ELSEVIER

Automation in Construction 10 (2000) 181–197

AUTOMATION IN CONSTRUCTION

www.elsevier.com/locate/autcon

Structural design in an automated building system

Rafael Sacks [a,*], Abraham Warszawski [b], Uri Kirsch [c]

[a] *Department of Construction Management, Faculty of Civil Engineering, Technion, Israel Institute of Technology, Technion City, Haifa 32000, Israel*

[b] *Faculty of Civil Engineering, Technion, Israel Institute of Technology and National Building Research Institute, Technion City, Haifa 32000, Israel*

[c] *Faculty of Civil Engineering, Technion, Israel Institute of Technology, Technion City, Haifa 32000, Israel*

Accepted 15 March 2000

Abstract

Structural design of buildings has proven particularly difficult to automate. Parametric templates are too limited to be practicable, and pure AI-based approaches have found little application in design offices. This paper presents 'Intelligent Parametric Templates' (IPT) for structural design within an Automated Building System. It demonstrates that, for rectangular plan building types, comprehensive automation of general and detailed structural design is feasible. The software 'knowledge modules' that were developed deal with rectangular buildings. IPTs for two complete slab solutions have been implemented. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Structural design; Computer integrated construction; Building project model; Automation; Information technology

1. Introduction

While computers are currently used to perform diverse design and management tasks in the construction industry, the full impact of Information Technology on design automation and process integration has yet to be felt [1]. It is expected that fully computer integrated construction (CIC) will fundamentally change the ways in which project participants perform and communicate [2,3]. The concept of an Automated, Computer Integrated Building Realization System (ABS) was set out previously in this journal [4]: its aim is to automate, as far as possible, the generation of technologically feasible design and planning alternatives for selection by the operators [5]. The system stores the data describing the project using an object-oriented Building Project Data Model (BPDM) [6]. The specific goal of the research presented here was to propose, develop and demonstrate ways of automating structural design for rectangular buildings, whose shape does not change through their height, within such an integrated system. The principal difficulty is that of generating design schema. This is the most complex part of the design process to computerize, as it is 'creative' in nature.

The challenge is to address numerous and diverse design situations effectively, without requiring exten-

* Corresponding author.

PII: S0926-5805(00)00074-1

sive libraries of preset design solutions or of extensive domain knowledge. Parametric templates or frames are too inflexible [3]. Various Artificially Intelligent (AI) approaches to structural design and floor layout have been researched: Algorithmic, Knowledge-Based Expert Systems (KBES) [7], Case-based Reasoning (CBR) [8], Genetic Algorithms [9], Fuzzy Logic [10], Neural networks, and hybrids of these [11].

The 'Multilevel Selection and Development' (MSD) method [12] proposes an algorithmic method, in which design is considered as a series of steps of Selection (of a possible solution) and Development (more detailed design of the assembly or part). At each step, development of a selected solution introduces the need for selection of component parts for that solution.

Numerous KBES efforts have been made (e.g. HI-RISE [13], KTISMA [14], EIDOCC [15]). Synthesis of structural schemes for prefabricated slabs has been demonstrated [16]; a similar approach is used by the KTISMA system [14] to propose placement of beams and slabs for the ceilings of apartments. In both of these systems, the structural design is assumed to be secondary to the architectural design of the spaces; the possibility that a change in the architectural partitioning might allow a superior structural solution is not considered. In the HI-RISE expert system [13], structural schemes for high rise buildings are formulated as being composed of distinct sub-systems. Each sub-system is created by using 'templates' of predefined solutions, such as frames, grids, trusses, cores, etc., which are adapted for use by setting values for their pre-defined parameters.

The IBDE system [17] used a number of expert system type modules sequentially in order to perform building design and construction planning. Its basic aim was to investigate a scenario in which computer technology facilitates integration of a building team so as to approach the cohesion that characterized the 'master builder' construction paradigm of the past. Its function implied a high degree of automation, but it lacked flexibility and cohesiveness owing in part to the absence of a comprehensive Project Data Model. This was a 'Tool-centered approach,' in which the process was essentially defined by the available knowledge-based program tools.

Systems such as ARCHIE-II [18] use Case-Based Reasoning (CBR) techniques to help produce new solutions on the basis of past cases. Oxman [19] details three possible steps for using a case retrieved from a data base of cases in a specific design situation: parametric adaptation, substitution of a part, and topological adaptation. These techniques have not commonly been employed in structural engineering design applications. The CADRE system [20] is one example; it uses CBR to perform spatial design of large warehouses or factory sheds. The aim of the SEED system [17] is to enable rapid development of design alternatives at the early stages of building design. It is intended to be operated interactively by the architects and engineers involved in the design. SEED proposes incorporation of CBR capabilities, and contains mechanisms to enable storage and retrieval of design cases. This approach is useful here since the *operator* navigates the selection and use of the cases or parts of cases — this is not done by rules or by any automatic program.

FLEXPERT [10] performed facility layout based on the theory of fuzzy logic. Genetic algorithms have also been used in prototype layout design systems [21], and Park and Grierson [9] applied a Multicriteria Genetic Algorithm to lay out the column bays and floor heights of rectangular buildings. This system set values for predefined parameters to layout the structure — the number of bays and the bay sizes in each orthogonal direction, the number of floors and the floor heights. Its fixed set of parameters does not allow for vertical openings, building cores, irregularly placed columns, and it does not consider the effect of lateral loads.

These approaches have as yet had limited application in engineering practice [11]. They require large investments to build, are difficult to maintain [22], have tackled isolated problems within the spectrum of the design and construction process [16,23], and many were built without consideration for the requirements of data integration. Therefore, a new approach to automating structural design for the type of buildings considered here is needed. Such an approach can, and should, make use of the advantages presented by product modeling technology and the underlying object-oriented paradigm, which has been adopted for construction information integration [1].

The proposed approach, using Intelligent Parametric Templates (IPTs), and rooted in the object-oriented Building Project Data Model of the ABS, represents a step in this direction. At the outset, the BPDM was refined so as to more comprehensively reflect the information needs of the structural design process. The structural knowledge modules were developed using the IPT approach, and then tested in design of rectangular buildings. Due to the limitations on the scope of the research, emphasis was placed on attempting to cover the full design process, while restricting the 'breadth' of the possible design alternatives at each stage. Two complete technical solutions for structural slabs (Reinforced Concrete Flat Slabs, and RC Ribbed (Joist) Slabs) were implemented.

2. An automated building system (ABS) and its building project data model (BPDM)

On the basis of the ABS concept outlined in Ref. [5], a system intended for the design of multi-story buildings with rectangular, uniform floor plans has been proposed [4]. The ABS functions in a step-by-step manner through distinct predefined stages; it uses knowledge-based software modules ('Knowledge Modules', or 'KMs') to generate and manipulate the project information, and has on-line access to various construction databases. Each of the knowledge modules incorporates procedural knowledge of its domain, such as structural design, floor layout, electrical system design, HVAC, etc. The knowledge modules add new data to the building project model at each step. The system can produce, on demand, any of the project documents required by the user. The user can respond and intervene at each stage to approve or reject the results presented. The user can also change the project data directly, and, if necessary, instruct the system to redo any previous step.

The Building Project Data Model (BPDM) defines how the data that describe a building project will be stored and used by the system. The majority of previous research in the fields of Building Product and Process Models has focused on their use as tools for integrating the work of the individual members of a project team [2,24]. Sharing building product information is the primary goal of the building product model being developed in conformance with the STEP — ISO 10303 [25] protocol. More recently, the 'Industry Alliance for Interoperability' consortium has led a modeling effort intended to produce a model definition in the form of a set of 'Industry Foundation Classes' (IFCs) [26]. Various facets of the ABS placed specific data storage demands on the BPDM [27]. For this reason, data storage for the ABS is performed using a custom designed and built Building Project Data Model (BPDM). The BPDM is briefly described here, and in more detail in Ref. [6].

The model supports storage of the information through the full project life-cycle. As such, the model schema must account not only for the varied data requirements of different system modules, but also for the different ways in which they require that the data be structured. This is a daunting challenge for the case of a generic building model [24]; the BPDM presented here is tailored specifically for rectangular shaped, multistory buildings whose floors do not change shape through the height of the building.

The central schema of the model has three main axes, which define the spatial, physical and construction activity aspects of the building (Fig. 1). Each axis is decomposed into three main levels, which describe the whole, the assemblies and the parts of each aspect. The first main axis is that of the spaces: this includes the 'Building', the 'Primary_Space' (a floor of the building) and the 'Secondary_Space' (a room or other designated area of a floor). Physically, the building is composed of 'Building_Assemblies', which are subdivided at the floor level into 'Work_Assemblies'. These are defined by the technology (type of work and materials) with which their parts are installed; for example, all the interior partition walls of drywall construction on a particular floor are grouped in a *work assembly*. The individual parts of a *work assembly* are called 'Elements'; each partition in the previous example is an *element*. The classes of the third axis define the way in which construction process information will be stored. The 'Task' represents all of the work to be done by a particular organizational unit (a 'Team'), usually of a particular type, e.g. 'flooring'. Each *task* contains 'Activities', which are defined as the work required to install one *work assembly* in one *primary space*. 'Basic_Activities' install elements and also serve to

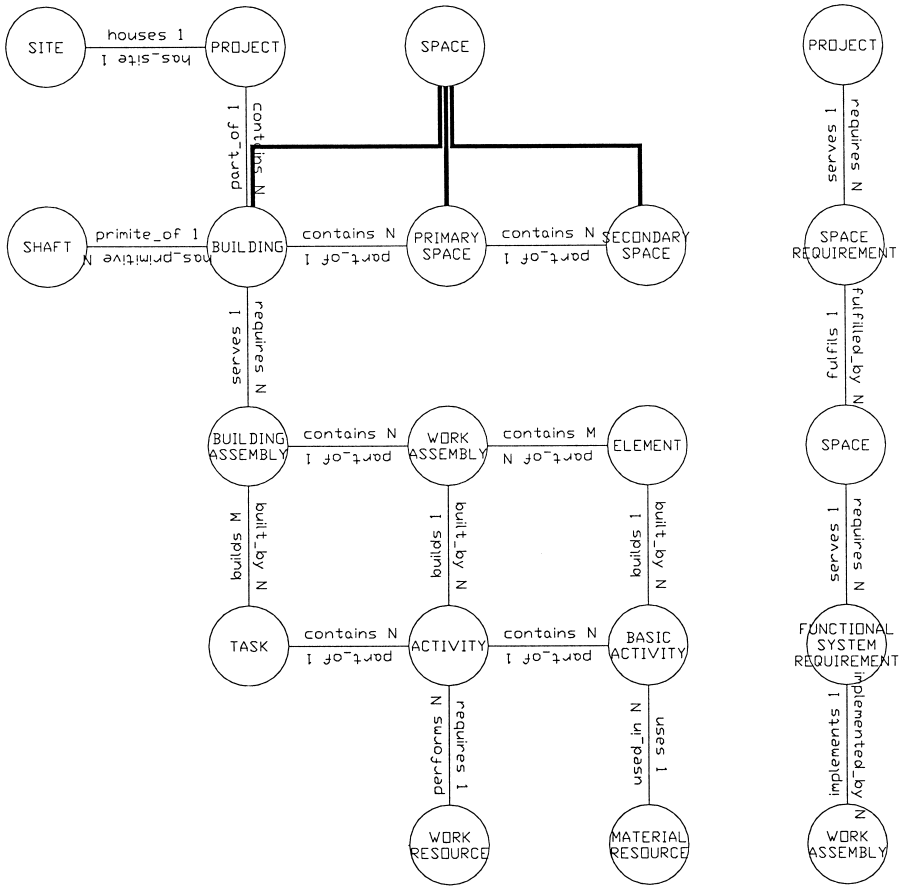

Fig. 1. The central part of the Building Project Data Model schema.

link the material resources (components or bulk materials) to the *work assemblies* and *elements*.

3. The intelligent parametric template technique

The challenge of automating synthesis of structural layouts for the ABS required development of a new approach. While the use of simple geometric templates is common in most CAD systems, building systems require complex templates: static definition of parameters restricts a template to only the simplest of real design situations. In addition, automation requires that the knowledge relating to any building assembly or element be available: in most CAD systems no knowledge is stored, and in Knowledge-Based Expert Systems the knowledge is stored in a knowledge base. In the Intelligent Parametric Template (IPT) paradigm developed here, three information technologies — geometric templates, production rule processing and object-oriented design — are used in concert. Each IPT is based on a dynamic geometric template, which incorporates distinct data objects. Each object includes production rule sets and object methods, which enable it to calculate the values of its parameters and to display behavior.

For the sake of explanation, consider how a parametric template for a reinforced concrete slab, supported on orthogonal RC beams, with openings for elevator and stairwells, could be defined. To start with, a simple box (which has three geometric parameters — width, depth and height) could be considered as a geometric template for the slab (Fig. 2a)

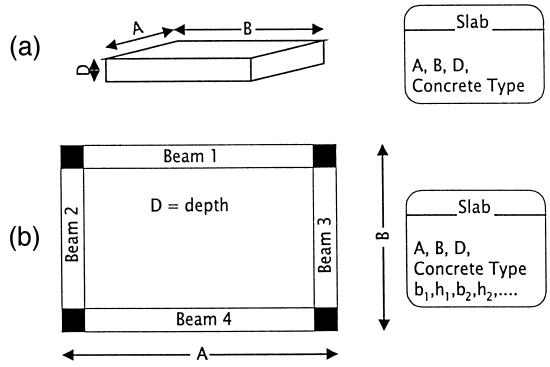

Fig. 2. Non-feasible approach to Template design.

(other spatial parameters, such as those defining location and orientation in space, are omitted here for the sake of clarity). The slab template also requires non-geometric parameters, such as material type, cost, etc. Rule sets, which define how the thickness of the slab should be calculated, and how the reinforcing should be apportioned, could also be added. Any particular instance of a simple slab, with only one bay, no holes and no beams, could be defined using this template. Now consider adding supporting beams along the sides of the slab, as shown in Fig. 2(b). One possibility would be to define a template object with 11 parameters: slab length A, width B, depth D, and the width and depth of each of the four beams b_1, h_1, b_2, h_2, etc. The logic of this approach implies that the number of templates, which would be required in order to satisfy all possible design situations, would be combinatorial. In addition, the numbers of parameters and rules grow as the complexity of the template increases. Any irregularity would also require addi-

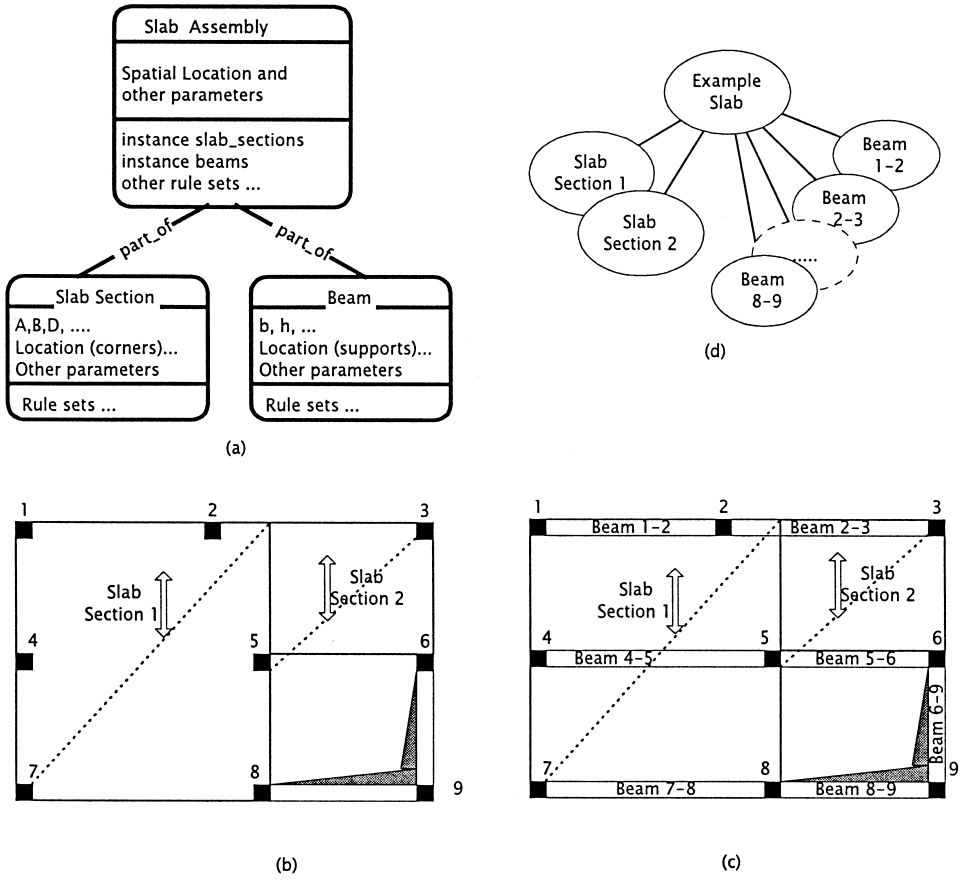

Fig. 3. Intelligent Parametric Template design.

tional parameters. This approach is not feasible, and is therefore rejected.

A technique is required whereby a template can adapt itself to all possible geometric and topological configurations that might be required. A good solution must allow for description of situations in which there are any number of bays in each direction, the bay sizes can differ, beams can be present or absent and have varying dimensions, the slab can have openings, etc., *using just one basic parametric template*. The solution must also account for the fact that the rules and algorithms for different work assembly technologies are significantly different; for example, the layout of a Reinforced Concrete Flat Slab is quite different from that of a Concrete and Steel Composite Deck.

The IPT approach provides these capabilities. The solution is to define an IPT not as one complex parametric object, but rather as **an intelligent mechanism for instantiation of a set of objects and of the relations between them**. The geometric base of an IPT is composed of many distinct data objects rather than being limited by a single predefined template. Its object-oriented design enables an IPT to dynamically instantiate its parts — the number and nature of parameters are set according to the geometry of each specific design situation.

A simplified IPT for a slab such as in Fig. 2 would comprise (as seen in Fig. 3a) the following.

(1) An object definition for a slab assembly as a whole.

(2) Object definitions for a slab section and a beam. The location and spatial orientation of each element is defined using co-ordinates and links to other elements to the slab assembly object; its shape and other local properties are defined by parameter values.

(3) Object methods (object-oriented software functions) for creating instances of the slab assembly, slab sections and beams, for linking the beams and slab Sections to the slab assembly with 'part of' relationships, and for computing or deducing the values of the parameters of each instance. The methods incorporate AI strategies such as rule-processing and inferencing, case-based reasoning, fuzzy logic, genetic algorithms, neural networks, etc. (in the current demonstration system, only rule processing has been used).

For example, consider how beams would be laid out for a one-way slab, such as that in Fig. 3(b). A possible algorithm, expressed in 'pseudo' code, might be:

```
Slab assembly :: instance beams
{ FOR EACH adjacent column pair
  { IF the axis direction is perpendicular to an
  adjacent slab section span direction
  OR IF there is no adjacent slab
  THEN { instance a beam }
  }
}
```

The resulting beams are shown in Fig. 3(c). The overall slab data generated are stored as a set of linked object instances (Fig. 3d), which together fully describe the slab and enable it to exhibit 'behavior' — for analysis and detailed design.

Full-scale IPTs use objects based on the assemblies and parts parent objects of all three aspects of the BPDM. Inclusion of 'Activity', 'Basic Activity' and 'Resource' objects enable an IPT to provide the construction process data as well as the physical design data.

In summary, an IPT can be defined as "a collection of object definitions, rules and methods, which fully describe a set of building spaces or elements and define the ways in which they can be inserted into a building model". The object class definitions include the attributes of each class. The rules and methods (the knowledge) of an IPT are stored as part of its object definitions. Also, each rule set in an IPT may be used by more than one knowledge module. An important feature of the IPTs and the object-oriented BPDM structure is that not only class properties and methods are inherited, but also IPT rule sets. This structure for storing the knowledge greatly facilitates the addition of any new technical solution: this is done simply by defining it as an IPT, defining its class inheritance, and adding only those specific properties, methods and rules that distinguish it from the (previously defined) solution(s) from which it inherits. Further detail can be found in Refs. [6,28].

4. General and detailed structural design — an overview

Engineering design is commonly divided into four main phases [29]: planning and clarifying the task (specification of information), conceptual design (specification of principle), embodiment design (specification of layout) and detail design (specification of production). This breakdown is apparent in the design phases defined for the ABS which include: Brief Generation, Conceptual, General and Detailed design [4]. In the first phase, the client's requirements and the site data are acquired and clarified. Next, the building's shape, height, position and the number of floors are proposed. General design covers the selection and layout of all of the building's spaces, assemblies and their components, and construction activities. The dimensions and details of the objects are calculated and added in the Detailed design phase.

The discussion that follows focuses only on the structural aspects of the General and Detailed design phases for rectangular buildings intended for office, commercial, or light industrial use. In such buildings, partitioning of each floor is usually defined after general design is completed; changes in the partitioning of the floors may also be common through the building's life.

In General Design, the spacing of the vertical structural elements (columns and/or walls) must facilitate partitioning suitable for the function of the building's spaces; the placement of the building's core(s) must also satisfy both architectural (transport and fire safety) and structural requirements. The sub-division of the core spaces must fulfill space requirements (e.g. storage, restrooms, etc.). The strategy adopted here is to divide general structural design into: layout of the structural scheme, positioning of the core(s), layout of the core spaces and core walls, and finally fixing positions for columns. This is shown in an SADT diagram in Fig. 4 and in simplified form in Fig. 5.

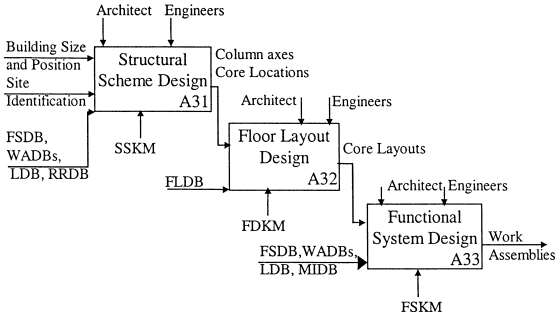

Fig. 4. The sub-stages of General Design. (Abbreviations defined in text.)

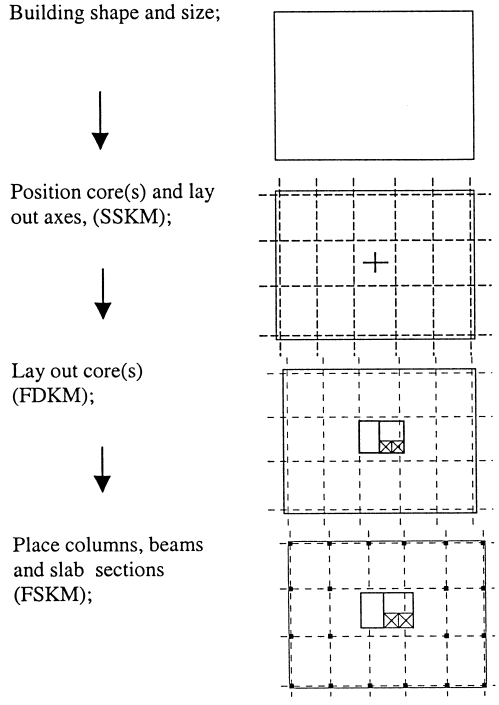

Fig. 5. The sequence of General Design.

In the first sub-phase of general design, the *Structural Scheme Knowledge Module* (SSKM) receives the perimeter of the building and its position on the site, the number of floors, and the space requirements for the building's core spaces as input. It calculates optimal column spacings subject to the floors' functional requirements, and then selects both the foundation system type and the type of horizontal load system (the latter have yet to be implemented). Its output includes the basic grid of the gravity load support systems, (building axes) and the location of the center of the core(s). The module uses the data bases of Functional Systems and Work Assemblies (expressed as IPTs) as well as a Land Data Base [4].

Next, the *Floor Design Knowledge Module* (FDKM) fixes the specific shape and sub-division of each of the core spaces around the core positions, which were set by the SSKM. Candidate IPTs for cores of different compositions and geometries are evaluated, and the selected IPT is processed to lay out the core spaces. This has not been detailed in the current work.

Lastly, the *Functional System Knowledge Module* (FSKM) selects a work assembly for each structural assembly, using IPTs stored in the Work Assembly Data Base, and lays out the structural elements. For the slab systems, the elements are columns, beams, slab strips, etc. The input is the gravity load system grid, the core spaces and the partitions. The module also uses external data base files, containing information on materials, imposed loads, design codes, etc.

In the Detailed Design Phase, the *Work Assembly Knowledge Modules* (WAKMs) perform detailed design of the elements of each Work Assembly in the building. The dimensions and other specifications of each element are set, and the activity and resource data objects are instanced and evaluated. Each kind of Work Assembly has a specific WAKM, much of whose functioning involves initiating the methods and rule sets of the IPTs on which the Assembly was based.

Of course, some of the knowledge used by the various design modules is common to two or more of them. For example, rules governing the feasibility of a particular type of slab may be used in both the structural scheme design and in functional system selection processes. This has been implemented by embedding the specific knowledge about a technical solution within its Intelligent Parametric Template. It should be noted that the approach applied here to the structure might equally be applied to other building systems; terms such as 'Functional System' and 'Work Assembly' are generic.

5. Structural system knowledge module (SSKM)

The structure of any building must satisfy three key structural requirements: support of gravity loads, resistance of horizontal loads and foundation. A technological solution must be selected for each requirement. Candidate IPTs are checked for feasibility, those found feasible are evaluated, and that with the lowest cost is selected.

Throughout the following discussion, a numerical example is used to enhance the explanation. The basic data describing this example are; building plan dimensions of 23×27 m, seven floors, open-plan office space is required on all floors, and the core space must include a stairwell and three elevator shafts. Symmetric positioning of the cores is assumed; in all of the core positioning IPTs the cores are placed such that their combined center of stiffness is coincident with the center of inertia of the building as a whole. For the sample building, fire escape access rules and occupant traffic requirements allowed a single central core solution to be selected.

The minimum possible column spacing in each direction (X and Y) is set by rules that express the architectural requirements imposed by the functions of the building spaces. For example, the minimum column spacing that allows open-plan office design is 6 m [30,31]. In order to generate candidate sets of column spacings in each direction, the overall building dimensions are divided into increasingly larger numbers of spans until the minimum column spacing is reached. The 27-m building depth can be divided into two spans of 13.5 m, three spans of 9.0 m, or four spans of 6.75 m; division into five spans of 5.4 m would be insufficient for open-plan offices. Additional rules check whether the building's width is less than three times the minimum column spacing: if so, it would be convenient to place a narrower corridor along the length of the building, and the minimum spacing for one of the longitudinal bays would be set to match the minimum width for a corridor.

In order to select the column spacing that will result in the lowest overall structure cost, the available slab solutions must be assessed. The SSKM first assesses possible solutions by processing the feasibility rules contained in the IPT for each available slab solution, and for each proposed set of column spacings. The factors that govern the feasibility of a particular slab type include: the type of solution selected for resisting horizontal loads, the magnitude of the loads on the slab, the function of the space below the slab and other non-structural systems and considerations, and the availability of necessary construction equipment and competent work teams.

Table 1
Sample rules of a ribbed slab IPT

Ribbed (Joist) Slab Rule (1)	Source (2)	Comment/Note (3)
The minimum slab depth required is at least 1/24 of the larger column spacing	[32], Expert interviews	Joists (ribs) will span in the longer direction
The minimum depth required is at least 1/16 of the smaller column spacing	[32], Expert interviews	Hidden beams will span in the shorter direction
The minimum depth of the slab is at least 15 cm	[33]	

Three sample rules for a Ribbed (Joist) Reinforced Concrete Slab without drop beams, which enable checking whether the slab will conform to a maximum depth restriction imposed by overall building height restrictions and minimum clear heights required for each floor function, are listed in Table 1.

Each of the possible slab type/column spacing combinations that are feasible are then evaluated and compared. The direct cost of the slab itself, and the cost of external building finishes, which must be added for any increase in building height, increase as spans are lengthened; other costs, such as the cost of the building core and the foundation cost, decrease as column spacing is increased. Also, the value of the resulting space to the user may rise as spans increase. All of the costs are combined, so that different column spacing and slab combinations can be compared (currently, only direct costs are computed). The cost factors are estimated by processing the IPT evaluation rules of each feasible slab solution. The base costs are drawn from a database of construction costs similar in format to the 'R.S. Means Square Foot Costs' catalog [34], in which the costs per unit area of different assemblies are listed as a function of the assembly type and the typical span or bay size.

The final column spacings can now be selected. The user is presented with a graph showing the relationship of cost per unit area to column spacing for different technical solutions. The lowest cost

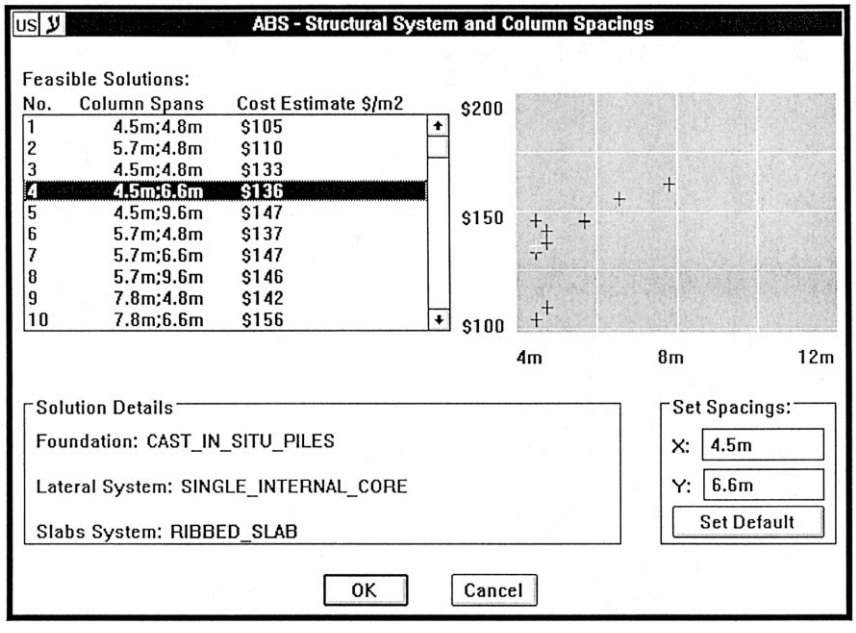

Fig. 6. User Interface showing the relationship of cost per unit area to span for feasible slab solutions.

solution, which also satisfies all other criteria (in this case at least one direction greater than 6.0 m), is highlighted as the default (Fig. 6). For the example building, spans of 4.5 and 6.6 m are suggested. After approval by the user, the axes are placed at these spacings.

6. Floor design knowledge module (FDKM)

In the current implementation, the secondary spaces that make up the building's core(s) are now laid out by selecting and running a core IPT, which arranges the stair wells, elevator shafts, etc. (note that only the position of the core was set in the previous stage).

7. Functional system knowledge module (FSKM)

Once floor layouts are complete, the FSKM selects a work assembly for each of the functional systems, and lays out the elements of each work assembly. The main structural goal at this stage is to select a solution for each of the slabs and lay out its elements. The slab assembly according to which the spacings were set cannot be adopted automatically as the slab system for all of the building's floors. There are three reasons for this:

- The final column spacings may be different from those which were associated with the optimum slab system at the structural scheme design phase,
- More detailed project data is now available (e.g. imposed loads, core walls, etc.)
- The same slab system may not be appropriate for all of the floors of the building. It is possible that floors with loading or other constraints (such as height limits), which differ from the norm, may have different slab systems. Slabs in parking floors, for example, are likely to be different from those in upper floors.

For each floor, selection rules are processed for each feasible slab solution IPT. At this stage, the level of detail is greater than that at the schematic stage: the spans have already been fixed, and specific floor loads and other performance parameters have been set and can be used for detailed evaluation of the slab. The most economical slab solution is selected from those that are feasible. The user can peruse the selections and change them if necessary. As can be seen in Fig. 7, the system selected a reinforced concrete ribbed (joist) slab with hidden beams for floor 6 of the example building. The nominal depth, span directions, and nominal span lengths of the slab are calculated and stored in an instance of the 'Ribbed_Slab' class. In the example, the nominal slab depth is 30 cm, the beams span in the X direction with nominal spans of 4.5 m, and the ribs span in the Y direction with nominal spans of 6.6 m. Note that the opening in the slab for the stairs and elevators has not yet been taken into account.

The slab IPT now lays out its elements by processing its methods and rules. The slab solution IPTs are based principally on the 'Work Assembly' and 'Element' class definitions of the Building Project Data Model (as shown in Fig. 8). Thus, one Work Assembly object is inserted in the Building Project Model to represent the slab (the object is 'instanced'); next, the IPT 'instances' multiple Element objects for the beams, columns, slab strips or slab sections, etc., and links them to the Work Assembly instance.

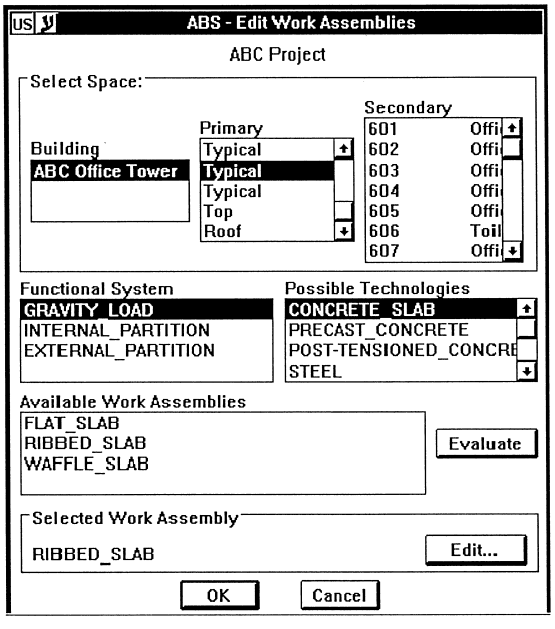

Fig. 7. Work Assembly Selection Screen.

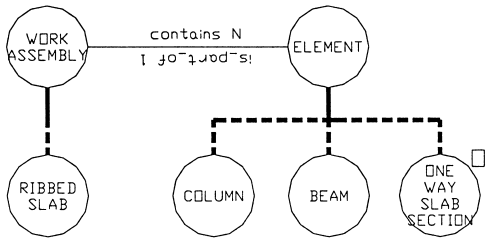

Fig. 8. Ribbed Slab IPT Classes and Relationships (dashed lines denote intermediate classes in the hierarchy, which are not shown).

In the example, the RC ribbed slab IPT will instance one 'Ribbed_Slab' (work assembly), a 'Column' (element) for each column, a 'Beam' (element) object for each beam, a 'One_Way_Slab_Section' (element) object for each unique field of continuous joists, and an 'Activity' object as follows.

(1) *Columns*. The columns are included with the work assembly of each floor as they will be built together with the slab (the advantage of this approach is that it effectively supports the construction planning view of the building). Columns are placed at each intersection of building axes, except for those intersections that fall within a core secondary space, or those that fall within one passage width (d_w) from a core wall. As can be seen in Fig. 9, two situations are possible. In the first case, where the distance from the proposed column to a core wall (d_{wc}) is less than d_w, the slab support lines are changed in order to utilize the core for support. The ends of the core walls are defined as support points and the beams or slab strips will rest on them. In the second case, where $d_{wc} > d_w$, columns are placed at the intersections of the building axes.

(2) *Beams*. The rules for beam layout for the RC Ribbed Slab IPT state that a beam will be placed between each two columns along an axis, or between a column and a wall along the axis, in the beam span direction. While the columns are linked to the Slab work assembly, the core walls are not, and so determining which core walls will participate in supporting beams requires that the method scan all the core walls in order to take them into account in laying out the beams.

A wall can support a beam if the beam axis falls along the wall length, or if it falls a short distance w_s from the wall, whether parallel or at right angles; in this case, the beam axis is diverted to the wall edge. The algorithm for locating support points along the walls is shown in Fig. 10. At each such point, a

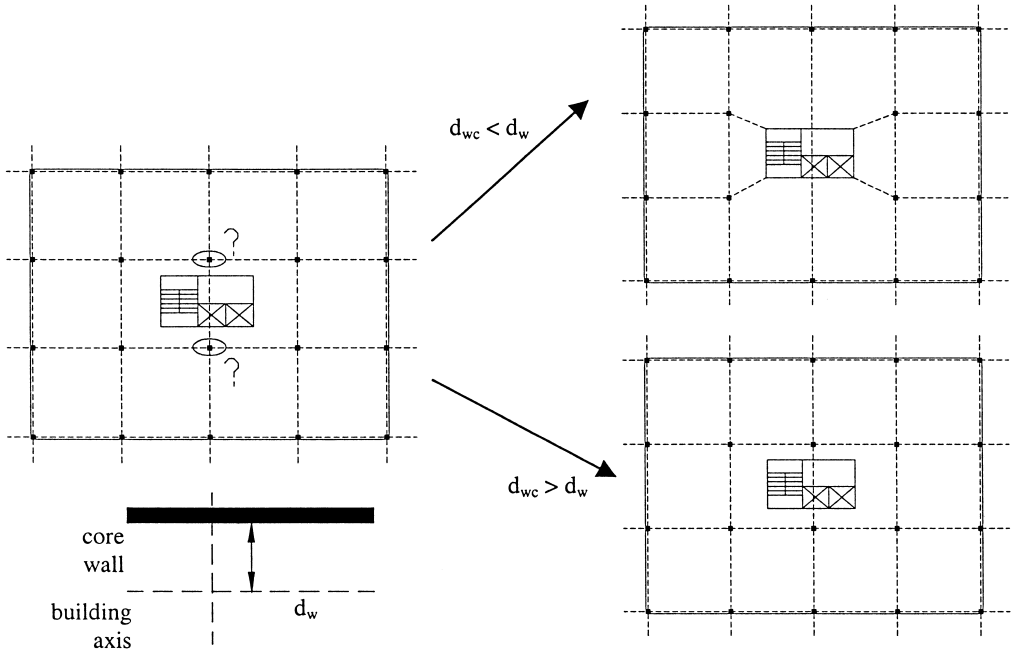

Fig. 9. Placement of Columns near Building Cores.

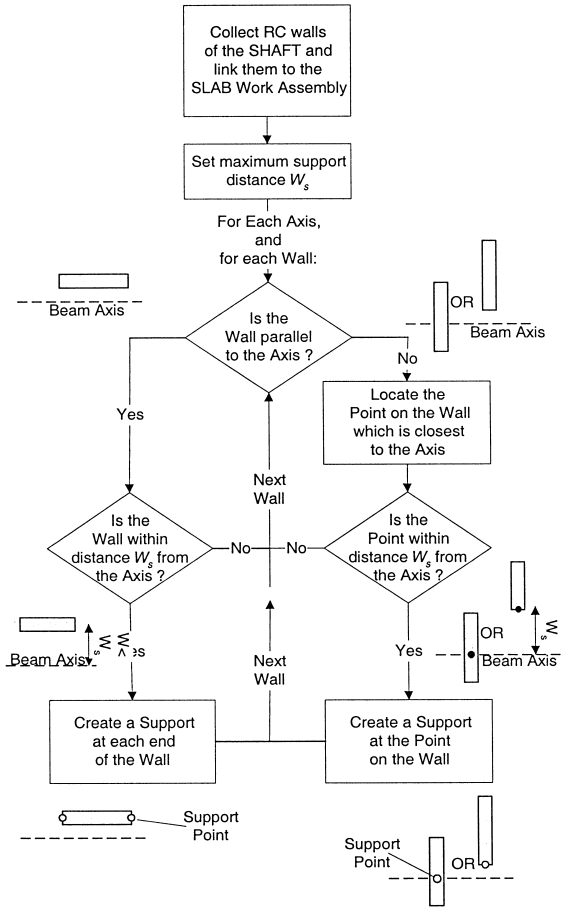

Fig. 10. Algorithm for instantiating supports and beams.

special topological object instance, called a 'Structural_Support', is created and linked to the column or wall. Once all of the supports are created, beams are

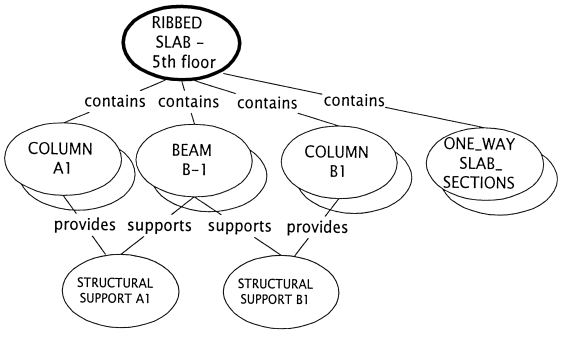

Fig. 11. Diagram of model object instances — columns, walls and beams.

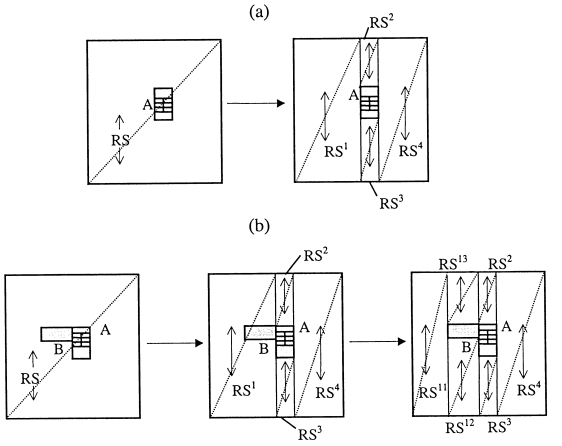

Fig. 12. Example of sub-division of a slab into one-way strips.

laid out between them, except where two support points are linked to the same wall (no beams are required along walls). The links between the beams, structural supports and columns/walls are retained, as they are later used during structural analysis and detailing. Fig. 11 shows these object instances schematically.

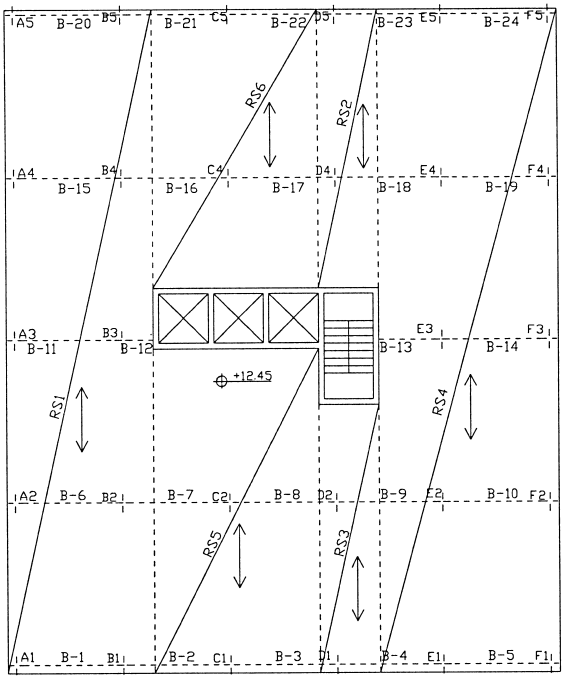

Fig. 13. Ribbed Slab Plan after placing all elements.

(3) *Slab section.* In the example IPT, slab section layout is done using a recursive algorithm, which accounts for one opening in the slab at each recursion, as shown in principle in Fig. 12(a). The algorithm starts with a single slab section (RS) spread over the entire floor area. It then replaces the slab section with new parallel and contiguous sections (RS^1 to RS^4), removing one opening area (denoted 'A' in the figure). Fig. 12(b) shows how a slab with more than one opening is sub-divided: slab section 'RS' is first divided into sections 'RS^1', to 'RS^4' considering opening 'A' only (opening 'B' is ignored). In the second step, opening 'B' is considered; this leads to slab section 'RS1' being further sub-divided into sections 'RS^{11}', 'RS^{12}' and 'RS^{13}'. In this way, any slab can be divided into unique slab sections such that each slab section contains ribs of uniform spans.

In the example under discussion, the slab obtained after layout of all of its elements is shown in Fig. 13 — the drawing itself is produced automatically using the draw methods of the IPT objects. The values of the dimensions and other parameters of the elements have not yet been computed — these will be filled in at the detailed design stage.

8. Work assembly knowledge modules (WAKMs)

In the final step of the design process, the Work Assembly Knowledge Modules (WAKMs) detail the elements. These modules use IPT methods to analyze the work assemblies and to compute the details of the elements of which they are comprised. For the structural work assemblies, this includes structural analysis and element design (selection of standard steel sections or, in the case of reinforced concrete, fixing member dimensions and reinforcing detailing).

The functioning of a typical WAKM performing detailed design is presented by continuing the exam-

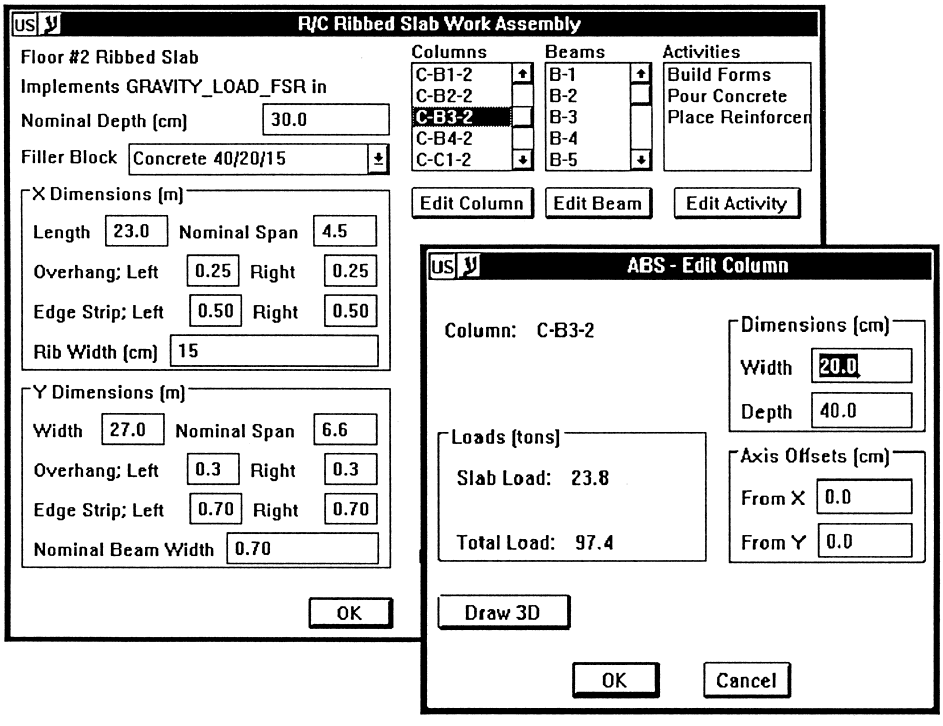

Fig. 14. Ribbed Slab IPT user interface dialog screen.

ple begun in the previous section. At the layout stage, the slab was divided into distinct strips, in such a way that all of the ribs in a given section have identical spans. The initial thickness for the slab is assumed to be the thickness calculated during the general design stage. The initial sizes of the columns are set by a method of the column class; at this time, the initial slab depth is checked for resistance to punching shear at the head of each column. A sizing method uses production rules (based on the nominal slab loads, rib and beam spans and non-structural requirements), and the Resource Requirements Data Base, to select the filler block type and size, and to set initial dimensions for the widths of each rib and beam. The user can edit these selections: Fig. 14 shows an instance of a Ribbed Slab IPT, together with a secondary dialog box which shows the data for one of the columns.

The detailed design method then continues by analyzing each set of ribs (a 'one_way_slab_section' instance) as continuous beams of 'T' cross-section. The resulting moment, shear and deflection results are stored in temporary instances of the 'RC_Design_Section' class, which are placed at each end and at the center of each span of each rib section. This class defines a typical reinforced concrete cross section with attributes of section depths and widths, top and bottom steel reinforcement areas, and structural analysis results (moments, shears and deflections). It has a method that calculates the steel area required at the top and bottom of the Section for resisting the applied moments. The beams support the ribs; the reaction results from the rib analysis define the beam loadings. The beams are also analyzed as continuous beams, also of 'T' cross section, using the same algorithm as is used for the ribs.

After structural analysis, the steel reinforcement is detailed. As defined by the Building Project Data Model, material resources are consumed by 'Basic_Activities'. Therefore, an instance of the 'Fix_Steel' 'basic activity' class is first created and linked to the Flat Slab activity. The Ribbed Slab IPT places reinforcing mesh (for cracking prevention) over the ribs, places individual bars along each rib and above each rib over the beams, and places individual bars and stirrups along each beam, and along each beam above each column.

The information for each group of bars in an individual element (beam, column, etc.) is stored in an instance of the 'Rebar_Use' class. Each such instance is linked to a slab work assembly through the fix steel basic activity. At first, the bar group is defined only by its position, orientation, and length. A 'Rebar_Use' class method then calculates and sets the number of bars, the bar diameter, material type and shape. While the number of bars in a group is stored in the 'Rebar_Use' instance, the diameter, type and shape are stored in instances of the 'Rebar' class. Fig. 15 shows (a) how these classes are related, (b) how they are instanced, and (c) how the rebars are drawn. This data structure allows a later method to collect all similar 'Rebar' instances and rationalize them into as small as possible a list of different bars for manufacture and delivery to site.

The ABS can produce outputs (drawings, reinforcing schedules, bills of quantities, etc.) automatically. In the current implementation, the production of detailed structural drawings, reinforcing mesh schedules, and isometric views have been success-

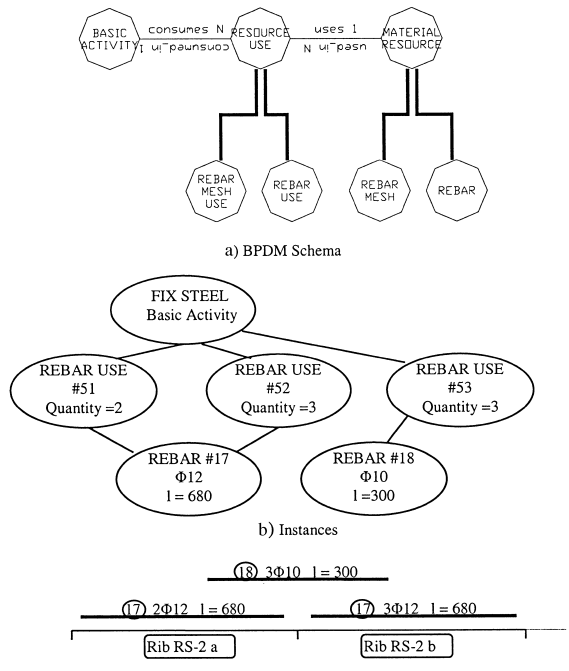

Fig. 15. The Definition of Reinforcement in the Building Project Data Model.

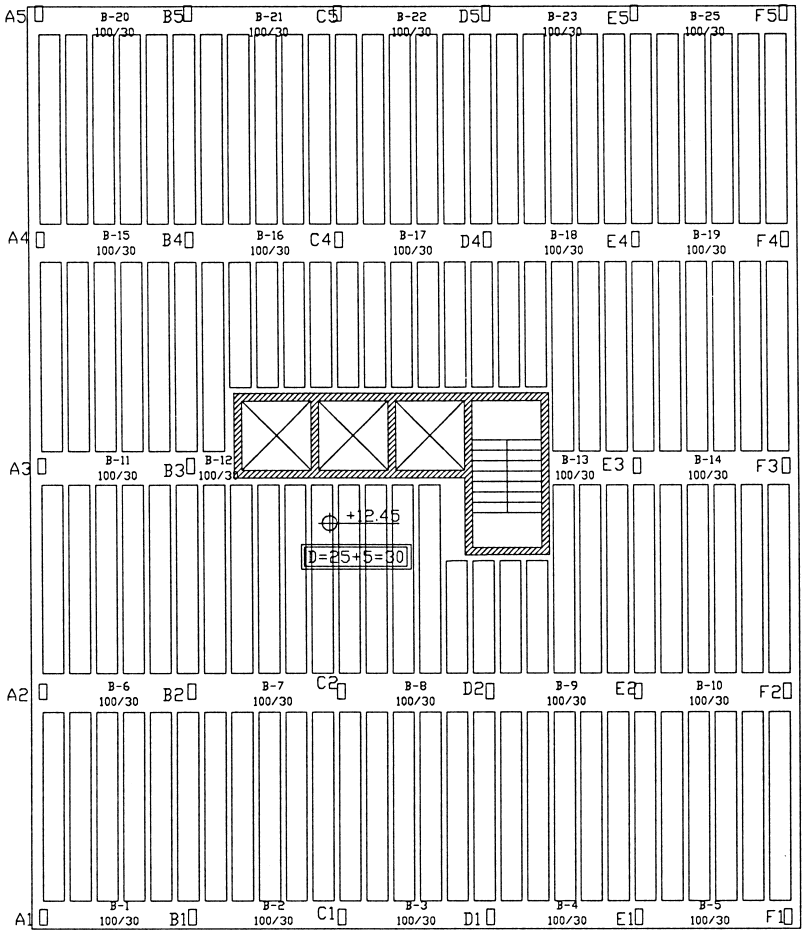

Fig. 16. Detailed Ribbed Slab filler block layout drawing.

fully tested. Fig. 16 shows the resulting detailed ribbed slab layout.

9. Discussion and conclusions

A prototype computer system for the structural design of rectangular-shaped buildings with rectangular floors has been implemented as a vehicle for testing the 'Intelligent Parametric Template' (IPT) strategy for automated design. The system is part of a general framework of an Automated Building System and is based on the Building Project Data Model designed for the automated system. The structural design modules presented in this paper highlight a number of features of the Automated Building System, its Building Project Data Model, and the IPT approach to design automation.

As distinct from the data sharing approach adopted in most Building Project Data Model development efforts, the ABS addresses Computer Integrated Construction not only as an issue of data sharing between project participants, but also aims for comprehensive automation of the building process. For this reason, the data model of the ABS is specifically designed to suit the system's processing stages. It closely links the three main aspects of project data (Space, Product and Activity) at each of three main levels of detail, to provide a coherent basis for storing the information that the ABS modules produce.

The IPT approach is different to that of most other AI-based tools. Instead of collecting all of the knowledge of a particular design domain in a knowledge base (which includes a system decomposition tree, constraints, functions and rules), the knowledge required for each design solution is encapsulated with that solution. IPTs encapsulate production rules, simple and AI algorithms, data base queries and user queries together with object definitions that are rooted in the BPDM. Thus, addition of a new building technology does not require editing the knowledge base, but simply programming the new technology's objects and behavior and then registering its existence in the system. The object-oriented nature of the approach also enables the effort that must be invested in programming IPTs to be distributed among various organizations in the industry; the fact that it is based on a common Building Project Data Model ensures compatibility. Inheritance of object class definitions, rule sets and object methods make the addition of new IPTs relatively cheap when compared with the investments required for expanding existing software or expert systems. These features may prove valuable in enabling the construction industry to overcome the inflexibility inherent in its legacy software, which has been a significant obstacle to the development of computer integrated construction.

The system's success in producing designs for rectangular buildings of various sizes and heights, with different core configurations, using only one IPT for each solution type, indicates the suitability of the IPT approach for automating structural design. The approach combines knowledge-based programming with project model technology and geometric templates: an IPT can be defined as "a collection of object definitions, rules and methods, which fully describe a set of building elements and define the ways in which they can be inserted into a building model". Two full IPTs, for design and detailing of flat and ribbed reinforced concrete slabs, have been implemented and tested.

The role of the users of the system (the owner/developer, professional consultants, contractors, etc.) has been clarified. Although the system may be technically capable of progressing without user involvement, it is not realistic to expect that such designs will be acceptable. This is because requirements are not stated explicitly in certain cases, and because the system cannot deal with local deviations from regular design that cannot be predicted a priori. The value of the system is not in achieving complete automation, but in achieving a degree of automation which relieves the users from the vast majority of repetitive design tasks.

The system has two important limitations.

At present, it only deals with rectangular-shaped buildings with uniform floor sizes. It is expected that preparation of IPTs for buildings whose shape is composed of rectangles should be straightforward; however, the use of the IPT method for arbitrary building shapes is not contemplated.

Initially, each basic IPT must be programmed. The IPT method does not provide any solution to the problems associated with knowledge elicitation for knowledge-based expert systems [7]. The system does not at present have any automated 'learning' capability, although this is desirable in a full system.

While certain issues have yet to be resolved (such as expanding the system to deal with more complex building shapes, layout of rooms on a building floor, and developing a model for assessing the value of net space to the developer), these are mostly technical in nature. The present system indicates that large potential savings could be achieved through use of a full-scale system in the building industry. Firstly, the time required for a user to produce design alternatives, and to evaluate their cost, would be reduced from days to hours. This could both reduce direct design costs and enable consideration of more alternatives than is currently practical. Secondly, the system is capable of producing designs of high quality, in the sense that the solutions selected are cost-effective and fulfill the requirements defined for them. Lastly, the integrated nature of the system could reduce the construction costs associated with design coordination conflicts between the designs of different disciplines.

References

[1] F.P. Tolman, Product modeling standards for the building and construction industry: past, present and future, Autom. Constr. 8 (1999) 227–235.

[2] P. Teicholz, M. Fischer, Strategy for computer integrated construction technology, ASCE J. Constr. Eng. Manage. 120 (1) (1994) March.

[3] T. Seebohm, W. Wallace, Rule-based representation of design in architectural practice, Autom. Constr. 8 (1998) 73–85.
[4] R. Sacks, A. Warszawski, A project model for an automated building system: design and planning phases, Autom. Constr. 7 (1997) 21–34.
[5] A. Warszawski, Industrialization and Robotics in Building, Harper and Row, New York, 1990.
[6] R. Sacks, Issues in the development and implementation of a project model for an automated building system, Int. J. Constr. Inf. Technol. 5 (2) (1998) 75–101, Salford University, Salford, UK.
[7] F. Hayes-Roth, D.B. Lenat, D.A. Waterman (Eds.), Building Expert Systems, Addison-Wesley, Reading, MA, 1983.
[8] A. Aamodt, E. Plaza, Case-based reasoning: foundational issues, methodological variations, and system approaches, Artif. Intell. Commun. 7 (1) (1996).
[9] K.-W. Park, D.E. Grierson, Pareto-optimal conceptual design of the structural layout of buildings using a multicriteria genetic algorithm, Comput.-Aided Civ. Infrastructure Eng. 14 (1999) 163–170.
[10] A. Badiru, A. Arif, FLEXPERT: facility layout expert system using fuzzy linguistic relationship codes, IIE Trans. 28 (4) (1996) 295–309.
[11] R.S. Liggett, Automated facilities layout: past, present and future, Autom. Constr. 9 (2000) 197–215.
[12] R. Sause, G.H. Powell, Design process for computer integrated structural engineering, Eng. Comput. 6 (3) (1990) July.
[13] M.L. Maher, Preliminary design: HI-RISE, in: H. Adeli (Ed.), Chapter 6.4 Expert Systems for Structural Design, Prentice-Hall, Englewood-Cliffs, NJ, 1988.
[14] G.E. Tsakalias, KTISMA: A blackboard system for structural model synthesis of asymmetrical skeletal reinforced concrete buildings, in: B.H.V. Topping, M. Papadrakakis (Eds.), Artificial Intelligence and Object-Oriented Approaches for Structural Engineering, CIVIL-COMP Press, Edinburgh, Scotland, 1994.
[15] R. Sacks, O. Buyukozturk, Expert interactive design of R/C columns under biaxial bending, ASCE J. Comput. Civ. Eng. 1 (2) (1987) April.
[16] A. Retik, A. Warszawski, Automated design of prefabricated building, Build. Environ. 29 (4) (1994) 421–436, Elsevier.
[17] S. Fenves, U. Flemming, C. Hendrickson, M.L. Maher, R. Quadrel, M. Terk, R. Woodbury, Concurrent Computer Integrated Building Design, Prentice-Hall, Englewood Cliffs, NJ, 1994.
[18] E.A. Domeshek, J.L. Kolodner, A case-based design aid for architecture, Artificial Intelligence in Design '92, Kluwer Academic Publishers, Boston, MA, 1992, pp. 497–516.
[19] R.E. Oxman, Case based reasoning in knowledge based design, Proceedings of European Symposium on Management, Quality and Economics in Housing and Other Building Sectors, Lisbon, Portugal, September, 1991.
[20] S.F. Bailey, I.F.C. Smith, Case-based preliminary building design, ASCE J. Comput. Civ. Eng. 8 (4) (1994) October.
[21] J. Gero, V. Kazakov, Space layout problems using evolved design genes, Artif. Intell. Eng. 12 (3) (1998) 163–176.
[22] I. Watson, The case for case-based reasoning, http://www.salford.ac.uk/survey/staff/IWatson/ita01.htm.
[23] O. Shaked, A. Warszawski, Knowledge based system for construction planning of high rise buildings, ASCE J. Constr. Eng. Manage. 121 (2) (1995).
[24] B.-C. Bjork, A unified approach for modelling construction information, Build. Environ. 27 (2) (1992) Pergamon, Press, Oxford, UK.
[25] ISO 10303, Product Data Representation and Exchange, International Standard ISO 10303, International Organization for Standardization, Geneva Switzerland, 1993.
[26] C.S. Han, J.C. Kunz, K.H. Law, Building design services in a distributed architecture, ASCE J. Comput. Civ. Eng. (1999) January.
[27] A. Warszawski, R. Sacks, The Project Model of an Automated Building System, Modeling of Buildings Through their Life Cycle, CIBW78 Workshop Proceedings, CIB Publication No.180, August 195.
[28] R. Sacks, Computer Integrated Construction: Project Data Model Representation and Use, unpublished DSc dissertation, Faculty of Civil Engineering, Technion Israel Institute of Technology, 1997.
[29] G. Pahl, W. Beitz, Engineering Design — A Systematic Approach, Springer, Berlin, 1995.
[30] R. Crane, M. Dixon, Office Spaces — Architect's Data Sheets, Architecture Design and Technology Press, London, 1991.
[31] E. Neufert, Architect's Data, 2nd edn., Granada, Wiley, New York, 1984.
[32] E. Allen, Fundamentals of Building Construction, 2nd edn., Wiley, New York, 1990.
[33] ACI 318-94, ACI Manual of Concrete Practice, Part 3, American Concrete Institute, Detroit, MI, 194.
[34] R.S. Means, Means Square Foot Costs, 16th edn., R.S. Means Company, Kingston, MA, 1995.