*The Knowledge Engineering Review*, Vol. 9:3, 1994, 269–286

# Intelligent information presentation systems

BRYAN S. MURRAY

*LUTCHI Research Centre, Department of Computer Studies, Loughborough University of Technology, Loughborough, Leicester, UK (Email: B.S. Murray@lut.ac.uk)*

**Abstract**

This paper discusses the development of Information Presentation Systems (IPS). It identifies that these systems have had two development paths; generic and application-specific. It discusses the main pieces of early work, and identifies the significant problems and shortcomings associated with them. The development of later systems is traced through to the present day, and outstanding research issues are identified.

## 1 Introduction

The generation of a graphic display from scratch involves a considerable amount of work in specifying the details: each line, curve, shape, fill pattern, etc. This has meant that it is very expensive, and from quite early on developers have tried, in some way, to automate the process. Information Presentation Systems (IPS) are systems that automatically generate graphic/multimedia presentations of data using a range of selection and display criteria.

Information Presentation Systems are typically knowledge-based systems that use knowledge about graphic design, presentation styles and the target domain (where available) to automatically generate graphical presentations of data. The systems may employ multiple media and automatically select the data to present. IPS have been developed in a variety of domains, from naval ship movements to technical repair explanations, to generic presentation systems.

The type of presentations can vary considerably, from standard "business" graphics such as bar charts, to domain-specific 3D images. Thus, for example, Mackinlay's generic system APT (Mackinlay, 1986) would select a scatter graph rather than a bar chart to represent nominal domain sets of information (fig. 1a,b).

Feiner's (1985) domain-specific system APEX produces a 3D image of the component that is to be worked on and pictorial guidance as to how to access the component (fig. 2).

A range of research has contributed to the development of such systems. Graphic standards, for example, made a considerable contribution in that development could directly benefit from previous work. Similarly, the somewhat earlier work of various authors in detailing the principles and theories of graphic design have made their contribution (Bertin 1983; Bowman, 1968; Tufte, 1983, 1990; Schmid, 1983). In addition, various research detailing the cognitive psychological aspects of perception and graphics usage (Goldstein & Lamb, 1967: highlighting/luminance, retinal indexing perceptual operators; Green & Anderson, 1956: colour; Julesz, 1981: texture; Neisser, 1964: shape; Kahneman & Henik, 1981: vision colour) has been important to the research development.

There has also been the body of research in the computer science field that has utilized the background work to address a range of problems. Information presentation system are, for the most part, intelligent systems. They are intelligent in a specific way that has been categorized by Edmonds (1993) based on earlier work (Edmonds, 1987; Malinowski et al., 1992). In Edmonds' taxonomy, the intelligent behaviour is in the category of the "services available to the user within the system", in particular, intelligent user interface services. IPS developers have had to utilize the

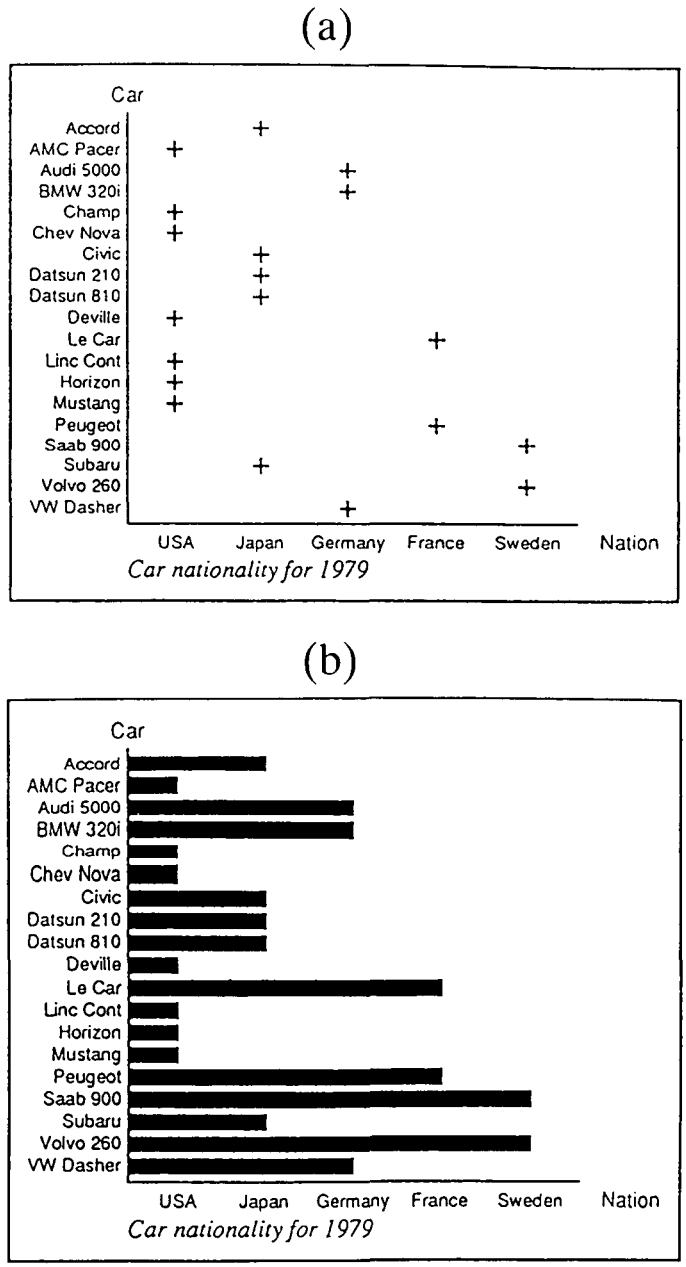


**Figure 1** (a) Correct use of a scatter chart, (b) incorrect use of a bar graph (from the original paper by Mackinlay, 1986)

power of intelligent systems to address a complex and knowledge intensive problem. Thus, advances in knowledge representation, system architectures and general AI techniques have all contributed to the development of IPSs. Advances in technology have also been significant, in particular, the development of raster graphics and the bit-mapped colour screen, as well as the many orders of magnitude increase in machine processing power.

The actual realization of working systems has involved a great deal of innovative research and system development that still continues today. Essentially, the researchers have had to formally articulate graphic design theories into working computer algorithms, and thus make explicit their graphic design criteria. It is important to note that our understanding of the issues is far from complete, and all systems make compromises to achieve an operating graphic presentation system.

Systems that involved automatic graphic presentation were being proposed in the early 1970s, for example, Bartlett and Smith (1973). They applied facilities-allocation algorithms developed for engineering problems, such as plant layout, to the problem of locating an optimal layout of a set of

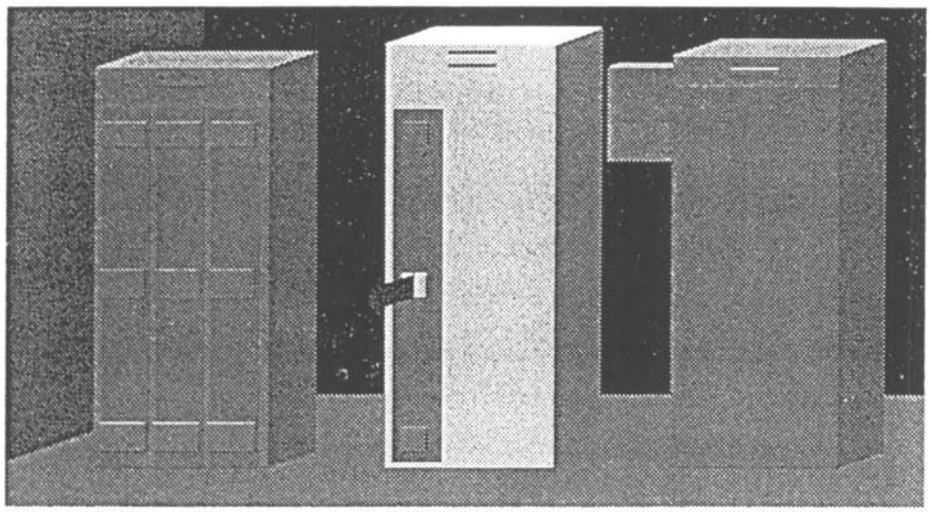

**Figure 2** APEX: Opening the centre cabinet's drawer (from the original paper by Feiner, 1985)

dials and controls on an instrument panel. The program co-ordinates usability factors such as eye travel and hand distance between instruments, frequency of use and accuracy of acquisition.

It was not, however, until the early 1980s that specific information presentation systems started to be described in the literature. Approaches to automatic presentation have varied along a continuum which reflects the degree to which presentation knowledge is domain-independent or specialized. Thus, systems range from general IPS such as AIPS (Zbydel et al., 1981) and BOZ (Casner, 1990) to application-specific ones, such as VIEW (Friedell, 1983), Presentation Manager (Arens et al., 1988; 1991) and CUBRICON (Neal & Shapiro, 1991).

The development of these systems can be seen to divide into two main areas that reflect the division between the generic and the application-specific. The generic approach has been represented by several important systems; of particular note is APT (Mackinlay, 1986). APT provided the basis for many of the presentation systems subsequently developed. Mackinlay established that combinations of conventional presentation graphics can be designed automatically using domain-independent graphical description languages. The alternative, application-specific approach, has seen systems that offer a range of sophisticated features; for example, CUBRICON and APEX (Feiner, 1985).

Information presentation systems are usually attached to databases or knowledge bases. An IPS involves the abstraction of the display function into a separate tool that is part of the overall user interface. Significant advantages accrue at both the conceptual and practical level because of this separation; for example, the ability to encode a consistent set of human factors decisions about graphic displays and store them in the IPS. Mackinlay categorized the work in IPS as representing one or more of three areas; content selection, graphic design issues and range of graphic designs available to a system. In general, this is still a useful categorization:

- *Content selection:* primarily a domain-specific activity, thus the application-specific systems tend to concentrate, particularly the early systems, on this area.
- *Graphic design issues:* an important generic requirement of all IPS that involves a range of issues that are listed below:
  - — data characterization: that is, to identify a set of characteristics that are relevant to the domain and which are able to be encoded. The range and formalization of these characteristics has changed and been extended considerably since Gnamangari's early work, described below.
  - — representation of graphic design knowledge: originally this knowledge was implicit; however, later explicit formalization enabled far greater sophistication of graphic creation, for example, Mackinlay's expressiveness criteria.
  - — selection criteria; for example Casner's task analytic criteria (see below).

— theories of presentation; for example Mackinlay's information analysis theory (see below).

- *Range of graphic designs:* early systems were limited to simple bar and line graphs. Mackinlay employed a combination algebra to extend the ranges of graphic designs. However, further significant developments have been reported, particularly in application-specific systems with a sophisticated range of graphical multimedia images.

The remainder of this paper discusses the development of IPS in two sections, reflecting the division of generic and application-specific systems. The development is traced from the earliest complete systems through to the current representatives of each type. The main development themes are discussed, and future research areas are identified. A summary of the systems discussed is given in Appendix A.

## 2 Generic Information Presentation Systems (GIPS)

Essentially, the basic premise of information presentation as a domain-independent activity is that it is possible to make reasonable decisions about display structure based on limited knowledge about display content (Zbydel et al., 1981). Thus, it is possible to make discriminating choices among alternative display formats on just this limited knowledge. This basic premise was to govern the development of IPS during the early 1980s.

### *2.1 Early work*

Gnanamgari's work BHARAT (1980) was the first system (Mackinlay, 1986) to demonstrate that the above premise was realizable in a working system. BHARAT showed that information characteristics could be used to form the basis for the selection of graphics. The 2D graphics were non-interactive, although the user could alter the display by altering the characteristics of the underlying data.

In BHARAT, the graphic design issues were the governing factors in the system. This knowledge was represented in the system as a set of rules. The set of rules, used for selecting the graphic format, is based on the graphic design principles of Bertin (1983) and Bowman (1968). Gnanamgari identified five characteristics of data that were sufficiently generic to enable them to describe data from a variety of sources:

1. Continuity: a series, e.g. 1,2,3, a,b,c.
2. Totality: the whole of a recognized set, e.g. week is Mon–Sun.
3. Cardinality: number of elements of a set, e.g. week = 7.
4. Multiplicity: number of values assigned to each element in a domain set by a mapping.
5. Units: set of labels defining the units of increment.

The design algorithm was quite straightforward: the system accepted a description of data sets and chose either a bar chart, pie chart or line chart as the presentation graphic. The choice was determined by the data type; a line chart for continuous data, a pie chart for proportional data, and bar charts for all others. The actual graphics used were canned (predefined) graphics. This early work is interesting, but clearly lacks any of the sophistication of later systems. While some of the design knowledge is available as rules within the system, a large part is implicit within the predefined graphic presentations.

The system utilized a set of rules to define the data types and to provide the design knowledge to enable a mapping to the graphic representation. The data was represented in the system as a database table. There was no attempt to employ standard production rules, or other formalisms; rather, the representation mechanisms were procedural rules, and Gnanamgari (1980) identified knowledge representation as the current outstanding area of research work.

AIPS (Zdybel et al., 1981) has knowledge representation as its main emphasis, just as Gnanamgari took the graphic design issues as hers. While Zdybel concentrated on knowledge representation issues, the 2D graphics available in AIPS were interactive. The presentations are

not novel graphics but are a selection from a set of pre-determined graphics that the system has available.

AIPS is a domain-independent system for automatically generating graphical presentations of information stored in a knowledge base. It can present a wide range of information that is coded in the KL-ONE (Brachman & Schmolze, 1985) knowledge representation language. KL-ONE is a frame structured hierarchical network formalism. AIPS matches the KL-ONE description of the data against a pre-defined taxonomic hierarchy of presentation formats and chooses the format that best fits the data characteristics. AIPS is a knowledge based system with the interpreter (written in LISP) distributed over the hierarchy as inheritable attached procedures. In this way, the developers have minimized the extent to which AIPS behaviour is expressed in a programming language, and maximized the extent to which it is declarative. (The intention was to enable the maximum extension of AIPS by a non-programming user.)

AIPS concentrated on raising the level of interaction between a knowledge base and its graphic display function by concentrating on the knowledge representation issue. The developers thus took an AI/intelligent systems approach rather than focusing on the graphic design issues that must, however, underpin any serious fully functional graphic design system.

The graphic design issues were directly addressed by Beach and Stone (1983), who describe a system for producing illustrations using a set of graphics style rules. The system takes a textual description of the geometric properties of an illustration and applies style rules to create an illustration. The rules, called a "graphical style sheet", can be used to provide a uniform "look" over a range of illustrations. It provides a means of capturing the knowledge of graphic arts standards, expert typographers, etc. Complicated effects such as two-dimensional shadows, arrows and complex border patterns are available. In this way, each user of the system can gain access to this level of sophistication without the need to learn the large body of domain knowledge necessary to achieve the same effect from scratch.

The emphasis of this system is on graphic design issues and achieving a high standard of presentation. The "style" sheet allowed explicit description of the graphic design features of the illustration. It was also possible to combine the default style with some input from the user to cover just those aspects that were of particular interest. It is interesting because it captures designers' knowledge, and it enables the graphic design algorithm to be made explicit and thus editable. While BHARAT looked at the generic design problem, Beach and Stone concentrated on the much smaller area of document illustration.

The system was developed using the Cedar programming environment and language, the latter derived from the MESA (Mitchell et al., 1979) language. The document is represented as a tree structured hierarchy of nodes, where each node represents a page of the document or an illustration description. Each document has an associated file of style rules. The design knowledge is represented in the system as rules using an interpreted rule language to enable computation while formatting.

These early systems have several limitations:

1. They used only pre-determined (canned) graphics: pre-determined graphics, a bar graph for example, have already been designed and the presentation program just slots in the values for a particular instantiation of the graphic. Therefore, the design knowledge and human factors knowledge is *implicit* in the code that was developed by the original programmer.
2. No selection of data: all the available data is displayed, whereas later systems enable the tailoring of the selection of a sub-set of the data based on the user, task or other criteria.
3. Information analytic graphic selection: characteristics of data are the sole determinant of the preset type of display methods/graphics. This results in the use of the same graphic for a particular piece of data irrespective of any other criteria (same data same graphic whatever the context).
4. Limited range of displays available, no underlying theory of presentation involved except as implied in the selection of the original design of the objects.

### 2.2 *Later work*

The work in the field over this time was dominated by Mackinlay's basic approach. Therefore, Mackinlay's work will be described in some detail in the first part of the section, and then work that has developed and extended this approach will be described. The final part will look at alternative approaches.

#### 2.2.1 *Mackinlay*

The development of subsequent systems has addressed some of these limitations. Between 1981 and 1985, several notable systems (VIEW, APEX) were developed, but these were in the application-specific area and will be discussed in the next section. Mackinlay's APT (Mackinlay, 1986, 1991) was a significant development, and has shaped every IPS that followed, up to the present day (Casner, 1990).

APT is an automatic graphical presentation tool that designs non-interactive graphical displays of relational information. The significance of APS is that it formally characterizes the fact that graphic presentations can be expressed as sentences in a formal graphical language having a precise syntax and semantics in the form of propositional formalisms. The advantages of having a formalism for graphical presentations is that it provides a set of criteria for deciding the role of each visible sign or symbol placed in a graphic, and improves the integrity of a graphic by providing a formal method for mapping/transforming relational facts into graphic facts.

Mackinlay identified two problems that APT had to address: the inclusion of graphic design issues, and providing a wide range of designs. Graphic design issues were codified as *expressiveness* and *effectiveness* criteria for each graphical language. This provides a set of criteria for evaluating the contribution of each element of the graphic and making decisions about its inclusion or not.

Mackinlay identified a set of primitive graphical languages derived from using Bertin's (1983) vocabulary of graphical encoding techniques. These languages were classified by their primary encoding technique:

1. Single position languages: encode by position of mark on single axis (Y or X axis).
2. Apposed position languages: encode by position of mark between two axis (line, bar, plot charts).
3. Retinal list: colour, shape, size, saturation.
4. Map: road map, topographic map.
5. Connection: tree, acyclic graph network.
6. Misc (angle..): pie chart, Venn diagram.

Analysis of the semantic properties of these languages leads to the "expressiveness" criteria, that is, what each can represent. For example, a single position language can represent binary relations with a functional dependency and line charts represent relations that describe a continuous function. "Expressiveness" criteria determine whether a language can express the desired information, based on an analysis of the semantic properties of the language.

Effectiveness criteria are based on the cognitive psychological work of Cleveland and McGill (1984) that has been extended by Mackinlay. The criteria compares the effectiveness of different graphical presentations in cognitive psychology terms. It provides a measure of how well the presentation conveys different types of information. For example, position has a higher ranking than area for encoding quantative data.

The provision of a wide range of graphic designs is solved using a composition algebra approach, which uses a collection of primitive graphical languages and composition operators to form complex presentations (Mackinlay, 1986). APT's design algorithm utilizes an information analytic approach, and does not consider other factors such as the user tasks or a user model to govern the design of the presentation.

APT was developed on a Symbolics LISP machine using MRS, a representation and logic programming system (Russell, 1985). APS was composed of two parts; the rendering system that

was developed in LISP; and the logic program that implemented the synthesis algorithm. The set of graphical languages, their expressive and effectiveness and the composition operators form the basis of the logic program that consisted of some 200 rules. The program's controlling algorithm is a depth-first backward chaining implementation of the Residue (Finger & Genersereth, 1985) deductive algorithm.

Interestingly, Mackinlay opted for the declarative precision of logic programming, although acknowledging that the actual implementation was somewhat slow. Clearly, with far more efficient implementations of logic programming languages this would be far less of a problem today.

In the period 1986–1989, further application-specific systems were reported in the literature (see sec. 3 for details). Also at this time, a major effort was directed towards the development of automatic presentation as part of UIMS (Edmonds & Murray, 1992), which are outside the scope of this paper.

*2.2.2 Post Mackinlay*

While the basic information analytic approach to presentation selection had been successful, there are other factors that influence the suitability of the presentation, and one of the most important is the task for which the user requires the presentation (Casner, 1990). Until 1989, the original premise of basing graphic design/selection on data characteristics (however sophisticated) had been adhered to and expanded on, but fundamentally not changed since Gnanamgari and Mackinlay. However, with SAGE (Roth and Mattis, 1990) and later BOZ (Casner 1990), the criteria and sophistication of graphic selection took another major step forward with the addition of user task information. SAGE used informational goals while BOZ used task goals of what the graphic is to support to govern the selection, and thus provide much more sophistication.

SAGE (Roth and Mattis, 1990) is described by its authors as an explanation system that uses text and graphics to explain changes that occur in quantative modelling systems such as spreadsheets. However, the authors' stated intention is to create a system that is composed of components that are as "generic and application independent as possible" (Roth et al., 1991). Thus, the development of the presentation component has taken the fundamental generic approach of Mackinlay in APT, albeit considerably extended (Roth and Mattis, 1991). A separate text discourse planner has also been developed along similar generic lines. Thus, the level of independence of the presentation components qualifies it for inclusion in this discussion of generic IPSs.

In SAGE, explanations are generated in response to user queries. A language has been developed to enable the user to specify query and information goals. The application database is a frame-based one concerning project management; for other example applications see Roth and Mattis (1991). The presentation design algorithm takes into account the goals of the user, employing a small set of task operators to enable this selection to be implemented. This is an important departure from the previous wholly information analytic approach to selecting a suitable graphic.

The work on SAGE has extended the original work by Mackinlay in all three areas; the data characterization has been extended, a taxonomy of data characteristics has been created, and the composition operators have been extended. The idea of informational goals has also been added to the graphic selection process. These goals refer to the function of the presentation and the distribution of data within the presentation. For example, where accurate value look up is required a text table would seem most suitable (Roth & Mattis, 1990). The extended data characterization is employed, as with APT, and combined with the informational seeking goals in the definition of *expressiveness* criteria and the relative *effectiveness* of graphical techniques. Thus a greater level of refinement is possible and a wider range of graphics is available. SAGE's use of users' informational seeking goals is particularly interesting (cf. Casner, 1990), although only a small set are implemented in SAGE. It does mean that a more powerful selection mechanism is available to the system.

The text and graphics parts of the presentation are combined to provide a multimedia presentation system. A fundamental problem of systems employing multiple media presentations

is the effective coordination of the media. SAGE's initial problems of coordinating the two modes required the authors to address this problem, and in providing a solution to it they have developed an architectural model for multimedia coordination.

The system has been developed using LISP and Knowledge Craft, and employs an object oriented approach. In the graphical presentation component the object classes represent graphic design components. For example, graphemes represent marks, bars and text; encoders represent axes, keys, coordinate systems and network nodes; and symbols represent clusters of graphemes which express facts. This approach is combined with a language for assembling the objects into graphic designs which employ a set of synthesis rules (Roth, 1993).

SAGE is important because of the development of the idea of informational goal/task goals. Clearly, this is a sophisticated increase in the factors used to govern graphic selection and generation. In addition, it provides an interesting solution to the media coordination problem. This work is complimented by Casner, who develops the informational goals/task goals idea to a much more sophisticated level.

The usefulness of a graphic presentation is heavily influenced by the task it is to be used for (Casner, 1990). There is a growing body of research that shows that the basic premise (that graphics help/aids task completion) is not specific enough (Jarvenpaa & Dickson, 1988). In particular, it is suggested that graphics can only help in a task-specific manner. Thus, one needs to clearly identify what task the graphic is intended to support, and then to design the graphic appropriately. It is on this premise that Casner's work is based. It utilizes a prescriptive task-based design theory for graphic presentations.

BOZ is a system for generating presentations, the design of which are based on an analysis of the task for which the graphic is required. At present the task description must be entered by the user. BOZ analyses a logical description of the task to be performed by a human user and designs a provable equivalent perceptual task by substituting perceptual inference steps in place of logical inferences in the task description. It then analyses the data required to be displayed and, in combination with the set of perceptual operators, creates a description of the perceptual procedures and their graphics. In addition to the graphic, BOZ also produces a perceptual procedure describing how to use the graphic to complete the task.

The system achieves this by utilizing a logical task description language employed by the user (e.g. interface designer) to describe the information processing tasks to be supported. This is used to identify the individual problem-solving steps (logical operators) required to complete the task. BOZ uses a database containing perceptual operators and their logical operator equivalents to perform "perceptual operator substitution". The perceptual data structuring component determines the optimal grouping of data in a graphic. Then perceptual operator selection is carried out to produce a description of each perceptual procedure and an accompanying graphic. This is then used by the graphic presentation rendering component to create the specified graphics.

The result is that BOZ can design a different graphic presentation for the same information based on the task requirements. Thus, where these are different, the resulting graphic presentations will differ. This provides a much richer, broader range of graphic presentations that can be designed by the system. It also produces much more effective presentations, as the utility of a graphic is task dependent: this is the premise on which BOZ is predicated.

The system was developed using InterLisp-D on a Xerox 1186 Lisp machine. The graphical objects were represented as association list schema, with each sub-list holding the value of a perceptual dimension. Production rules were used to represent the search and inference operators (Casner, 1993).

### *2.2.3 Other approaches*

Marshall (1993) describes a system, Playfair, that supports the user in selecting suitable graphical presentations of business graphics (e.g. line charts, scatter plots, etc.). The system uses an interesting combination of design templates (cf. Beech & Stone), a rule-based system for evaluating the suitability of the selected templates for a particular data set, and the inclusion of the

user in the presentation selection cycle. The system is able to present to the user with the templates and to provide guidance in selecting the most suitable. This is a considerably different approach and reflects trends in other areas (Clarke & Smyth, 1990; Edmonds et al., 1993; Fischer, 1993) to provide support to designers rather than replacing them in the design process. However, it does mean that the user requires sufficient design experience to make the optimum selection.

WIP (Rist & Andre, 1993) is a multimedia document presentation system. The developers have taken a speech-act theoretic approach (Searle, 1969) with the generation of a multimedia document being considered as an act sequence that aims to achieve certain goals (Rist & Andre, 1993). Document presentation is a hierarchical structure of elementary speech-acts that results in a complex presentation act. A set of coherent relations has been established that hold between parts of a document; for example, text that provides an interpretation of a picture.

The system uses a set of multimedia presentation strategies that are employed as operators in the presentation planning system. The system creates a hierarchical structure plan of the document, the leaves of the plan represent specifications of elementary acts of presentation. The planner coordinates the operation of the various separate media presentation tools; a text generator, a graphics generator and a layout component, to produce the final presentation. The system supports content selection based on user characteristics and the presentation situation, together with media selection and content realization. Recognizing that a more generic approach would be desirable, the authors have not tied their system to a single application area, and state that "WIP has been designed for interfacing with various back-end systems" (Rist & Andre, 1993).

WIP has been developed on Symbolics XL 1200 and MacIvory workstations using Symbolics Common Lisp under Genera 8.0. WIP is an interesting system that uses sophisticated content and media selection and coordination to provide an essentially generic, rather than application-specific, systems approach.

Vibhu et al. (1993) describe another system that takes the speech-act approach to document generation. However, Vibhu et al. are concerned with the somewhat different area of automatic generation and integration of text and examples to generate on-line documentation. This is part of the Explainable Expert System framework (Swartout et al., 1992), which is a framework for building expert systems capable of explaining their reasoning as well as their domain knowledge.

## 3 Application-specific information presentation systems

Application-Specific Information Presentation Systems (ASIPS) were developed in response to the problems of providing greater sophistication at the generic level. Thus, by limiting the domain of interest the problems of providing more sophisticated systems became tractable. The development has been in three main areas:

- the range of criteria used to select a suitable graphic;
- the range and integration of the media employed;
- the explicit representation of graphic design knowledge in the system.

The early systems did not have graphic design issues as their central concern, rather the correct selection of data was their major focus. The criteria used for graphic selection tended to be quite limited. Similarly, the media employed in early systems was limited to text and graphics.

Clearly, these systems did not just spring up from nowhere; they represent the culmination of a lot of previous work. For example, APEX (Feiner, 1985) was based on work in an earlier system (Feiner et al., 1982). Similarly, there was a lot of early work on constraints before Borning produced his system. However, the aim of this section is to look at those systems that represent a utilization of the background work, and which represent important developments in the area.

The application-specific systems developed have, almost exclusively, addressed two areas; military information presentation and explanation generation. There were exceptions, however, and Borning's work with constraint-based systems was perhaps the most notable. This section will

be organized around these three areas, rather than purely chronologically, to provide a more coherent structure.

### *3.1 Constraint-based systems*

ThingLab (Borning, 1979) was developed in the object oriented language Smalltalk (Goldberg & Robson, 1983). The system is not a specific IPS in the terms of VIEW or APT, but it has previously been classified as an IPS (Zdybel et al., 1981) and does perform the automatic presentation of data. ThingLab involves extensions to Smalltalk consisting of a constraint-based toolkit for building various graphical presentations: geometric demonstrations, simulation of physics experiments and user interface frameworks are some examples of use. The toolkit enables developers to put together a set of presentation objects that the end user can then employ to model their domain of interest. ThingLab contains a general constraint satisfier featuring incremental compilation of methods for satisfying constraints.

A constraint is a relation that is established between objects that must be maintained by the system. How it is maintained is up to the system. For example, dynamic display is achieved because once a constraint is established between an on-screen object and an application data value, if either changes, the other is changed to reflect this; the system does this automatically. There have been a range of systems developed using constraints from Sutherland's early system in 1963. There have also been a range of approaches to implementing constraints Levitt (1984) provides a survey of constraint languages.

ThingLab is of note for several reasons: it was created in 1979, it used the constraint to control the presentation, it was very extensible and it provided an interactive display with dynamic graphical presentation. However, Borning acknowledges the debt to Sutherland and his much earlier system employing constraints, Sketchpad (Sutherland, 1963), a remarkable system, particularly given the power of computers at the time at which it was developed.

Much of the development of constraint-based system has been in the area of user interface construction tools, from early work described by Borning and Duisberg (1986) to sophisticated user interface development environments described by Myers (GARNET: Myers et al., 1990). The other area of note has been "programming by example", which includes PYGMALION (Smith, 1975) and Peridot (Myers et al., 1989).

### *3.2 Military presentation systems*

The first part of this section looks at early work in the area, while the second part discusses later developments.

#### *3.2.1 Early work*

The rest of the ASIPS discussed have concentrated on the two areas mentioned above. Probably the earliest example of a military information presentation system is VIEW (Friedell, 1983). This creates graphic presentations (2D icons) of naval information about the properties of ships, in response to user queries. It was able to select a suitable sub-set of data for display from the available data, rather than display all the available data (content selection is a function common to all the systems in this section). Of particular importance is the ability to utilize principles, other than simple information analytic (based on an analysis of the data) ones to select presentation formats. The system uses a knowledge base of presentation formats encoded using KL-ONE (Brachman & Schmolze, 1985) that includes user and task characteristics in the description. Thus, selection is not only based on the nature of the data, but also on the current user-specified task, the nature of the query and the identity of the user. This produces a much more sophisticated selection criteria and enables different presentations for the same data. This is possible because the system is domain-specific, and there is only a need to define domain tasks rather than to identify and represent generic tasks.

However, as in AIPS and BHARAT, VIEW can only select from a set of pre-determined graphics. The process involved generating icon templates from sub-icon templates. These had to be designed by hand rather than by the system. It does not design novel presentation formats of its own, although the selection criteria enables it to generate a different graphic for the same data in different contexts. The combination of selection criteria made VIEW quite sophisticated. Other work at this time (for example, AIPS and BHARAT) based selection on information analysis alone. While there has been a steady increase in the sophistication of such systems, they are still based on much of what Friedell did in VIEW.

*3.2.2 Later work*

CUBRICON (Neal et al., 1988, 1991) extends military presentation systems by utilizing a multimedia approach. The media available are colour graphics, tables, text and spoken language. CUBRICON is based on extensive knowledge base of application-specific data. In addition, detailed task description are used by the system. In response to a user query, the system makes several basic decisions. A subset of the data available relevant to the user task must be selected. Suitable media must then be selected (either one or more) to present the data. Having selected the appropriate media there are a limited number of pre-determined presentation formats available for the system to employ. Overall, the system uses guidelines based on information type for presentation design. However, it does support an interactive interface, allowing user input with mousing and utilizes a wide range of interface media, most notably spoken language.

The system uses the SNePS (Shipiro, 1987) fully intentional propositional semantic network processing system. This provides the semantic network formalism used in the three main knowledge bases that form CUBRICON. The declarative representation enables the rules to be reasoned about, like any other data. It also employs a formal grammar (ATN) and a lexicon defining the language used to describe multimedia input and output. This combination provides the basis of a powerful system that is able to handle combined multiple input and output modalities.

Integrated Interfaces (Arens et al., 1988, 1991) is a set of tools integrated around an application-specific knowledge base (naval ship movements) that provides a multimedia information presentation system. This is a rule-based multimedia system. The system can produce multi-modal output, i.e. natural language (Penman system), graphics/maps (GDA), tables, forms (QFORMS). It employs a model-based theory of presentation design. Thus, there are models of the application domain, the interface and a final model of functions and data structures of available applications; these are integrated in the knowledge base.

There is a set of rules to map from the application model to the presentation model, to identify the type of data (i.e. information analytic; but the data here is quite complex in comparison with Mackinlay's much simpler relational data), and to find the most appropriate presentation design from the knowledge base. A stated aim of the system is to simplify the production of multimodal interfaces, whereas CUBRICON, for example, takes an intelligent systems approach to the aim of research into new integrated communication modalities.

The system employs two knowledge representation systems: NIKL (Kaczmarek et al., 1986) and KL-TWO. NIKL is a hierarchical network system based on KL-ONE (Brachman & Schmolze, 1985). In essence, NIKL is a frame system with formal semantics that produces a rigorous representation that can be directly translated into predicate calculus expressions from which a theorem prover can make the same inferences. However, NIKL is optimized for the limited inferences it makes, and is therefore more efficient. NIKL is required because of the complexity of the data that the system has to handle. It is also used to hold the application and presentation models.

KL-TWO is a hybrid knowledge representation system that combines NIKL's formal semantics with a propositional logic reasoner PENNI (RUP; McAllester, 1982), thus forming a propositional semantic network. PENNI manages a propositional database and includes a truth maintenance system that keeps track of the facts used to deduce others. The addition of NIKL to PENNI provides the ability to handle taxonomic reasoning.

Integrated Interfaces' major importance is in its use of the model-based approach, i.e. the mapping from one model to the other using rules to achieve the presentation. It is also of importance because of its combination of representation formalisms to achieve a powerful and rigorous representation system that can handle the inherent complexity of the data and tasks. The system demonstrates that multiple modes can be treated as an integrated whole. Also, it demonstrates that the selection of suitable media can be represented by explicit rules which relate information to be presented to the method of presentation.

### *3.3 Explanation systems*

The first part of this section will look at the early work in the area, while the second part will discuss later developments.

#### *3.3.1 Early work*

The other main application area, explanation systems, has several examples, the earliest of which is Feiner's explanation system APEX[1] (Feiner, 1985, 1991). This creates graphical explanations of how to use various physical devices, and employs 3D in the presentations. The explanations are generated by a Problem Solver (AI system). The Problem Solver contains rules about the known tasks and knowledge about the current state of user task awareness. Each task is composed of constituent actions that are represented using a frame structure. An explanation consists of several frames that depict the sequence of steps in a particular action. Input is not supported.

There are two basic parts/elements to APEX that enable it to function. There is a frame-based database of objects known to the system. Objects are hierarchically structured as trees of 3D parts, where each level in the hierarchy contains more details about the objects. The second part employs rules to enable the system to determine how much detail is needed in each frame and to limit the display to that information. In addition, rules are used to determine which rendering style is to be used and which viewing specifications will be employed. The objects are highly domain-specific, and therefore must be created before APEX can be used. Presentation is quite sophisticated, in that frames and sequences of frames are created which combine text and icons. In Mackinlay's terms, content selection, graphical techniques, graphic design issues of managing icons and 3D images are all considered. It is the combination of selection criteria, graphic design issues and the sequencing of frames, that makes APEX of interest. APEX is a rule-based system written in LISP, and interfaces to problem solvers written in Frail and Micro-Nasl (Charniak et al., 1983). It also employs a frame-based representation for the database of known objects. The rendering software is written in C. The combination of rule- and frame-based representation reflects the increase in sophistication in this area over the earlier systems, and sets the pattern for many of the subsequent systems.

#### *3.3.2 Later developments*

COMET (Feiner & McKeown, 1990) is an explanation system that uses interactively generated text and 3D graphics to explain repair and maintenance tasks. It employs another interesting idea, that of communicative goals to govern the generation of graphics. COMET is composed of several different modules. The Content Planner is media independent and selects a set of communicative goals and an associated set of information. These are represented as a hierarchy of Logical Forms (LF) (Halliday, 1985). The LFs are passed to the Media Co-ordinator which selects the presentation format for each action and fact and annotates the LF to reflect this selection. There is a set of predetermined formats and display decisions that are based on the six categories of information

[1] The basis for some of what is found in APEX was described in an earlier paper (Feiner et al., 1982) that describes a system for creating and presenting interactive graphical documents. The emphasis is on separating out the various functions of the system, one of which is the presentation function, that were to form the basis of the APEX approach.

recognized by the system (Lombardi, 1989). The LFs are then passed to the separate media generators. The text generation is handled by a separate tool, FUF (Elhadad, 1988). The graphic generation part of the explanation is created using another tool, IBIS (Seligmann & Feiner, 1989). A Media Layout component then combines the output of the two generators for the completed presentation. The emphasis is on fine grained media co-ordination (Feiner, 1991).

IBIS uses *communicative goals* represented by the LF to guide the generation of the graphic presentations. Communicative goals utilise the cognitive psychological aspects of graphic design. Thus components can be highlighted or not, to focus the viewers attention. They are used to describe the presentation characteristics of individual components of a device that can be manipulated. For example, *include* or *highlight* an object are both primitive communicative goals. IBIS can generate composite pictures that included hierarchically nested sub-pictures. It uses the knowledge base of known objects common to the rest of the system, and employs a rule-based control component which builds and evaluates a representation of the presentation. The completed presentation is then passed back to the Media Layout component.

This is a sophisticated system that builds on earlier work by Feiner (1985), with the emphasis on fine grained media coordination. The system employs an object oriented approach and uses C++. Represented objects are both physical objects that can appear in a presentation, and logical objects, such as the illustrator objects associated with each presentation. This object oriented approach is combined with a rule-based representation, using the CLIPS production system language (Culbert, 1988) for the control components and graphical knowledge.

## 4 Discussion

The developments in IPSs discussed have shown that the concerns of the generic IPS developers has remained largely the same; graphic design issues and the range of presentations available, to the exclusion of the third of Mackinlay's categories, content selection. Clearly, because it is an application-specific activity, content selection will remain outside the remit of generic systems, unless some suitably generic classification of tasks, user and data can be developed to allow the identification of a relevant sub-set of the data (possibly in the style of CUBRICON's content selection mechanism or the WIP approach).

However, within the areas of concern, considerable advances have been made over the period reviewed. Thus, for example, the range of graphic designs available have been extended considerably from the earliest system that only used bar, pie and line charts, through APT that employed a composition algebra and primitives to create a wide range of designs, to BOZ, that by adding task-analysis to the APT approach further extends the range of graphic presentations that can be generated by the system.

The underlying theory of presentation used by such systems has also seen a considerable increase in sophistication. Originally, simple principles such as "graphical is best" or "if the data is continuous a line graph is most suitable" have been replaced by more detailed theories of presentation. APT employs a theory based on an extensive analysis of the information to be presented. BOZ takes this considerably further by developing a task-analytic theory of presentation; although utilizing basic information analysis, it considerably extends this with task-dependent criteria.

The presentation theory underpinning a system provides the basis for both the nature of the data characterization required by the system and also the criteria used for selecting the graphic presentation to be used to represent the information to the user. Thus, as the theories of presentation have increased in sophistication, so, necessarily, has the sophistication of these other two categories. The one area that has remained the same has been the representation of graphic design knowledge. Mackinlay's original expressiveness and effectiveness criteria have remained the standard for representing this type of information to the system, because they are an economic and effective solution; albeit that there have been extensions to the original categories and levels used.

The earlier Application Specific Information Presentation Systems (ASIPS) had a limited concern with graphic design issues, although clearly, they often had sophisticated design algorithms. Content Selection was an important area of concern for early systems such as VIEW and APEX (Mackinlay, 1986), although APEX did consider the graphic design issues of merging icons and 3D images. However, an increasing concern has been shown by ASIPS developers in graphic design issues. This is not to suggest that Content Selection is not a major area of development; it has seen increasing levels of sophistication employed. For example, CUBRICON uses a combination of user and task models to select relevant information items/entities using a rating system to represent entities importance to the current task.

Systems such as SAGE have actually taken Mackinlay's original encoding of graphic design issues and developed and extended them. There are three main areas that have shown particular advances. The first is the range of media employed in the systems. Clearly, it is not enough merely to provide access to a range of media. It is the theories and models that govern the employment and integration of media that is important. Thus, advances have taken place in the utilization of multimedia; the coordination of multimedia and the sequencing of frames of data and graphics.

The second area is the employment of various principles to govern graphic selection and presentation. Early systems relied solely on data characterization. However, while this has remained a basic criteria, it has been combined with various others; for example, user identities, user tasks, user informational goals and communication goals. Clearly, the quality of the presentation will rely heavily on the sophistication of the criteria used to govern the selection of the presentation format and the media used to deliver it to the user.

The third area is about the way in which the concern with graphic design knowledge, and particularly the explicit representation of graphic design knowledge in the system, has become of increasing importance. Major representations have been style sheets, rules and the formal *expressiveness* and *effectiveness* criteria (Mackinlay, 1986), which have been extended in the SAGE system.

All but the earliest systems described in this paper have taken an AI/intelligent systems approach to developing automatic presentation systems. Even the very earliest system (BHARAT) identifies knowledge representation as a critical issue in the development of IPSs. Consequently, knowledge representation (KR) has been of particular importance, especially where several different types of knowledge must be represented. These types include, for example, knowledge about items to be displayed and knowledge about the relationships between these items and suitable presentation techniques. Clearly, these classes of knowledge place different requirements upon KR formalisms.

The early systems took a wholly declarative approach; AIPS, for example, employed KL-ONE and limited the interpreter to discrete LISP procedures. Mackinlay took an alternative approach and utilized the declarative precision of a logic program (employing some 200 rules) to represent his graphical languages and synthesis component. APEX was to set the trend for knowledge representation that is still the main approach employed today. This involved exploiting the relative strengths of different formalisms and combining them in the system. Structured object representations have been used for representing the objects known to the system, and production rules, or logic-based representations, used for control programs and graphical knowledge. Feiner (1985) employed LISP to represent the relations and a frame-based formalism for the objects known to the system.

There is an increasing formalization and sophistication of the knowledge representations used in the systems described. This development can be traced through the systems, thus SAGE, for example, combines text and graphical presentations, and employs an object oriented approach (C++) for the known objects (actual and virtual) and production rules (CLIPS) to govern the control. Further sophistication was required by the inclusion of natural language, and wide ranging multimedia systems that resulted in the combination of semantic network representations with propositional logic to obtain a more rigorous formalism. The utilization of this formalism has been further extended in the CUBRICON system with the inclusion of truth maintenance system. As

can be seen from the above, structured object representations have provided a common basis of KR in a majority of these systems. This reflects the concern with static data (Bench-Capon, 1990) such as known objects and their semantics, the semantics of data, graphical representations or the various media employed.

The development of these intelligent systems has reflected, and will continue to reflect, the main stream developments in AI/intelligent systems, particularly with respect to the utilization of knowledge representation formalisms.

### *4.1 Further research*

The utilization of multimedia generates many problems; of particular importance is the need for the coordination of the various media. There have been several solutions adopted in the systems described with, perhaps, CUBRICON's media language being the most complete. However, the seamless integration of a wide range of media has still to be achieved.

User interaction with the system in some manner has, surprisingly, not figured largely in the systems described. A notable exception was the very early system AIPS that provided an interactive display. However, it was not until CUBRICON provided for input presentations that this interaction was again provided. SAGE provides a further sophistication in a query style language that enables the user to tailor the presentation by refining their information goals. A further dimension to user interaction comes from voice input/output: CUBRICON is alone in currently providing voice input/output. However, this and other types of interaction and subsequent control will be another important area for future research.

The assignment of graphical presentation techniques and media to data will have to be done using far more sophisticated criteria if there is to be exploitation of the full communication bandwidth of multimedia presentations. Useful starts have been made with communication/informational/task goals, but these are only a good beginning.

Content selection is another area of future research. Currently, it is the sole preserve of the application-based systems, largely because of the difficulty of finding some suitably generic classification of tasks, user and data to allow the identification of a relevant sub-set of the data. (Possible solutions may be in the style of CUBRICON's content selection mechanism.)

Finally, access to the design knowledge in the system is an important aspect of current work in intelligent UIMS (Edmonds & Murray, 1992), because by providing access to this knowledge, developers enable their system to be easily updated with new design knowledge as it becomes available. However, this has not been an area that has received much attention in the IPS field. Clearly, if systems are to be more responsive to users' requirements, then easy and structured access to the design knowledge is an important property of future systems.

## 5 Conclusion

This paper has discussed the development of Information Presentation Systems. It has identified that the work can be seen to divide into two main development paths; generic and application-specific. The major pieces of early work have been discussed, and the significant problems and shortcomings associated with them have been identified. The development of later systems has been traced through to the present day. Various characteristics of the systems have been highlighted and discussed; for example, their use of knowledge representation formalisms. Finally, a set of outstanding research issues has been identified.

**Acknowledgements**

Many thanks are due to E. A. Edmonds and A. A. Clarke for their numerous helpful comments on earlier drafts of this paper. Thanks are also due to the anonymous referees for their comments on an earlier draft of this paper. The work was partly funded by an ESPRIT 2 project FOCUS (2620).

## Appendix A Summary of systems discussed

| *System Name/Type* | *Content selection* | *Display media* | *Graphic selection* | *Graphics type* | *Data to graphic mapping* | *Implementation* |
|---|---|---|---|---|---|---|
| BHARAT - Pres | No | Grph | I.T. | Canned | Fixed | Procedural Rules |
| AIPS - Pres | No | Grph | I.T. | Canned | Fixed | KL-ONE LISP |
| Beech & Stone—Pres | No | Txt & Grph | I.T. | Primitive | Variable | Cedar |
| APT - Pres | No | Grph | I.T. | Primitive | Fixed | Logic Program |
| SAGE - Pres (Expln) | Yes | Txt & Grph | T-G | Canned | Variable | Obj. Orient Rules |
| BOZ - Pres | No | Grph | I.T.-T | Primitive | Variable | InterLisp-D Rules/Lists |
| Playfair Pres | No | Grph | I.T.-Usr | Primitive | Variable | Lisp |
| WIP Pres | Yes | Txt & Grph | T-U | Domain Primitive | Variable | Lisp |
| VIEW - App Pres | Yes | Txt & Grph | I.T.-T-U | Canned | Variable | KL-ONE LISP |
| CUBRICON App Pres | Yes | Txt & Grph | I.T. | Canned | Fixed | Prop. Sem. Net. Rules |
| Integrated Interfaces - App Pres | Yes | Txt & Grph | I.T. | Canned | Variable | Prop. Sem. Net. Rules |
| APEX - Expln | Yes | Grph | I.T.-T | Canned | Variable | LISP Rules |
| COMET - Expln | Yes | Txt | I.T. | Canned | Fixed | Obj. Orient C++ |
| IBIS - Expln | Yes | Grph | C.G. | Canned | Variable | CLIPS |

*System Type*: Pres – Generic Information Presentation; Expln – Explanation; App pres – Application Specific Info. Pres.
*Graphic Selection*: I.T. – Information Type; T – Task; U – User Type; C.G. – Communication Goals; G – User Goals; Usr – User selection.
*Display Media*: Txt – Text; Grph – Graphics.

## References


Arens, Y, Miller, L, Shapiro, S and Sondheimer, N, 1988. "Automatic construction of user-interface displays". In: *Proceedings of AAAI-88*, pp. 808–813.

Arens, Y, Miller, L and Sondheimer, N, 1991. "Presentation design using an integrated knowledge base". In: JW Sullivan and SW Tyler (eds.), *Intelligent User Interfaces*. ACM Press, New York, pp. 241–258.

Bartlett, MW and Smith LA, 1973. "Design of control and display panels using computer algorithms". *Human Factors* **15**(1) 1–7.

Beach, R and Stone, M, 1983. "Graphical style towards high quality illustration". *Computer Graphics* **17**(3) July, 127–135.

Bench-Capon, TJM, 1990. *Knowledge Representation: an approach to Artificial Intelligence*. Academic Press.

Bertin, J, 1983. *Semiology of Graphics*. University of Wisconsin Press.

Borning, A, 1979. "ThingLab-a constraint-oriented simulation laboratory". PhD dissertation, Department of Computer Science, Stanford University.

Borning, A and Duisberg, R, 1986. "Constraint-based tools for building user interfaces". *ACM Transactions on Graphics* **5**(4), 345–374.

Bowman, WJ, 1968. *Graphic Communication*. Wiley.

Brachman, RJ and Schmolze, JG, 1985. "An overview of the KL-ONE knowledge representation system". *Cognitive Science* **9**(2) 171–216.

Casner, SM, 1990. "A task-analytic approach to the automated design of information graphics". PhD Thesis, University of Pittsburgh.

Casner SM, 1993. Private communication.

Charniak, E, Gavin, M and Hendler, J, 1983. "The Frail/Nasl reference manual". CS Tech. Report CS-83-06, Department of Computer Science, Brown University, RI.

Cleveland, WS and McGill, R, 1984. "Graphical perception: theory, experimentation and application to the development of graphical methods". *Journal of the American Statistics Association* **79**(387) 531–554.

Culbert, C, 1988. *CLIPS Reference Manual.* NASA/Johnson Space Center, TX.

Edmonds, EA, 1987. "Adaptability response and knowledge". *Knowledge Based Systems* **1**(1) 3–10.

Edmonds, EA, 1993. "The future of intelligent interfaces: not just 'How?' but 'What?' and 'Why?'". In: Grey, Hefley and Murray (eds.), *Proceedings of International Workshop on Intelligent User Interfaces*, Orlando, FL, 1–8 January, pp. 13–17.

Edmonds, EA and Murray, BS, 1992. "Intelligent presentation and dialogue management". In: Marik, Stepahkova and Trappl (eds.), *Advanced Topics in AI.* Springer Verlag, pp. 465–476.

Edmonds, EA, Candy, L and Murray, BS, 1993. "Knowledge Support Systems for Designers". In: *IJCAI '93 Workshop on Artificial Intelligence in Design, 13th International Joint Conference on Artificial Intelligence.* Kaufmann, pp. 7–12.

Elhadad, M, 1988. "The FUF functional unifier: user manual". Technical Report, Columbia University.

Feiner, S, Nagy, S and van Dam, A, 1982. "An experimental system for creating and presenting interactive graphical documents". *ACM Transactions on Graphics* **1**(1) 59–77.

Feiner, S, 1985. "APEX: An experiment in the automated creation of pictorial explanations". *IEEE Computer Graphics Applications* **5**(11) 29–37.

Feiner, S, 1991. "An architecture for knowledge-based graphical interfaces". In: JW Sullivan and SW Tyler (eds.), *Intelligent User Interfaces.* ACM, pp. 259–279.

Feiner, S and McKeown, KR, 1990. "Coordinating text and graphics in explanation generation". In: *Proceedings of AAAI '90 National Conference on Artificial Intelligence,* Boston, MA, pp. 442–449.

Finger, JJ and Genersereth, MR, 1985. "RESIDUE – a deductive approach to design synthesis". Tech. Rep. KSL-85-1, Computer Science Department, Stanford University.

Fischer, G, 1993. "Turning breakdowns into opportunities for creativity". In: *Proceedings International Symposium on Creativity and Cognition,* 13–15 April, Loughborough, UK.

Friedell, M, 1983. "Automatic graphics environment synthesis". PhD dissertation, Department of Computer Engineering and Science, Case Western Reserve University.

Goldberg, AJ and Robson, D, 1983. *Smalltalk-80: The language and its implementation.* Addison-Wesley.

Goldstein, DA and Lamb, JC, 1967. "Visual coding using flashing lights". *Human Factors* **9**(5) 405–408.

Gnanamgari, S, 1980. "Providing automatic graphic displays through defaults". *Proceedings of Third National Conf. of CSCSI*, May.

Green, BF and Anderson, LK, 1956. "Colour coding in a visual search task". *Journal of Experimental Psychology* **57** 19–24.

Halliday, M, 1985. *An Introduction to Functional Grammar.* Edward Arnold.

Jarvenpaa, SL and Dickson, GW, 1988. "Graphics and managerial decision making: research based guidelines". *Communications of the ACM* **31**(6) 764–774.

Julesz, B, 1981. "Textons, the elements of texture perception, and their interaction". *Nature* **290** 91–97.

Kaczmarek, T, Bates, R and Robins, G. 1986. "Recent developments in NIKL". In: *Proceedings of AAAI-86,* August, Philadelphia, PA, pp. 978–985.

Kahneman, D and Henik, A 1981. "Perceptual organisation and attention". In: Kubovy and Pomerantz (eds.), *Perceptual Organisation.* Lawrence Erlbaum, pp. 181–211.

Levitt, D, 1984. "Machine tongues X: constraint languages". *Computer Music Journal* **8**(1), 9–21.

Lombardi, C, 1989. *Experiments for determining the Assignment of Information to Media in COMET.* Columbia University.

McAllester, DA, 1982. "Reasoning utility package user's manual". AI Memo 667, MIT, Cambridge, MA.

Mackinlay, J, 1986. "Automating the design of graphical presentations of relational information". *ACM Transactions on Graphics* **5**(2), 110–141.

Mackinlay, JD, 1991. "Search architectures for the automatic design of graphical presentations". In: JW Sullivan and SW Tyler (eds.), *Intelligent User Interfaces.* ACM, pp. 281–292.

Malinowsky, U, Kühme, T. Dieterich, H and Schneider-Hufschmidt, M, 1992. "A taxonomy of adaptive user interfaces". In: Monk, Diaper and Harrison (eds.), *People and Computers VII.* The British Computer Society, pp. 391–414.

Marshall, R, 1993. "Improving the quality of computer generated charts". In: Salvendy and Smith (eds.), *Human-Computer Interaction: Software and Hardware Interfaces–HCI International '93.* Elsevier, pp. 597–602.

Mitchell, JG, Maybury, W and Sweet, R, 1979. *Mesa Language Manual.* CSL-79-3, Xerox PARC, USA.

Myers, BA, Vander Zanden, B and Dannenberg, RB, 1989. "Creating graphical objects by demonstration". In: *Proceedings of ACM SIGGRAPH on User Interface Software,* Banff, Alberta, Canada, pp. 95–104.

Myers, BA, Guise, DA, Dannenberg, RB, Zanden, BV, Kosbie, DS, Pervin, E, Mickish, A and Marchal, P, 1990. "GARNET: comprehensive support for graphical, highly interactive user interfaces". *IEEE Computer,* November, pp. 71–85.

Neal, JG, Bettinger, KE, Byoun, JS, Dobes, Z and Thielman, CY, 1988. "An intelligent multi-media human-computer dialogue system". In: *Proceedings of Workshop Space Operations, Automation and Robotics (SOAR'88).* Wright State University, pp. 245–251.

Neal, JG and Shapiro, SC, 1991. "Intelligent multi-media interface technology". In: JW Sullivan and SW Tyler (eds.), *Intelligent User Interfaces.* ACM, pp. 11–43.

Neisser, U, 1964. "Visual search". *Scientific American* **210** 94–102.

Rist, T and Andre, E, 1993. "Designing coherent multi-media presentations". In: Salvendy and Smith (eds.), *Human-Computer Interaction: Software and Hardware Interfaces–HCI International '93.* Elsevier, pp. 434–439.

Roth, SF and Mattis, J, 1990. "Data characterization for intelligent graphics presentation". In: *Proceedings of CHI '90 Conference,* Seattle, WA. ACM, pp. 193–200.

Roth, SF and Mattis, J, 1991. "Automating the presentation of information". In: *Proceedings of the 7th IEEE Conference on AI Applications,* Miami Beach, FL, pp. 90–97.

Roth, SF, Mattis, J and Mesnard, X. 1991. "Graphics and natural language as components of automatic explanation". In: Sullivan and SW Tyler (eds.), *Intelligent User Interfaces.* ACM, pp. 207–239.

Roth, SF, 1993. Private communication.

Russell, S, 1985. "The compleat (sic) guide to MRS". Tech. Rep. KSL-85-12, Computer Science Dept., Stanford University.

Seligmann, D and Feiner, S, 1989. "Specifying composite illustrations with communicative goals". In: *Proceedings UIST '89 ACM SIGGRAPH Symp. on User Interface Software and Technology,* Williamsburg, VA, pp. 1–9.

Schmid, CF, 1983. *Statistical Graphics: Design Principles and Practices.* Wiley.

Searle, JR, 1969. *Speech Acts: An essay in the philosophy of language.* Cambridge University Press.

Shapiro, SC and Papaport, W, 1987. "SNePS considered as a fully Intensional Propositional Semantic Network". In: *Proceedings of AAAI-86,* August, Philadelphia, PA, pp. 278–283.

Smith, D, 1975. "PYGMALION: A creative programming environment". Rep. AIM-260, Department of Computer Science, Stanford University.

Sutherland, I. 1963. "Sketchpad: A man-machine graphical communication system". PhD thesis, Massachusetts Institute of Technology.

Swartout, WR, Paris, CL and Moore, JD, 1992. "Design for explainable expert systems". *IEEE Expert* **6**(3) 58–64.

Tufte, ER, 1983. *The Visual Display of Quantitative Information.* Graphics Press, Cheshire, CT.

Tufte, ER, 1980. *Envisioning Information.* Graphics Press, Cheshire, CT.

Vibhu, O, Paris, M and Paris, GL, 1993. "Automatic document generation: the interaction of text and examples". In: *Proceedings of 13th International Joint Conference on AI,* Chambery, France, pp. 1158–1163.

Zdybel, F, Greenfeld, NR, Yonke, MD and Gibbons, J, 1981. "An information presentation system". In: *7th International Joint Conference on Artificial Intelligence,* Vancouver, Canada, pp. 978–984.