Various Perspectives of Automatic Configuration of Structured Graphical Documents

Vytautas Čyras, Kristina Lapin
Vilnius University, Dept. of Math. and CS, (Vytautas.Cyras,Kristina.Lapin)@maf.vu.lt

Abstract. The paper is aimed at presenting several views to automatic synthesis of structured 2D drawings. We view the subject from different standpoints: document configuration, knowledge representation and document management. Our case study is in the domain of configuration of electroplating lines. As a result of the configuration process, a set of technical drawings is produced. We present an approach (and a software system) SyntheCAD to the preparation of technical drawings for generating layouts automatically rather than building interactively. We treat SyntheCAD as a combination of two tasks: configuration (as in artificial intelligence) and document preparation. The method combines basic concepts of (1) the component oriented approach to configuration, and (2) generic coding introduced in document preparation.

Key words: document preparation, configuration, document management, logical and layout structures of documents, electroplating lines

1. Introduction

In the paper we generalise Vilnius University team’s experience since 1985 in the domain of computer-aided design (CAD) for electroplating lines. The development and implementation of the SyntheCAD system in two design offices in Lithuania and Russia in the past are treated from different standpoints. The efforts are compared within the context of established approaches. The aim of the current research is to place SyntheCAD among established approaches. Examples of drawings produced with SyntheCAD are out of scope of this paper and are presented earlier in [9].

The SyntheCAD as a CAD system has been implemented in different platforms including PDP, VAX and PCs. The SynthCAD approach to synthesis of drawings is derived from the system. We treat the method as a combination of two tasks: configuration (as in Artificial Intelligence, AI) and document preparation. Both mentioned terms can be treated as formal ones.

Structured documents are traditionally associated with textual documents. Logical structure and layout structure of textual documents are addressed in [6]. Graphical documents have a structure, too. According to established structured document processing approaches, a structured document processing system provides the user with the following methods:

- a method to describe the logical structure, and
a method to derive the layout structure from the logical structure.

Two approaches to layout configuration are generally proposed: first, automatic generation, and, second, an interactive approach. The paper addresses the first one — automatic generation of technical drawings.

The paper is structured as follows. First we present different views to document creation. Second, we describe a document preparation task. Third, we present an approach to configuration as implemented in the SyntheCAD system.

We aim to treat the views to document creation in terms of conceptual models of appropriate problem domains.

2. Characterisation of Several Viewpoints to the Production of Drawings

The usage of two-dimensional computer graphics in machine industry can be viewed from different standpoints. Possible standpoints are:

- 2D graphics application. Creating of a drawing is an aim.
- Knowledge representation. Principles of AI are applied in order to represent knowledge about document creation.
- Document management. Document creation is viewed from business perspective.

The granularity of objects handled in each of the processes above is very different. The different views above are consistent with the characterisation of programming in various time periods since the 1950s:

- Programming as you can
- Programming in small
- Programming in large
- Programming in the world.

2.1. 2D Graphics Application

This view dominated 20-30 years ago. As a result computer graphics tools were developed. Software packages appeared first and later interactive computer graphics and CAD systems. The applications are targeted primarily at creating drawings. Thus the task can be characterised as drawing synthesis. Different grades of automating are put in the forefront. The grades of drawing synthesis can be characterised similarly as program synthesis in the 1960s and 1970s including automatic, semi-automatic and intuition guided.

Traditionally business aspects are not considered in the research of synthesis. One of the reasons is to “divide and conquer.” A synthesis task can be formulated simpler under such consideration.
2.2. Knowledge Representation

The key task is the representation of design knowledge. To provide solutions, expert systems in design domains were developed. This problem domain was investigated actively in the 1970s and 1980s. A generic model of configuration tasks was addressed in [5]. The tasks are viewed in the context of AI.

2.3. Document Management

The key task within this view is managing and archiving design documentation in an enterprise. To provide solutions, Electronic Document Management Systems (EDMS) are developed. The problem domain is active since the 1990s.

Document creation is in the foreground of an EDMS document lifecycle. An EDMS document lifecycle considered in [18] is depicted in Fig. 1. A simplified EDMS document lifecycle comprises three principal phases Create-Manage-Distribute as illustrated.

in Fig. 2. The documents are synthesized in the first phase.

As one can see from Fig. 1, the document management view to the document creation is actually a business perspective. Bielawski & Boyle [18] advocate a user-centered approach to document management: “The value of a document is realized when it is consumed or used, not when it is created.” In the user-centered paradigm the focus is on the consumer of a document, not creator. This is illustrated in Fig. 3. Thus the user-centered approach is consistent with a modern business view which focuses on the consumer, not the producer.

3. Document Preparation

We treat structured document as an object formed as a hierarchy of more primitive objects and thus use an object model of documents presented in [7]. Each object is an instance of a class that defines possible constituents and representations of instances. Objects further are classified as either abstract or concrete. One or more concrete objects correspond to an abstract object. The term logical object is used as an informal synonym for the abstract object. Concrete objects are defined over one or more two-dimensional page spaces and represent possible formatted images of a logical object. We use the term graphical object (element) for concrete objects. Graphical objects form the layout structure of the document.

Typical document classes include books and articles. Common lower-level classes include such document components as sections, paragraphs, headings and figures.

The interests in the area of structured documents have led to the creation of several
standards. The best known are Open Document Architecture [4] and Standard Generalised Markup Language [2]. Tools for processing structured documents are addressed, e.g. in [12, 14]. They process the structure of textual documents. The concepts of graphical documents are not established yet. Therefore graphical tools poorly support the structure of graphical documents.

The core idea of SyntheCAD is to derive the structure of graphical documents from the model of artefacts that are constructed in the application domain.

Document preparation involves two tasks:
- editing — defining the content and structure of a document, and
- formatting — generating the document from specification of its appearance.

The paper aims at the characterisation of both aspects of graphical document processing in SyntheCAD. In the editing we use a method of generic coding. In the formatting we use a component-oriented configuration method [20].

Different categories of configuration knowledge have to be represented in the formatting. Two categories of configuration knowledge are considered in [21]:
1. Problem-specific knowledge (input knowledge: the logical structure of drawings);
2. Persistent knowledge (including domain-specific method independent and method-specific domain knowledge).

In the synthesis of technical drawings we encounter the third category — visualisation requirements. We consider the technical drawings as graphical documents that possess a structure which can be visualised on paper or screen.

4. SyntheCAD: First a Software System, then a Method

The paper is aimed at presenting SyntheCAD as a method. Historically, first the software system was implemented, then the method generalised.

The SyntheCAD approach to synthesis of technical documentation is based on:
2. Integration of various representations for different knowledge categories:
   - arrangement language in order to express the hierarchy of key components and their attributes. The arrangement language serves for a source (input) document;
   - a high-level language, GCL, for representation of configuration knowledge. The knowledge representation is as in the Generative Constraint Satisfaction Problem [20]. The introduced components are subject to a set of constraints depending on the component type.
3. A general purpose CAD tool that serves to handle static layouts and visualise the drawings.

In examining the production of the technical drawings we try to answer the following questions:
1. Can graphical document processing be based on structured textual document processing methods? What do the methods of textual and graphical document preparation have in common?
2. What is the logical structure of the technical drawings? How can this structure be represented?
3. What kind of knowledge do we have?

In order to answer the questions above, we investigate the following subjects:
- Characterisation of document structures by using of different specification methods including DSSSL, ODA, SGML and XML.
- The structure of application domain viewed as the logical structure of the technical documents.
- Knowledge categories in graphical document preparation tasks and knowledge representation methods.

In order to prepare a technical drawing of an artefact, the artefact has to be assembled from simpler components. Such assembling is addressed in AI area configuration. The configuration task is synthetic, i.e. the configuration system is expected to produce a new artefact.

5. Overview of Configuration, Silicon Compilation and Document Processing

Configuration, silicon compilation and document processing are utilised essentially in SyntheCAD. Therefore we present a short introduction to these subjects.

5.1. Configuration

The current mass production process does not turn out the huge numbers of identical and uniform products. Current industrial production process has to provide the ability to adapt individual delivered products according to the wishes of the customer. Even if we have fixed a set of parts, varying their choice allows the assembly to become more flexible and adaptable to individual order.

This is directly translated into the advantage in sales and marketing. But as a consequence the configuration of artefacts can become a highly complex and time-consuming task.

Variations can be of various kinds of [24]:
- A set of components. The component set can be
  - determined in advance. In this case the freedom may exist in specifying the properties of components, such us variable number of places to which other components
can be connected (ports);
- parameterised;
- given as component types.

Assembly. The arrangement may be
- fixed. In this case the configuration task is to assign values to component parameters;
- less restricted. In this case we have only partial arrangement called a skeleton arrangement. The goal of configuration task in this case is to determine the specific arrangement;
- not limited by the specification of the problem.

Relations. The relations between the components and requirements may be specified as either initial value assignment to component parameters or determined as a required function.

The combinations of possible variants divide configuration tasks into eight types:
1. Local verification
2. Incremental verification
3. Functional verification
4. Assignment
5. Layout design
6. Parametric design
7. Skeletal design
8. Full configuration design.

The goal of automated configuration is the composition of complex technical systems from a catalogue of parts. No new components can be designed and a set of requirements is complete.

Another classification of configuration tasks is given in [20] and is shown in Fig. 4. Here the approaches to configuration are divided into two major subgroups: representation-oriented approaches and task-oriented ones. The central issue in the representation-oriented approaches is finding the correct representation for expressing the structure and properties of the problem domain.

According to the task-oriented views, identifying the subproblems to be solved is the most important issue. The task-oriented approaches are based on assumptions that description of tasks the system has to perform is more important for building the knowledge base than representation formalism.

In accordance with this classification, SyntheCAD is within component-oriented configuration approaches.

5.1.1. Representation Oriented Approaches to Configuration
Knowledge representation in the first-generation expert systems was based on production rules in the form IF condition THEN action [10]. When knowledge is expressed as
rules, new facts can be obtained by deduction. Using forward chaining, we start from conditions, which we know to be true towards the conclusions we want to establish. In backward chaining, we move from the conclusions we want to be true towards the conditions of the rules, which must be true in order to establish the conclusions. The problem in this approach is ensuring the correctness of a rule-based system.

In structure-based configuration, the key component approach [5] was defined. In this approach components are complex objects which can be connected in different ways. The key components are those components that are central to the architecture of the system, and their choice is a basic decision in modelling a particular artefact.

A constraint satisfaction problem (CSP) is considered to be a formalism for describing application-specific configuration knowledge. This formalism is used in expressing expert knowledge, that have been described as a result of actual experience. CSPs can be represented in a declarative way by (1) identifying the variables of interest of the problem, (2) specifying the values for the variables which form the domains, and (3) restricting combinations of value assignments with constraints. Solving constraint problems involves finding an assignment to the problem variables subject to constraints.

Dynamic CSP extends the restriction of CSP that postulates the existence of a fixed, enumerated set of variables. The main goal of a configurator in this case is to select a correct set of components, based on the functionality they provide. The number of components in this case is not predefined.

Component-oriented configuration attempts to combine the advantages of structure-oriented and resource-oriented configuration. Instead of a fixed kit of individual com-
components, we have the set of component types, with each type determining the structure and constraints of its instances.

During the component-oriented configuration, three basic actions are performed: (1) creating components, (2) choosing type for a component, and (3) connecting two components. In SyntheCAD, according to the component-oriented approach, expert user knowledge is represented.

5.1.2. Investigation of Knowledge Categories Involved in the Layout Configuration Task

Computer support for certain engineering tasks tends to be integrated into one system where all modules are coherent. A system has to provide the communication between the modules and to keep data and knowledge available for the system by supporting the process of knowledge structuring. The maze model of knowledge structuring is investigated in [22].

The knowledge is taken directly from the technical specification of the components (processing units) and is also derived from long term experience with actual configurations.

In synthesis of technical drawings three categories of knowledge are considered:

• the logical structure of the drawings,
• formatting knowledge of expert users,
• visualisation requirements.

5.2. Silicon Compilation as Knowledge Representation Language

The SyntheCAD formatter language GCL (Graphical Circuit Language) is designed and implemented according to silicon compilation principles. More precisely, GCL was developed according to JCL [1].

Silicon compilers [3] are defined as programs that generate layout data from some higher level description: a set of boolean expressions, a behavioural description of a microarchitecture, an instruction set, etc. Silicon compilers contain knowledge of how to synthesize constructs from basic components and how to arrange the components. There are two focal points of concern in a silicon compiler:

• The source language: a language in which the user specifies desired function, or behaviour, to be performed by a new integrated circuit chip.
• The target language: the capabilities of silicon, utilised in a very complex two-dimensional colour picture called layout.

Silicon compilation improves design quality through “correct by construction” capability. The design errors introduced by designer on more abstract levels are easier to detect and correct. Instead of dealing with millions of polygons or transistors, the designer must deal with only two dozen microarchitectural blocks.

5.3. Document Processing

Processing of textual documents became the most popular computer application nowadays especially for the users which are not computer professionals. Typical text processing tools maintain the documents with established structure. The structure is represented using codes that describe the formatting of parts. However the structure is not defined explicitly. The author determines the document form in the input stage. He decides how elements of the structure will be represented. For example, section names are centered, the first line of a paragraph is intended. In another context the same text might be presented in a different way: section names and paragraphs start from the left margin, separated by empty line. This is a procedural approach to text processing.

Cognitive science states that users have a declarative view of the document. The user defines abstraction of the document at higher level than a physical representation level. He determines in advance the elements of a document, their hierarchy and representation. If the same document in different contexts has different images, the user needs only the change of representation rules of elements, without changing of the document instance.

5.3.1. Preparation of Structured Documents

The declarative approach presupposes that the document structure and formatting rules are represented in the system. Standard Generalised Markup Language (SGML) [2, 17, 15] is an international standard that defines the logical structure of a document. HyTime [8] is an extension to the SGML standard and an application of it. HyTime defines models for structures of hypermedia documents. HyTime was developed as a result of the work on developing Standard Music Description Language (SMDL). Document Style Semantics and Specification Language (DSSSL) [13, 17] complements SGML and standardises formatting rules. ODA (Open Document Architecture) [4, 17] is an international standard that defines both the structure and representation of a document.

The structure of logical objects in the processing of structured textual documents is investigated in [11, 12, 14]. The addressed methods do not suit for graphical structured documents. We use the terms of graphical documents and technical drawings as synonyms. Graphical tools heavily support the structure of documents because the concepts are not established yet. Such documents are hardly to modify. We try to solve these shortcomings in SyntheCAD.

5.3.2. Generic Coding

The concept of generic coding gained prominence in the 1970s and was implemented in Standard Generalised Markup Language [2]. Electronic manuscripts earlier contained control codes that caused the document to be formatted in a particular way. That was `specific coding'. In contrast, generic coding which began in the 1960s, uses descriptive tags, e.g. `section' rather than `14 point Palatino'. Central to the concept of generic coding is the separation of the information content of documents from the formal or
the appearance of the content. Information content has a structure. Therefore we talk about logical structure of the document [17]. The structure of the appearance of a document we call layout structure. The layout and the layout structures of documents are examined in [6]. These concepts make the background for Document Style Semantics and Specification Language. The aim of separating the logical and layout structures is to achieve flexibility in the preparation of structured documents. We understand the flexibility as multiple specifications that may be applied to a given document in order to yield various presentations of the same data.

In SyntheCAD we treat the document preparation as a configuration task. The logical and layout structures are separated and processed differently.

6. The Conceptual Model of SyntheCAD

Interactive work with a CAD tool is the most usual way to create a drawing in a design office. Quite often a match case is chosen among previously solved cases. Then modifications are performed interactively.

SyntheCAD assumes completely automatic configuration. A technical drawing is configured from a fixed given set of component types. A component stands for a processing unit with a set of attributes which specify the behaviour of parts to be assembled.

A new instance of an electroplating line is created starting from an individual order which specifies customer requirements.

Document processing consists of executing operations in order to define, manipulate, and view abstract and concrete objects. For this purpose, we distinguish between ordered and unordered objects.

In electroplating lines we have ordered objects — processing units. One-dimensional order is defined by the user. The logical structure of an electroplating line is indeed a hierarchy of processing units and associated devices (subunits) attributed with their parameters.

The layout structure of an electroplating line considers that a drawing is divided into pages (of format from A0 to A4). Each page is partitioned into areas for:

- Global properties (attributes)
- Controllers with cables (in the case of electrical drawings)
- Transporters
- Associated devices
- Processing units which, in turn, consist of smaller components
  - static image
  - dynamic layout (for cabling and piping).

The conceptual model of SyntheCAD (see Fig. 5) has much in common with the conceptual model of DSSSL [13]. Two processes are distinguished:
• the transformation process, and
• the formatting process.

For controlling of each processes, an appropriate SyntheCAD language was developed. The transformation language controls the transformation process. Likewise, the style language controls aspects of formatting process.

A SyntheCAD document is specified by arrangement language (see Section 6.1). The logical tree — the model of an artefact, an electroplating line — is formed by the logical parser-tree, which is decorated with synthesized attributes. The parser-tree nodes represent the components. The synthesized attributes represent the properties of the components.

6.1. The Arrangement Language

In SyntheCAD a context-free declarative language was developed for describing of both a SyntheCAD document and of its Document Type Definition (DTD) [15].

A SyntheCAD document specifies the logical structure of technical drawings. Comparing with classical attribute grammars, our language allows to describe the attributed tree shorter and without superfluous redundancy. This is due to the problem domain specific.

The Arrangement Template plays the role of a DTD. A SyntheCAD document with a concrete logical structure serves as the arrangement of electroplating line’s instance. The above mentioned context free language is called the Arrangement Language. The syntax of electroplating line DTD can be shown as grammar below:
SyntheCAD document → head component ( attributes* ) [ component* ]
component → component_id ( attributes* ) [ component* ]
attributes → positional attribute* keyword attribute*
positional attribute → type_id attribute_id
keyword attribute → attribute_id < value set > = default value
attribute_id → string
component_id → value, * value
type_id → int | real | bool | string
default value → int | real | bool | string

6.1.1. DTD — the Arrangement Template

The Arrangement Template represents syntax for the whole class of arrangements. The arrangement template describes artefact’s model including node names, the “part of” hierarchy of nodes, and field names. The arrangement in Section 6.1.2 corresponds to the following Arrangement Template:

PLATING LINE ( string type, /* ID of line */
    width <1120, 1500 >, /* Width: 1120 mm or 1500 mm */
    transporters <1, 2, 3, 4> = 2, /* Number of transporters */
    direction <direct, reverse> = direct /* Direction: */
) /* right to left or left to right */
    [ BATH ( string type, int position, drum <yes, no> )
    [ LEVEL SENSOR
    STEAM VALVE
    RESERVOIR
    RECTIFIER ( string type )
    [ ]
    DRYER ( string type, int position )
    LOADER ( string type, int position )
    STORE ( string type, int position )
    [ ]

6.1.2. An Example of a SyntheCAD Document

A SyntheCAD document represents a domain-specific logical structure, termed arrangement [19]. The arrangement establishes specific relationships between the components. The relationships will precisely locate one component with respect to another or with respect to some reference location.

The main information contained in the arrangement is a sequence of baths. Thus the problem domain complies with the general definition of a configuration task as well as two assumptions identified in [5], “functional architecture” and “key component per
Various Perspectives of Automatic Configuration of Structured Graphical Documents

Function.

Arrangement Language terms originated during analysis of the problem domain. The purpose and properties of bath types and other components was analysed. An example below illustrates a sample customer's order, i.e., an arrangement example:

```
PLATING_LINE ( 443214.001E4, 1120, transporters = 1, direction = reverse )
[ /* Type, position, drum */
  LOADER ( 009 , 1 ) /* Load and unload */
  STORE ( 057 , 2 ) /* Storing to wait */
  DRYER ( 001 , 3 ) /* Drying in hot air */
  BATH ( 002 , 4 , yes ) /* Rinsing in hot water*/
  [ LEVEL_SENSOR STEAM VALVE ]
  BATH ( 001 , 5 , yes ) /* Rinsing in cold water*/
  [ LEVEL_SENSOR ]
  BATH ( 006-01 , 6 , yes ) /* Chemical treatment */
  BATH ( 001 , 7 , yes ) /* Rinsing in cold water*/
  [ LEVEL_SENSOR ]
  BATH ( 006-01 , 8 , yes ) /* Chemical treatment */
  BATH ( 001 , 9 , yes ) /* Rinsing in cold water*/
  [ LEVEL_SENSOR ]
  BATH ( 006-01 , 10 , yes ) /* Chemical treatment */
  BATH ( 009-03 , 11 , yes ) /* Electroplating */
  [ RECTIFIER("TBPI-1600/12") ]
  BATH ( 009-01 , 12 , yes ) /* Electroplating */
  [ RECTIFIER("TBPI-1600/12") ]
  BATH ( 001 , 13 , yes ) /* Rinsing in cold water*/
  [ LEVEL_SENSOR ]
  BATH ( 006-01 , 14 , yes ) /* Chemical treatment */
  BATH ( 001 , 15 , yes ) /* Rinsing in cold water*/
  [ LEVEL_SENSOR ]
  BATH ( 002 , 16 , yes ) /* Rinsing in hot water*/
  [ LEVEL_SENSOR STEAM VALVE ]
  BATH ( 008-01 , 17 , yes ) /* Corroding */
  [ RECTIFIER("TBPI-800/12") LEVEL_SENSOR STEAM VALVE RESERVOIR ]
  BATH ( 008 , 18 , yes ) /* Corroding */
  [ RECTIFIER("TBPI-1600/12") LEVEL_SENSOR STEAM VALVE RESERVOIR ]
```

6.1.3. The Transformation Process

During the transformation process, a SyntheCAD document is transformed into a set of nodes with attributes (a logical tree). The obtained logical tree is used further as input to the formatting process.

All operations performed in the transformation process are independent of the formatting process. During the transformation process, the syntax analysis of a SyntheCAD document is accomplished and the following subjects are created:

- nodes and their attributes in accordance with user specification,
- relations between the nodes.

![Diagram of Transformation Process]

Fig. 6. The conceptual model of transformation process.

6.2. The Style Language

The style language specifies the formatting process (see Fig. 7). The formatting process applies presentation styles to source document content. The style language defines the visual appearance of a formatted document in terms of formatting characteristics attached to an intermediate tree.

The formatting process uses style-specification which may include construction rules, page-model definitions and other application-defined declarations and definitions.

6.2.1. The Formatting Process

The formatting process is comprised of the following subprocesses:

- Application of formatting rules to the objects in the logical tree.
- Definition of page geometry.

- Layout of the document content. Each object is formatted, its layout is produced and positioned by a parent in the tree.

The following formatting actions are performed:
1. Enrich the logical tree with semantic information.
2. Apply construction rules to the component nodes.
3. Calculate the layout structure.
4. Perform semantic analysis of layout structure including pagenuumbering, piping and cabling.
5. Prepare output code for a user-defined CAD system.

Fig. 7. The conceptual model of formatting.

6.2.2. The GCL Language

High level declarative language GCL (Graphical Circuit Language) was developed in frame of SyntheCAD for specifying and creating layouts. According to principles of silicon compilation, GCL serves for knowledge representation and 2D graphics. It has a syntax and 2D graphics features of JCL, i.e. a silicon compilation language [1]. A function written in GCL is associated with each component type.

One way to create the layout is to specify it in a language which admits convenient specification of geometric items and their combinations. All suitable languages for layout specification have common notations for specifying two-dimensional points.

Each component is defined by a function in GCL:

```
DEFINE name ( arguments ) : type;
BEGIN
 VAR variables if needed
```

DO  actions if needed
GIVE value of the type described above
END
ENDDEFN

The component’s layout is defined with the function notation DEFINE ... END-DEFN, where the output is of the recursive type LAYOUT. Since the layout is a picture, we can conceive that it is made up of a large set of individual polygons. The polygons are organised as a recursive data structure — a tree.

6.3. Plotting a Layout

The plotting process interfaces between a layout and a specific plotter. The layout is of the recursive type LAYOUT, the plotting program is recursive, too. Now, we can describe the PLOT program:

DEFINE PLOT ( L: LAYOUT, DISP: POINT );
BEGIN VAR L1 : LAYOUT;
CASE L OF
    POINT:    plot the point displaced by DISP
    BOX:      plot the box displaced by DISP
    CIRCLE:   plot the circle displaced by DISP
    TEXT:     plot the text displaced by DISP
    POLYGON:  plot the polygon displaced by DISP
    UNION:    FOR L1 ∈ L; DO
        PLOT ( L1, DISP, COLOR )
    END;
    MOVE:     PLOT ( L.DISPLACE, DISP+L.BY );
ENDCASE
END
ENDDEFN

This recursive program is easy to understand. For a complex layout, the function calls itself recursively until a graphical primitive including a point, line, box, circle, text and a polygon. In case the user requires a new CAD tool, the PLOT program has to be modified.

7. The Problem Domain of Electroplating Lines

This section is devoted to short description of the problem domain in which SyntheCAD has been used.

An electroplating line is a conveyor-based array of processing units as depicted in Fig. 8. One or two transporters move above the electroplating line. The transporters
carry workpieces and put/get them into/from the processing units. Thus the processing units are loaded/unloaded with the workpieces for a certain time period.

The processing units and transporters are monitored by microprocessor controllers. For example, the controller turns the bath’s heater on when the temperature of the solution in the bath becomes lower than a certain boundary value.

An electroplating line has global attributes including type, width and the number of transporters. It is configured from a fixed given set of typical components. Several components can be of the same type. The component types in include processing units (baths, loaders, stores and dryers), transformers, pumps and other devices.

The bath is the major processing unit type and has the following attributes:

- function (electromechanical, rinsing, etc.),
- characteristics (dimensions, material, etc.),
- optionally attached equipment (temperature sensor, filtration unit, agitation device, pump, etc.),
- associated devices (subunits) (rectifier, reservoir, safety level sensors, heater, cooler, steam/water electrovalves, etc.).

The knowledge associated with every processing unit is of three kinds as shown in Fig. 9. The technical specification to build an electroplating line specifies the types of all processing units. The subunits are represented as child-nodes to baths. A component is shown on the drawings as a two-dimensional image.

The problem domain is the subject to changes. New equipment and new modifications of processing units are developed. The graphical representation of existing equipment can also be changed. These changes cause the change of connection algorithms including cabling and piping. The most frequent activities during the design of technical drawings are the following:

- introducing new modifications of units,
• modifying the images of units,
• changing connection algorithms.

The natural decomposition is observed in the domain of electroplating lines. The structure of technical documentation is shown in Fig. 10. The set of documents is partitioned into mechanical drawings and electrical ones. The electrical drawings consist of the diagram of connections and the device layout. The mechanical drawings consist of assembly drawings, the piping plan, the framework and the fume ventilation plan. Each set usually takes several pages.

The components are shown in an appropriate diagram. For example, electrical connections of baths are shown in the diagram of connections. Piping of the baths is shown in the piping diagram.

8. Data Flow in SyntheCAD

The data flow in SyntheCAD is shown in Fig. 11. The logical structure (arrangement) of an electroplating line is given by the user and serves as an input. The script files of drawings for a general-purpose CAD system serve as output. The configurator’s knowledge base and the library of graphical symbols are prepared in advance.

Fig. 11. Data flow in SyntheCAD: from the customer’s order to drawings.
8.1. SyntheCAD as a Component Oriented Configuration Method

Section 5.1.1 introduced three basic stages, which are performed in an component-oriented configuration approach:

1. Creating components
2. Choosing type for a component
3. Connecting two components.

The logical structure (arrangement) in SyntheCAD defines what components are created. This is in the first stage in the component-oriented configuration process. Component types are specified by the user.

The configuration's second and third stages are performed during the interpretation of the logical tree. The tree is traversed. Semantical actions include the forming of component layouts, connecting of the components, cabling and dividing of the drawings into pages.

During the interpretation, the layout structure is derived from the logical structure. The actions are specified in GCL programs.

The architecture of SyntheCAD consists of arrangement input, GCL interpreter, GCL compiler, linker, and the PLOT program (see Fig. 12). During the arrangement input, the components are created. The logical parser-tree of an electroplating line results. The interpreter traverses the tree and adds new variables (components and attributes) to the configuration, enriches the logical tree with semantic information, calculates the layout structure and accomplishes semantic analysis (divides into pages, connects the components and provides the cabling). The layout structure of the drawings is yielded during the interpretation step.

9. Drawings Synthesis Perspective to SyntheCAD

In organisational context SyntheCAD assumes completely automatic configuration. This mode reduces the workload on experienced users. After synthesis with SyntheCAD, the changes to produced drawings can be performed interactively with a CAD tool.

SyntheCAD involves two sorts of users:

- **Expert users.** They provide configuration knowledge. They are interviewed for the purpose of knowledge acquisition.
- **Noneexpert users.** They prepare source documents and produce drawings.

The expert users perform the following tasks:

1. Analyse an application domain and provide the structure of its concepts. In accordance with this structure, the user creates a DTD for a designed document class.
2. Define a function (in GCL) for each element of DTD. The DTD definition serves for configuring and formatting.
3. Fill in the fields of graphical primitives in the PLOT program in terms of a chosen CAD system.

SyntheCAD has been used to process several document types. Several CAD tools running on different platforms have been used for visualisation including AutoCAD for PCs. The practice shows that the design of a DTD and a knowledge base eliminates the redesign of technical drawings.

10. Conclusions

The SyntheCAD method shows that the logical structure of graphical documents can be formalised in terms of an application domain and used further in the configuration.

The usage of the SyntheCAD system at several design offices proved the method in
real life. SyntheCAD as a document preparation system is designed for specifications which are applicable to 2D graphical documents that have an internal aggregation structure.

References

1983

1986

1988

1989


1990

1992


1993

1994


1996


