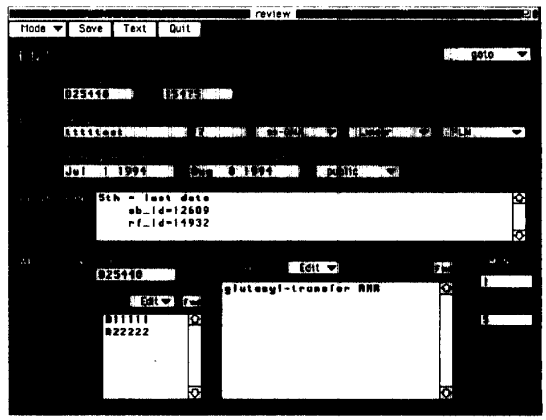


Figure 5. MAIN Window of YAMATO/MAC

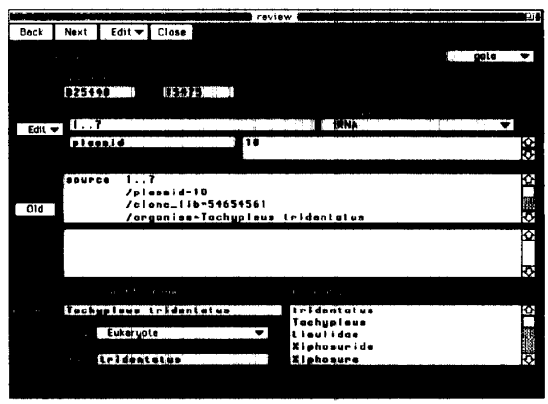


Figure 6. SOURCE Window of YAMATO/MAC

AWB has also an X-window interface and an character terminal interface. The X-Window interface is similar to the grammatical structure but the character terminal interface is not similar to it and not so easy to use for DDBJ's reviewers. YAMATO/MAC has more powerful interface than AWB with the character terminal interface over telephone networks.

ASUKA is shown in Appendix-3. The window shows the range of data from "locus_line" to "segment_line" in the flat file format. Other lines can be displayed by

Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

pressing the return key of the keyboard after users move the pointer to either "Seq!", "Comment!", "Reference!", "Source!", or "Feature!".

For example, if we select this "Seq!", we can open the SEQUENCE window shown in the right side of the appendix. AWB does not have a simultaneous processing interface for multiple entries sent by the Human Genome Project but ASUKA has processing interface for them over computer networks or telephone networks.

5. Additional Systems

We developed two additional systems for both YAMATO and ASUKA. One is a restructuring tool to convert data with the GenBank-schema into those with the DDBJ-schema, when we move AWB to the new system. Other is a tree search tool needed to repair the existing taxonomy database which is including many kinds of errors. These errors result in confuse with the management of DNA database. Both systems are implemented in the structured SQL-programming method.

A program generally is defined as an algorithm and data structure. For example, let us consider a internal sorting written in C. We can regard procedural statements for sorting as algorithm and a linear list stored with input data as data structure.

This structured SQL-programming is achieved by regarding set of tables as data structure and procedure written in both SQL and CFL (Control Flow Language) as algorithm. CFL means such control statements as "if-then-else", "go-to", "while", "begin-end" and so on.

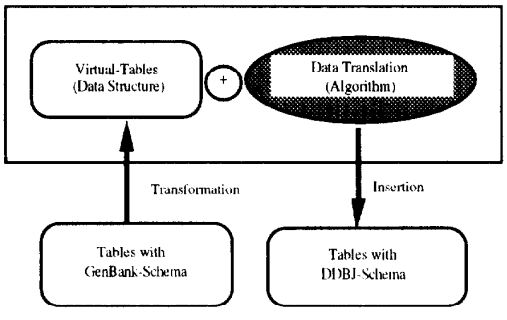


Figure 7. Restructuring mechanism

5.1 Restructuring

Figure 7 shows the restructuring mechanism [1] for interfacing between the GenBank-schema and the DDBJ-schema. It is implemented by the structured SQL-programming method. The method provides us with a program including the virtual tables as data structure. The program is written in both CFL and SQL.

The real tables with the GenBank-schema are transformed into virtual tables through the "view" function [1]. The virtual tables and the real tables with the DDBJ-schema are similar to each other, but they are not exactly the same. This is due to limitations in the capability of the "view" function. We implemented the data translation algorithm using both CFL and SQL of SYBASE. In our trial system, we successfully restructured from the GenBank-schema to the DDBJ-schema using both the data translation algorithm and the virtual tables.

5.2 Tree Search

The tree search functions [11] is useful in managing the taxonomy database. A typical algorithm of them can be defined as recursive join for the taxonomy database. All the taxonomy databases is constructed with the DNA databases of the international DNA data banks. They are also useful in defining as powerful electronic dictionaries which aid in biological research by computer. The taxonomy databases are, however not consistently integrated with a relational format. If we can achieve consistent unification of the taxonomy databases, it will be useful in comparing many research results, and investigating future research directions from existent research results.

In particular, it will be useful in comparing relationships between phylogenetic trees inferred from molecular data and those constructed from morphological data. The system has many kinds of the tree search functions which are lineage, homology, posterity, and so on [11]. All the functions are implemented in the structured SQL-programming method. These functions are shown in Figure 8.

For example, the lineage function searches for a path from a given node to the root node,

Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

which does not have any parent node in the tree structure. The path is a set of nodes found by searching in the "taxonomy" table. The lineage function can be implemented in the control flow language of SYBASE called the "stored procedure". This program is shown in Appendix-4.

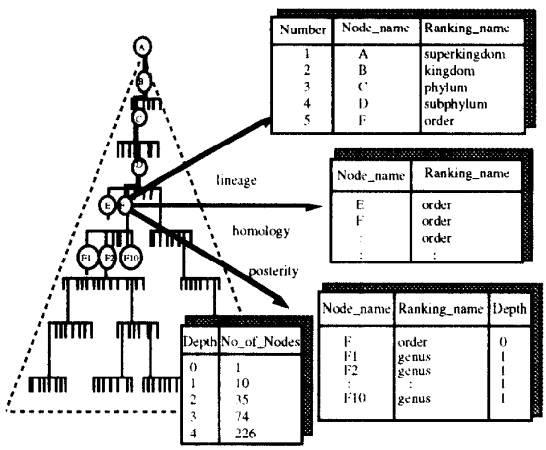


Figure 8. Neighborhood Search

Our approach includes the object-oriented database concept [13], since the procedure is a kind of method for hiding the table structures of the "taxonomy" table. If we apply an artificial intelligence approach, we can obtain another kind of a program implemented in prolog [14]. This approach shown in Appendix-5 is constructed with a smaller program than in the previous approach. If we need more complex and higher processing to unify taxonomy databases in the future, the artificial intelligence approach would be preferable in implementing more efficient processing.

6. Conclusions

We have realized effective join processing with the newly developed DDBJ-schema which is easy to use, because its structure looks like a grammatical structure of the flat-file format. The DDBJ-schema was successfully implemented as a relational schema with three parts. It is possible to provide the SQL-service for on-line and E-mail users. Moreover, we shown that the schema allowed us to implement two types of interfaces which are named YAMATO and ASUKA. We also implemented a tool to convert EMBL and NCBI/GenBank-formats into the relational format with the DDBJ-schema.

Finally, we proposed a structured SQL-programming method. The restructuring and tree search was implemented in the programming method. We have succeeded in a trial restructuring data with the GenBank-schema into those with the DDBJ-schema by use of the structured-SQL programming method. We are planning to move the current system to the newly developed system at DDBJ.

Acknowledgments

We wish to thank all the DDBJ-staff and Hitachi Software Engineering Co., Ltd. involved in developing the DNA database system for their help. We also thank Dr. Yukiko Yamazaki involved in designing the window interface for her help.

References

[1] Hajime Kitakami, Yukiko Yamazaki, Kazuho Ikeo, Yoshihiro Ugawa, Tadasu Shin-I, Naruya Saitou, Takashi Gojobori, and Yoshio Tateno: Building and Search System for a Large-scale DNA Database, Proc. of colloquium "Molecular Bioinformatics", IEE press, pp.61-69, London, 1994.

[2] Patricia Kahn and Graham Cameron: EMBL Data Library, Methods in Enzymology, Vol. 183, pp.23-31, 1990.

[3] Desmond G. Higgins, Rainer Fuchs, Peter J. Stoeher and Graham N. Cameron: The EMBL Data Library, Nucleic Acids Research, Vol.20, Oxford University Press, pp.2071-2074, 1992.

[4] Christian Burks, Michael J. Cinkosky, Paul Gilna, et al: GenBank: Current Status and Future Directions, Methods in Enzymology, Vol. 183, pp.3-22, 1990.

[5] Michael J. Cinkosky, James W. Fickett, Paul Gilna, and Christian Burks: Electronic Data Publishing and GenBank, Science, Vol. 252, pp.1273-1277, 1991.

Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

[6] Christian Burks, Michael J. Cinkosky, William M. Fischer, Paul Gilna, Jamie E.-D.Hayden, Gifford M. Keen, Michael Kelly, David Kristofferson and Julie Lawrence: GenBank, Nucleic Acids Research, Vol.20, Oxford University Press, pp.2065-2069, 1992.

[7] Research News, Managing the Genome Data Deluge, Science, Vol.262, No.22, pp.502-505, 1993.

[8] GenBank Release Notes, Release 69.0, Tape Distribution, IntelliGenetics Inc., September 1991.

[9] GB-Schema, News From GenBank, Summer/Autumn 1991, Vol.4, No.4, c/o IntelliGenetics, Inc., 700 East El Camino Real, Mountain View, CA 94040.

[10] SYBASE Transact-SQL User's Guide, Sybase Inc., May 1989.

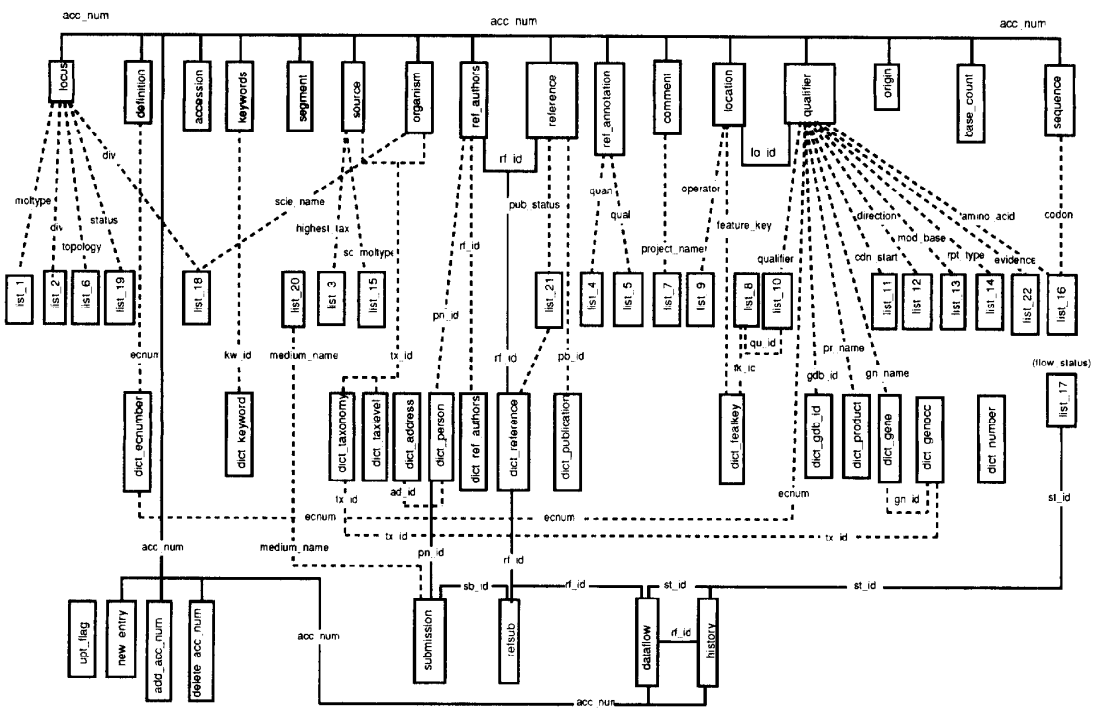
[11] Hajime Kitakami, Yoshio Tateno and Takashi Gojobori: Toward Unification of Taxonomy Databases in a Distributed Computer Environment, Proc. of the second International Conference on Intelligent System and Molecular Biology, AAAI Press, pp.227-235, 1994.

[12] Kousaku Okubo, Naohiro Hori, Ryo Matoba, Toshiyuki Niiyama, Atsushi Fukushima, Yuko Kojima and Kenichi Matsubara: Large Scale cDNA Sequencing for Analysis of Quantitative and Qualitative Aspects of Gene Expression, Nature Genetics, Vol.2, Nov., 1992.

[13] Prabhat K. Andleigh and Michael R. Gretzinger: Distributed Object-Oriented Data-Systems Design,Prentice Hall,Inc.,1992.

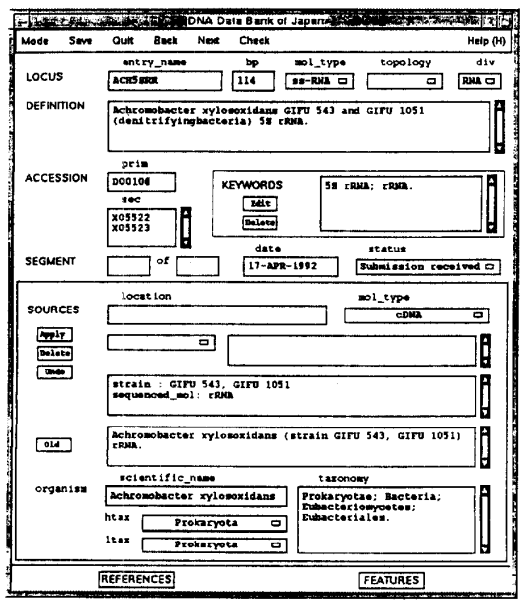
[14] Prolog by BIM(BIM_Prolog) Reference Manual, BIM (Belgium), 1990.

Appendix-1 DDBJ-Schema

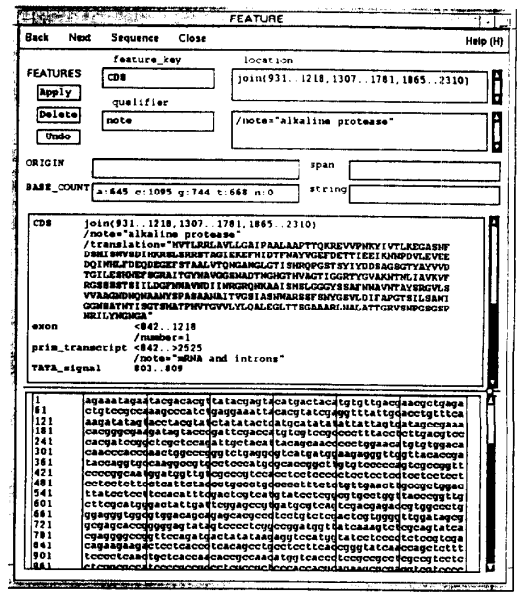


Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

Appendix-2. Window Interface of YAMATO/UNIX

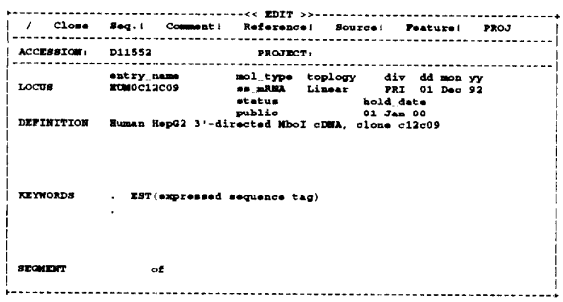


MAIN Window

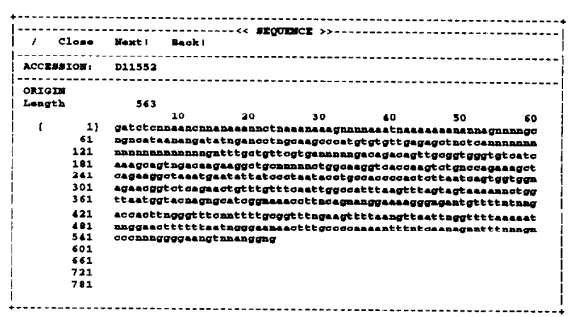


FEATURE Window

Appendix-3. Window Interface of ASUKA



MAIN Window



SEQUENCE Window

Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE

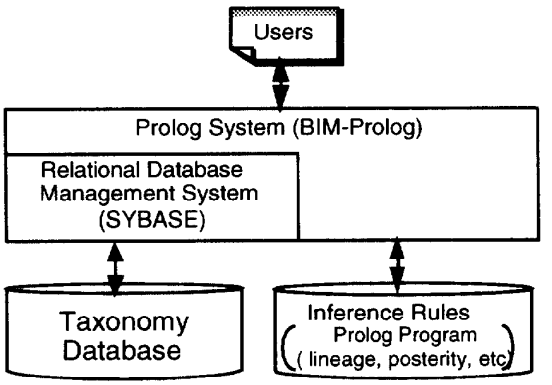
Appendix-4. An Example of Programming for a Lineage Search Implemented in both SQL and CFL

```
create procedure lineage @nodename char(80) as

declare @tuples int
declare @cnt    int

select  @cnt=1
create table tempdb..lineage(
        tx_id char(16),tx_idp char(16),
          level tinyint ,level_name char(32),node_name char(80))
create table tempdb..temp_table(
          tx_id char(16),tx_idp char(16),
          level tinyint ,level_name char(32),node_name char(80))
create table tempdb..work_table(
        tx_id char(16),tx_idp char(16),
          level tinyint ,level_name char(32),node_name char(80))

insert tempdb..temp_table
    select tx_id,tx_tx_idp,@cnt,tl_levname,tx_nodename
    from   taxonomy,taxlevel
    where  tx_tl_id=tl_id
    and    tx_nodename=@nodename
insert tempdb..work_table
    select * from tempdb..temp_table
insert tempdb..lineage
    select * from tempdb..temp_table
select @tuples=count(*) from tempdb..temp_table
delete tempdb..temp_table

while @tuples!=0                                    /* Begin Recursive Join */
begin
      select @cnt=@cnt+1
      insert tempdb..temp_table
          select x.acc_num,y.tx_id,y.tx_tx_idp,   /*     Single Join      */
                 @cnt,tl_levname,tx_nodename
          from   tempdb..work_table x,
                 taxonomy y,taxlevel
          where  tx_tl_id=tl_id
          and    y.tx_id=x.tx_idp

      delete tempdb..work_table

      insert tempdb..work_table
               select * from tempdb..temp_table
      insert tempdb..lineage
               select * from   tempdb..temp_table

      select @tuples=count(*)
      from   tempdb..temp_table

      delete tempdb..temp_table
end                                                 /*  End Recursive Join  */

select * from tempdb..lineage
drop table tempdb..lineage, tempdb..temp_table, tempdb..work_table
```

Appendix-5. An Example of Programming for a Lineage Search Implemented in BIM-Prolog

```
lineage(NODENAME,[[TX_ID,TX_TX_IDP,LEVELNAME,NODENAME]|Result]):-
      taxonomy(TX_ID,TX_TL_ID,TX_TX_IDP,NODENAME),
      taxlevel(TX_TL_ID,LEVELNAME),
      search(TX_TX_IDP,Result).

search(TX_ID,[[TX_ID,TX_TX_IDP,LEVELNAME,NODENAME]|Result]):-
      taxonomy(TX_ID,TX_TL_ID,TX_TX_IDP,NODENAME),
      taxlevel(TX_TL_ID,LEVELNAME),
      search(TX_TX_IDP,Result).
search(TX_ID,[]).

taxonomy(TX_ID,TX_TL_ID,TX_TX_IDP,NODENAME):-

retrieve(db_taxonomy(TX_ID,TX_TL_ID,TX_TX_IDP,NODENAME,_,_,_,_,_,_,_,_,
_,_,_,_)),!.

taxlevel(TX_TL_ID,LEVELNAME):-
      retrieve(db_taxlevel(TX_TL_ID,LEVELNAME,_,_)),!.
```

The "db_taxonomy" and "db_taxlevel" tables are two tables of the taxonomy database defined in section 2. A system configuration for the deductive approach is shown in the following Figure:



Proceedings of the 28th Hawaii International Conference on System Sciences (HICSS '95)
1060-3425/95 $10.00 © 1995 IEEE