A Document Preparation Paradigm Based on the Specification Method of Layout Structure

Toyohide WATANABE, Akihiro UDA and Noboru SUGIE

Department of Information Engineering,
Faculty of Engineering, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-01, JAPAN
(Email) watanabe@yoshida.nuie.nagoya-u.ac.jp

***** Abstract *****

The document preparation facility is one of the most important functionalities in various types of information systems. Now that high density laser-printers are commonly provided and various kinds of jobs are effectively computerized, electronic-form/paper-form documents related to the issue of the desk top publishing are easily composed by computers. Document preparation tools/systems(or formatters), that are currently utilized, embed the layout control data into the source text data together. These traditional formatters are not always successful because the task of embedding layout control data into the source text data is very troublesome and difficult. Namely, such a framework is in short of the flexibility, adaptability and applicability. Our approach enables a layout-independent document preparation mechanism because the original text data are separated from the layout control data. Our layout structure information is specified externally by the form definition language. Through this paper, we discuss the concept and framework of our document preparation facility, and also show a prototype system.

1. INTRODUCTION

Documents are usually attached with application-specific layout structures. The layout structure assists to make the meaningful information explicit in understanding the contents of documents. In case of formatting documents in computers, it is very effective to make up documents by individual original text data and the underlying layout structures. Current formatting tools such as troff, nroff, TEX, etc. introduce layout control characters, whose functions indicate attributes of words, lines, paragraphs and so on individually[1, 2]. The layout control characters are embedded into the original text data together. However, many of them are not always successful in the following points:

- It is troublesome to assign the layout control data to the original text data exactly;
- It is not easy to understand the document structure graphically from the source text data, which embed various layout control data together into the original text data;
- It is not instantaneous to alter the source text data from one document structure to another.

The approaches are in short of flexibility, applicability and adaptability for the data reusability and transparency.

Although the layout control data are basic information to compose documents constructively, they are supplementary data as the representation media. Namely, the layout control data and original text data are different objects in the document composition process. It is desirable that the document preparation facility should manipulate the layout control data and original text data separatedly[3]. In this paper, we address an experimental document preparation mechanism to deal with layout control data and original text data independently, and also show the prototype system. Our document preparation mechanism is looked upon as a kind of layout mapping function that transforms logically collected text data, which the layout control data are excluded, into physically layouted documents. This layout mapping function works with the interpretation of the layout structure, which is specified externally by the form definition language. This framework is different from those of the conventional document preparation tools/systems because the layout control data in our approach are separated exclusively from the original text data.

2. LAYOUT STRUCTURE AND DOCUMENT PREPARATION

The layout structure makes the meaningful content of document and mutual relationship among document items clear, on the basis of the geometric and spatial structure. Generally, the layout structures are too strongly dependent on application-specific usages. Even if the original text data were the same(or similar), documents formatted with different layout structures may be used appropriately in accordance with the applications. However, current document preparation tools/systems are not always successful to make up various kinds of documents from one source text data because the layout control data are embedded into the original text data together. The existing formatters such as troff, nroff and TEX(or LATEX) interpret the layout control data embedded in the source text data selectively. In such a framework of traditional formatters, it is not possible to reuse the source text data, which is already defined by one layout control data, in order to compose another formatted document directly.

Here, we define the original text data as a collection of meaningful item data, in which the layout control data are excluded at all. While, the source text data is defined as a compound data mixed with the original text data and layout control data. The issue about the data reusability and transparency is today one of the most important subjects to share the same text data effectively among different document structures. However, it is difficult to solve this requirement in case that the source text data are too much dependent on the application-specific structures. The data reusability and transparency can be successfully obtained if and only if individually managed data are not composed by any application-specific structures. Namely, it is the most necessary constraint that the layout control data and original text data should be managed disjointly or independently. We show such a conceptual framework in Fig.1. In this framework, the description of layout structures is interpreted by the document preparation procedure as the mapping process. Also, the specification task of layout structures is independent of the composition/editing process of original text data.

We review the layout control mechanism in the traditional document preparation tools/systems. We can observe the most usual formatting method in TEX or LATEX[4].

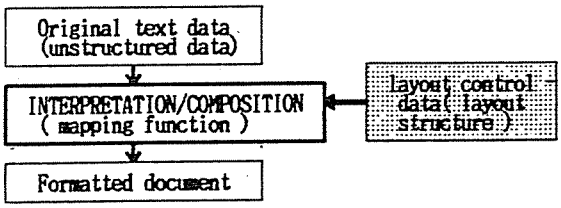


Fig.1 A framework of document preparation mechanism

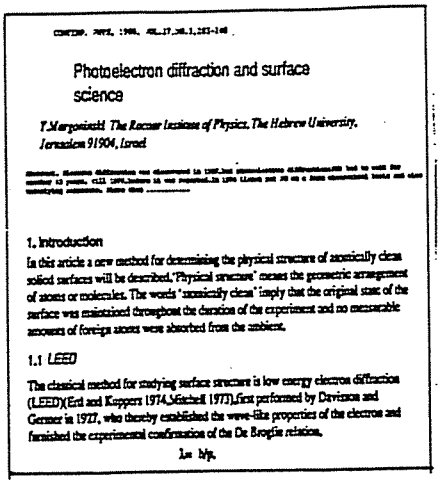
```
\documentstyle[epsf,12pt]{article}
\textwidth 10cm
\begin{document}
\section{Introduction}
Recently, the document preparation facility is one of
important facilities in information systems.
Although various types of document preparation facilities
such as \TeX, troff,nroff, etc. have been developed,
many of them are not always designed under the open
interface architecture.
Namely, they are constructed as slaves for peculiar
editing facilities.
     .......
\end{document}
```

Fig.2 Source text data in LATEX

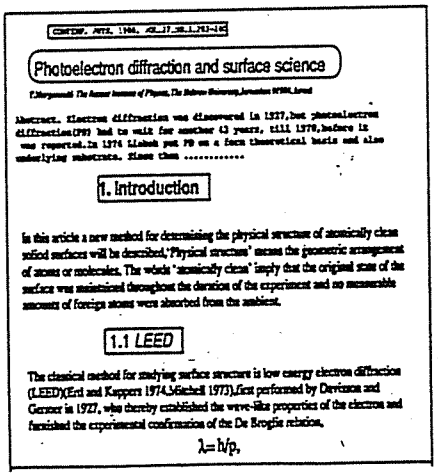
```
<!DOCUTYPE paper PUBLIC ''-//T+F//DTD
          Contemporary Physics//EN''>
<paper>
<title>Photoelectron diffraction and surface science
<author>Y.Margoninski
<position>The Rach Institute of Physics, The Hebrew
University, Jerusalem 91904 ,Isreal
<abstruct>Eelectron diffraction was discovered ...
...a list of acronyms is included.
<h1>Introduction
<p>In this article ...
... absobed from the ambient.
<h2 id='LEED'>
<p>The classical method for studying surface structure
is low energy electron diffraction (LEED)(<citref
refid=''Ertl-K''><citref refid=''Mitchell''>),first
performed ...
```

Fig.3 Source text data in SGML

This tool is originated in troff and nroff with respect to the specification paradigm of layout structures. We show a source text data, embedded with the layout control data in LATEX, in Fig.2. The layout control words, indicated by backend-slashes, are embedded into the source text data in order to distinguish individual items syntactically. In this case, only one document with the specified layout structure is determinately defined through the embedded layout control words. When we want to compose a document with another layout structure, we must modify the source text data directly by looking upon both layout control words and original text data as the same editing object. Next, we attach to SGML[5]. We show a source text data of SGML in Fig.3. In this case, the layout control words, surrounded with the symbols "<" and ">", take roles of "tag"'s for the following character strings. Although the specification method is similar to that in LATEX, the interpretation mechanism is different: the functionality of individual layout control words can be alternately redefined



(a) one formatted document



(b) another formatted document

Fig.4 Two documents composed in SGML

by assigning particular meanings to the tags. For example, two document structures shown in Fig.4 are generated selectively from the same source text data in Fig.3, by interpreting the alternative meanings of individual tags.

The approaches in LATEX(or TEX) and SGML are different in the concept of layout control, but are similar in the specification method. The layout control words in LATEX are predefined in the interpretation mechanism, while those in SGML are changeable in accordance with the currently defined meanings of tags. However, these approaches are not always successful for the data reusability and transparency because the source text data are composed compoundly of original text data and layout control data. The layout control data are inherently extra-data and only the original text data are useful information.

3. OUR FRAMEWORK

The layout control data are inherently independent of the original text data. However, TEX(or LATEX) and SGML adopt the method that embeds the layout control data into the source text data together. These approaches can not support effective data sharing functions in exchanging or reusing only the source text data among various processing facilities because they include formatting-specific control data. In order to make the data reusability and transparency successful, it is necessary to manipulate the layout control data and original text data independently. The layout structures, attached to peculiar applications,

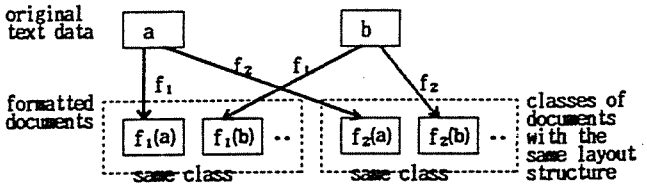


Fig.5 Layout mapping mechanism

are not interpretative by another processing facility even though they were adjustable to the applications.

Our method to separate the layout control data from the original text data is based on the framework of logical and physical structures. The logical structure is defined as a meaningful relationship among individual document items, while the physical structure is a layout structure. This distinct relationship between logical and physical structures is similar to two-layers schema in the database. In our document preparation facility, the original text data defined by the logical structure are arranged into the physically formatted document through the layout information. The interpretation mechanism of the layout information is a mapping of original text data from the logical structure to the physical structure. Our mapping mechanism makes up documents with various kinds of layout structures easily from an original text data through appropriate layout information.

【mechanism】

$f_i, f_j \in$ Set of layout mapping functions

$a \in$ Set of text data

$f_i(a), f_j(a) \in$ Set of formatted documents

$f_i(a) \neq f_j(a)$ iff $f_i \neq f_j$

We show our mapping mechanism in Fig.5. Mapping functions are available when they are applicable to the original text data. In this case, we will not consider the inverse function f_i^{-1} and the function composition $f_i \cdot f_j$ (or $f_j \cdot f_i$). We need some assumptions to implement our framework with such mapping functions. Namely, the original text data must be always constructed by the predefined specification method of logical structure[3, 6].

This logical structure is defined by the data definition language, like the data definition language(DDL) in the database, while the physical structure is specified by the form definition language. The original text data are translated into the corresponding document structure under the interpretation between the data definition language and form definition language. We show a logical structure in Fig.6, using our data definition language. This logical structure is usable to manage a paper-type document. The original text data, associated with these logical structures, can be composed by the data-editing/data-entry facility[6].

4. FORM DEFINITION LANGUAGE

The layout structures can be generally defined by means of a synthesis method. For example, many approaches about the recognition issue of document structures have proposed the methods that interpret documents with knowledge about layout structures intelligently[7-9]. Of course, the layout structure in our objective is available to make up documents though that in the recognition issue is effective to analyze documents. However, the objectives in both issues are to look upon documents as collections of data items, whose attributes such as positions, sizes, lengths, etc. are specified as a part of layout information.

In case of assigning individual item data as a document structure, our mapping function is defined as four main operations: "Element", "Region", "Relation" and "Control". They are illustrated in Fig.7. "Element" distinguishes

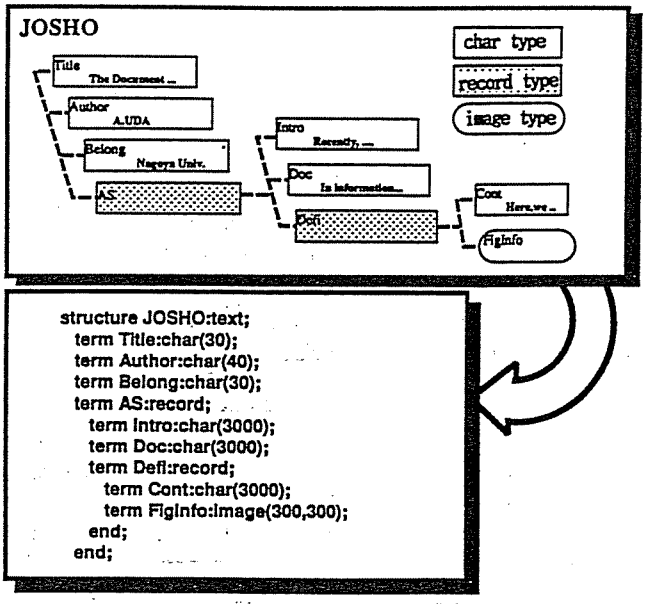


Fig.6 Data definition language

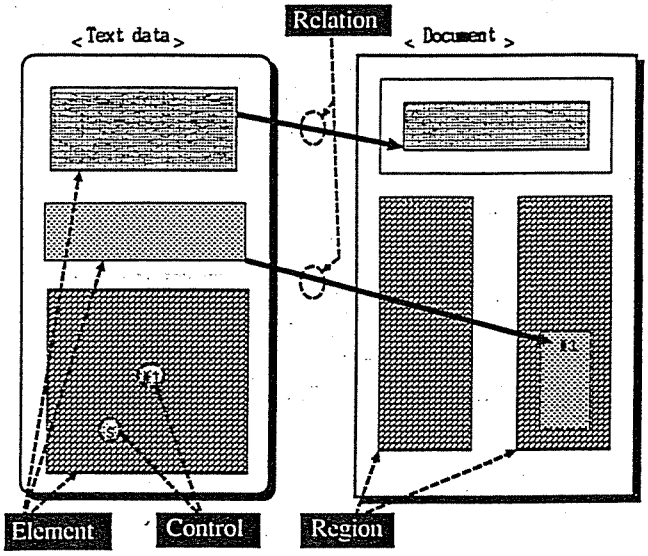


Fig.7 Roles of layout mapping function

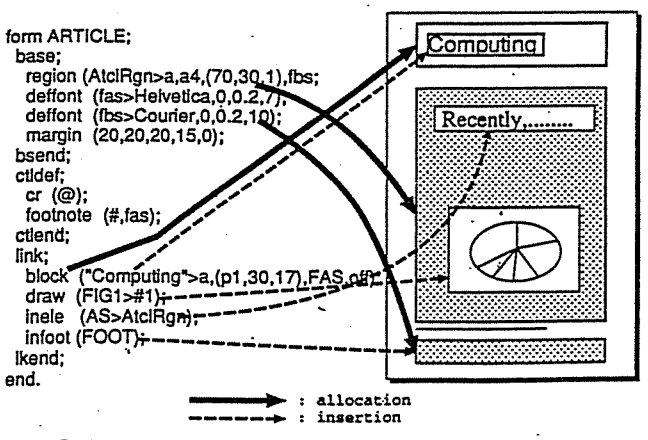


Fig.8 Layout description and document structure

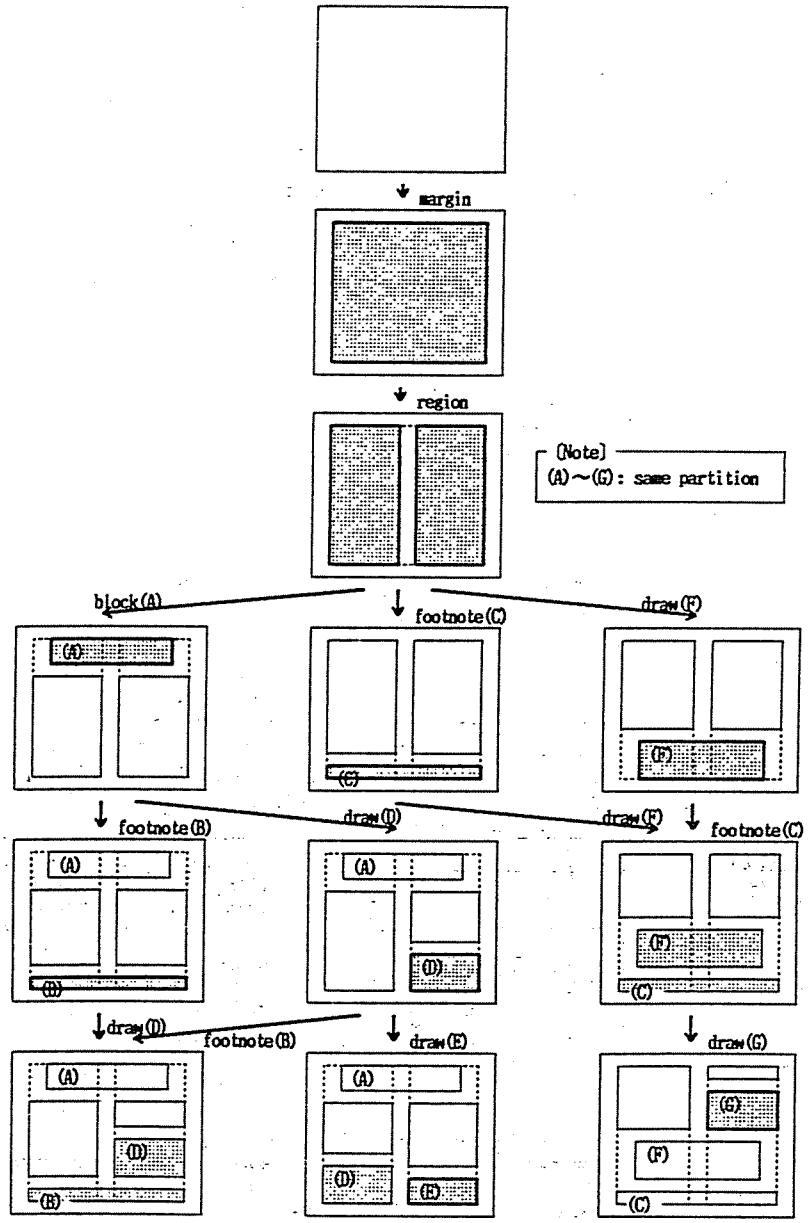


Fig.9 Effects of descriptors

```
form ronbun;
    base;
        region (AtclRgn > a,a4,(70,30,1),fbs);
        deffont (fas > Helvetica,0,0.2,12);
        deffont (fbs > Courier,0,0.2,10);
        margin (20,20,20,15,0);
    bsend;
    ctldef;
        cr (@);
        deffoot (,fas);
    ctlend;
        block (title > a,(p1,30,20),fas,off);
        block (auth > a,(p1,40,30),fas,off);
        block (belon > a,(p1,50,40),fas,off);
        draw (fig1 > ?1);
        in_ele (demo > AtclRgn);
        in_foot (kyaku > #);
end
```

(a) layout description-1

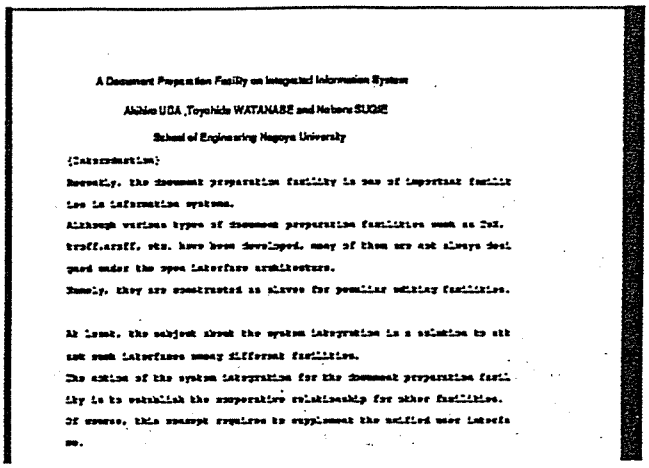
```
form ronbun;
    base;
        region (AR > a,a4,(70,30,1),fbs);
        deffont (font1 > Times-Roman,0,0.1,16);
        deffont (font2 > Courier,0,0.2,12);
        margin (20,20,20,10,0);
    bsend;
    ctldef;
        cr (@);
        deffoot (,font1);
    ctlend;
        block (title > a,(p1,30,20),font1,off);
        block (auth > a,(p1,40,30),font1,off);
        block (belon > a,(p1,50,40),font2,off);
        draw (fig1 > _1);
        in_ele (demo > AR);
        in_foot (kyaku > #);
end
```

(b) layout description-2

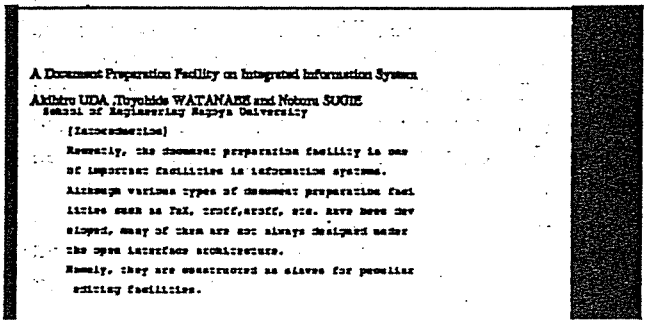
Fig.10 Layout description

individual item data in the original text data: our original text data must be composed of the composite item data constructively, according to the data definition information [6]. In our framework, all original text data are controlled logically through the data definition schema. "Region" defines the effective page areas on the physical document sheets. The page area is a platform to print out individual item data. "Relation" indicates the correspondence between item data of "Element" and the page areas of "Region". Finally, "Control" points out control characters/symbols to manage sentences in detail: to arrange footnotes automatically; to indicate the start of paraphrases; to represent character sets; and so on. The control characters/symbols are necessary to indicate the detail structures of document components though they must be embedded into the original text data. Of course, in our method it is unnecessary to define every control data newly. The existing characters/symbols which are parts of sentences can be redefined effectively as the control data. For example, we can use the existing symbol as an indicator for footnotes when a peculiar symbol which points out to the existence of footnotes is used in the original text data.

We show a layout description in Fig.8, with the corresponding document structure. This description has three major parts. The indicator "base"(and "bsend") defines properties about the region, common to all page areas. The indicator "ctldef"(and "ctlend") defines control characters/symbols. The indicator "link"(and "lkend") defines document components. In "base", the descriptors "region" and "margin" reserve printable page areas: "margin" indicates surroundings of a page area; and "region" allocates effective printting partitions. The descriptor "deffont" defines the property of printting characters and assigns a name to it. For example, in "deffont(fas > Helvetica; 0, 0.2, 7);", "fas" is the name of a font set. In "ctldef", the descriptor "cr" defines a symbol of carriage return for the end of one paragraph. The descriptor "footnote" defines a symbol to represent the footnote reference. For example, in "footnote(#, fas);",



(a) document-1



(b) document-2

Fig.11 Formatted documents

"#" is a footnote indicator and "fas" is the character set for footnotes. In "link", the descriptor "block" reserves a rectangular partition of denoted size on the page area in advance and assigns strings into the reserved partition. While, the descriptor "draw" also indicates to reserve a rectangular partition on the page area, but the reserving partition is relocatable. Namely, the partition in "draw" is reserved when the indicated symbol is encountered in sentences. For example, in "draw(FIG1 > #1);", when "#1" is scanned, the partition for "FIG1" is reserved in an appropriate page area, and then the item data labeled as "FIG1" is assigned to the partition. The descriptor "inele" indicates a data item to be allocated into the page area, defined by "region". The descriptor "infoot" indicates a data item to be allocated as the footnote. The relationships among "margin", "region", "footnote" and "draw" over a page area are shown in Fig.9.

5. EXAMPLES

Here, we examine our document preparation facility on the basis of the mapping mechanism between logical and physical structures. One layout description is shown in Fig.10(a) and another description is shown in Fig.10(b). These two descriptions are different only in some parameter values, though they are similar in our description forms. When these two different descriptions are applied to the same original text data, they generate two different formatted documents as shown in Fig.11, corresponding to the descriptions in Fig.10.

6. CONCLUSION

We addressed an experimental document preparation facility, in which any layout control data about the layout structure are not embedded into the source text data at all. Traditionally, popular formatters such as TEX(or LATEX) and so on manipulate the source text data directly, mixed with the layout control data and original text data at the same character string level. Although SGML is the same as TEX in the specification method of the layout control data, SGML is more flexible than TEX because the meanings for layout control data can be redefined easily. However, SGML as well as TEX embeds the layout control data into the source text data together. In comparison with our approach, the approaches in TEX and SGML are in short of the adaptability, flexibility and applicability.

Of course, we have some future work. Our goal is to attain the data reusability and transparency among various processing facilities in information systems. The distinction between the layout control data and original text data is a part of our goal only in the document preparation paradigm. Information systems must be also designed so as to satisfy these requirements[10].

Acknowledgements --- We are grateful to Prof.T.FUKUMURA of Chukyo University, and Prof.Y.INAGAKI, Prof.J.TORIWAKI of Nagoya University for their perspective remarks, and also wish to thank Ms.K.SUGINO and our research members for their eager cooperations and discussions.

References

1) B.K.Reid:"Scribe: A Document Specification language and Its Compiler", P.148, CMU-CS-81-100, Carnegie-Mellon Univ., Pittsburgh(1980).
2) T. Teitelbaum: "The Cornell Program Synthesizer: A Tutorial Introduction", P.51, Cornell Univ., New York(1980).
3) T. Watanabe: "Architecture of Integrated Office Information System: A Cooperative Integration Method for Various Data Processing Facilities", Proc. of 6th PCCC, pp.320-327(1987).
4) L.Lamport:"A Document Preparation System LATEX: User's Guide and Reference Manual", Addison-Weslay Publishing Company(1986).
5) M. Bryan: "SGML: An Author's Guide to the Standard Generalized Make-up Language", Quorum Technical Services Ltd.(1991).
6) T.Watanabe, Y.Yoshida & T.Fukumura: "Editing Model Based on the Object-oriented Approach", Proc. of 12th COMPSAC, pp.67-74(1988).
7) T.Watanabe, Q.Luo & T.Fukumura:"A Framework of Layout Recognition for Document Understanding", Proc. of DAIR, pp.77-95(1992).
8) A.Dengel & G.Barth:"High Level Document Analysis Guided by Geometric Aspects", Int'l J.of Pattern Recognition and Artificial Intelligence, Vol.2, No.4, pp.641-655(1988).
9) J. Higashino, H. Fujisawa, Y. Nakano & M. Ejiri: "A Knowledge-based Segmentation Method for Document Understanding", Proc.of 8th ICPR, pp.745-748(1986).
10) D.Tsichritzis(ed.): "Office Automation", on Topics in Information Systems, P.441, Springer-Verlag, Berlin Heidelberg(1985).