USING BUILDING TOPOLOGICAL INFORMATION TO CHECK FOR MEANS OF EGRESS BUILDING CODE COMPLIANCE

SUBMITTED: September 2004
REVISED: February 2005
ACCEPTED: September 2005 at http://www.itcon.org/2005/15/
EDITOR: C. Anumba

Khaled Nassar
University of Maryland Eastern Shore, USA
email: knassar@umes.edu

Tang-Hung Nguyen
North Dakota State University, USA
email: tang.neyuen @ndsu.nodak.edu

SUMMARY: *Means of egress code requirements are intended to provide provisions for design, construction and layout of building elements in order to provide a continuous and unobstructed path of travel from any point in a building or structure to a public way. Although research into automating building code compliance (i.e. for means of egress code requirements) is extensive, the question of how to efficiently represent and extract building data remains open, especially with the continuous development of new building models such as the IFC (Industry Foundation Classes). This paper presents an automated technique that allows for compliance checks to be carried out directly from CAD drawings with a solid B-representation of the building's spaces. An algorithm to extract topological relationships between the building spaces is developed. This information is used to automatically generate a means of egress code compliance diagrams (MECC) for the building. Additionally, a prototype computer tool, MEC^2 is developed to automatically generate and check means of egress code issues.*

KEYWORDS*: emergency exit, building code, topological information, automated deduction, computer-aided building design.*

1. INTRODUCTION

The provisions of Means of Egress section of the International Building Code is intended to control the design, construction and arrangement of building elements required to provide a reasonably safe means of egress from all structures. Means of egress can be defined as "a continuous and unobstructed path of travel from any point in a building or structure to a public way" [IBC]. Several researchers have developed frameworks for the representation and processing of building codes specifically and for design standards in general [Han et al 1996, 1997, 2002a, 2002b, DeWaard 1992, Fenves et al 1995, Garett 1989, Yabuki 1992, Sharpe 1994, Woodbury et. Al 2000, Law et. Al 2003]. A survey of developments for computer representation of design codes was reported by Fenves [Fenves et al 1994]. In addition, code compliance has been built into commercial software for checking structural integrity as well as other design issues (Moss 1992). Automated code compliance is used by some authorities to perform an automated code compliance checks of designs submitted for building permit/approval e.g. in Singapore (BCA 2004).

In this paper, we focus on a specific application of automated building code checking for means of egress. Among the numerous code provisions governing building designs, the two issues that have been identified by facility managers as most significant are accessibility and means of egress [Han et al 1997]. While there are various techniques for automated code compliance checks (i.e. Artificial Intelligence techniques), this paper explores the most suitable data representation and extraction technique.

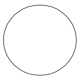
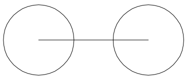
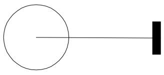
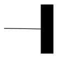
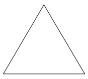
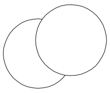
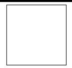
A simple solid B-representation of the building's spaces and an extraction incorporated into the IFC model to accomplish Egress Building Code Compliance Checks is developed. The developed technique allows for compliance checks to be carried out directly from CAD drawings of the building's spaces. A crucial part of achieving this is getting the computer to understand the topological relationships between the spaces of the building (e.g. adjacent-to, enclosed-in) in the CAD model. An algorithm is therefore developed to extract these relationships from a solid model representation of the building. There are several different 3D solid modeling representation techniques including Boundary representation (B-rep), Constructive Solid Geometry, and Feature-Based Representation. The research presented here makes use of 3D boundary representation or B-rep of solid modeling for representing building spaces since the B-rep representation provides both geometric and topological data necessary for deduction of spatial relationships between the building spaces required for the means of egress compliance [Nguyen and Oloufa 2001]. An algorithm is created to extract topological information from the B-rep solid model information. Furthermore, the topological relationships between the spaces, extracted by the developed algorithm, are then used to generate a Means of Egress Code Compliance (MECC) diagrams for the building. The MECC diagram is an abstract representation of the building that is proposed in this paper, which can be used to check for certain spatial or topological code compliance issues such as traveling distances, number of exit accesses etc… Although a variety of means of egress compliance checks can be performed using the developed automated technique, the main focus of this paper is on the checking the morphological and spatial aspects of building codes such as the International Building Code IBC.

The remainder of this paper is organized as follows: in the next section the MECC diagrams are described and the different modeling elements are presented. This is followed by a description of the algorithms used to generate the MECC diagrams from the B-rep solid models of the spaces and the various morphological relationships used are discussed. In section 4, an example is presented along with the computer implementation of the proposed technique. Finally, conclusions and recommendations for future work are given.

2. MEANS OF EXIT CODE COMPLIANCE (MECC) DIAGRAMS

In order to facilitate the process of checking the building design compliance with the means of egress code requirements, the MECC diagrams are proposed here. The MECC diagram is an abstract model of the building represents the various elements of the means of egress using straightforward symbols. In developing the MECC diagrams, we set forth the criteria for them to be simple so that they can be easily read and understood. In addition they had to represent several pieces of information needed to check for code compliance. Therefore a number of basic modeling symbols were developed as shown in TABLE 1. Each symbol corresponds to a particular element of the means of egress. The corridor for example represents an enclosed passageway which limits the means of egress to a single path of travel, as defined by the building codes (such as the International Building Code, IBC). An exit discharge on the other hand represents the portion of a means of egress between the termination of an exit and a public way.

TABLE 1. The MECC Modeling Elements

Means Of Egress Element	*Modeling Element*	*Code Definition*	*Properties/Attributes*
Space		-	Occupancy Load Use Code Area Accessibility Req.
Corridor		An enclosed passageway which limits the means of egress to a single path of travel	Width Surface
Passageway		An enclosed hallway or corridor that is an element of an exit, and terminates at a street or an open space or court communicating with a street	Width Surface
Exit access		Exit access is that portion of a means of egress which leads to an entrance to an exit	Width Height Door Type
Exit Discharge	 Public way: any street or alley or other parcel of land open to the outside air leading to a public street with a clear width and height not less than 10 feet	That portion of a means of egress between the termination of an exit and a public way	Width Height Door Type
Joined Space		-	Occupancy Use Code
Stair		-	

Building designs can be abstractly presented using MECC diagrams and in turn code issues can be checked either manually or automatically using a computer e.g., traveling distances checks as well as egress through adjoining spaces. Alternatively these code checks can be made automatically using computer software and code violations can be identified and marked, as will be presented in the developed prototype.

Although the MECC diagrams are intended to check for spatial or morphological code compliance issues (such as traveling distances, adjacency constraints and, connectivity constraints) they can also check other non-spatial or morphological issues. Each modeling element has a set of attributes or properties attached to it that represent an inherit property of the modeling element. For example, each space can have an attribute or property to specify its use (i.e. whether it is hazardous use or not), occupancy load (measured by number of people), etc… These properties/attributes can be then used to set constraints on their values so that they can be automatically evaluated (e.g. a constraint on the width of a corridor). Although MECC diagrams can be drawn manually for any building design, an algorithm is needed to automate MECC diagram generation. The crucial part of this algorithm is getting the computer to understand the topological relationships between the spaces in the building. This is described in the next section.

3. EXTRACTION AND GENERATION OF MECC DIAGRAMS

Formal representation of spatial data in general and topological data in particular of building components is a complex and challenging task in developing building design systems such as systems for code compliance. The complexity is in part due to the fact that each professional usually utilizes his/her own representation of topologies and dimensionalities to express spatial information of building components. Furthermore, different design tasks require different types of topological information. For example, information about the walls surrounding a particular space (e.g. hazardous occupancy) is needed for code compliance checking of that space, details of connections between individual structural members of a reinforced concrete frame should be provided for reasoning about constructability, and information about the adjacency among floors of a high-rise building is required for planning the sequence of construction activities.

In order to automatically generate the MECC diagrams, the topological information of the building space has to be extracted and analyzed so that the various topological relationships between two building components can be deduced. Therefore, an algorithm to initially extract B-rep geometric data from the CAD system and then checks for conditions of topological relationships is needed. The deduction task becomes more challenging since there is a variety of topological information to be used among different AEC professionals. While many different types of topological relations are used by AEC professionals, the current research effort on development of deduction algorithms will be concentrated on four of the five major categories of topological information classified by (Nguyen and Oloufa, 2001):

- *Adjacency*: One component is *adjacent to* the other
- *Separation*: Components are *separate from* each other
- *Containment*: One component lies within the other
- *Intersection*: One component intersects the other

The fifth relationship is connectivity and although it is essential for other automated code checks it is not needed to perform code checks for means of egress code analysis as will be described below.

The building spaces are represented as solid modeling objects in CAD environments. The solids being considered here in those CAD systems are represented through a B-rep modeling scheme. The information that an algorithm needs for deducing the spatial relationships can be obtained from the B-rep model of the solids and consists of vertex, edge, face, cell, and loop information. Basically, the determination of various spatial relationships between building components requires the relationships between vertices and faces of those components. There are three possible relationships between a vertex and a face, i.e. the vertex could lies on the face, to the right or above, or to the left or below the face, which can be identified by zero, positive, and negative values respectively.

Depending on the position of a vertex with respect to a face of a building object (e.g. space), the value representing the vertex-face relationship, called Relation Index, is assigned to 1, -1, and 0 indicating the vertex is outside, inside, and on the object respectively. These Relation Indexes are obtained by substituting the coordinates of the vertices in the equations of the face of the object. These indexes are used to determine whether two given building components are *adjacent to*, *contained in*, *intersected with*, or *separate from* each other.

Given a number of different spaces, the question is to determine these topological relations among these different building spaces. The algorithms for deduction of the four topological relations (i.e. adjacency, containment, separation and, intersection) among different spaces are described below. The following assumptions are made:

1. All outward normal vectors of faces are defined by the right hand side rule which takes the list of vertices to be counter-clockwise and specifies the outward normal to be the one giving a positive value for points outside the space, as shown in Figure 4.6.
2. All faces comprising a building space are assumed to be convex.
3. All faces bounding a building spaces are planar
4. The values obtained by the substitution of the coordinates of the vertices in the equation of the faces are called Relation Indexes (RI). The RI values resulted in the right hand side of the equation could be positive, negative, or zero, indicating that the vertex is outside, inside, or on the space respectively.

TABLE 2. Topological relationships and their inputs, outputs and, algorithms

RELATION	INPUT	OUTPUT	ALGORITHM
Adjacency	S1, S2, and S3 as spaces of a building where S1 ≠ S2 ≠ S3	S1 is *next-to* S2 and *below* S3	1. For every vertex of S2 (or S3) and every face of S1, determine vertices of S2 lie on a face of S1 (i.e. vertices that give zero RI values with respect to the face). These vertices define the common face between S1 and S2 (or S3). 2. For vertices other than those of the common face of S2 (or S3), compute the RI's indicating their positions with respect to S1. 3. If the RI is positive and the outward normal vector of the common face is parallel to XY plane, then S2 is *next-to* S1. In the case of S3 that also shares a common face with S1 and the outward normal vector of the common face is perpendicular to the XY plane, then S1 is *below* S3 or S3 is *above* S1.
Containment	S1 and S2 as two spaces of a building where S1 ≠ S2	S2 is *contained-in* S1	1. For every vertex of S2, determine the Relation Index with respect to all faces of S1. 2. If all the Relation Indices are negative (i.e. all vertices of S2 lie inside S1), then S2 is fully *contained-in* S1. If some of the RI's are zero (i.e. some of the vertices of S2 lie on faces of S1), then S2 is *contained-in* S1 and has a face, edge, or vertex touching S1.
Separation	S1 and S2 as two building components represented by 3D solids where S1 ≠ S2	S1 is *separate* from S2	1. For a face of S1, compute the Relation Indices for all of the vertices of S2 with respect to the face of S1. 2. If the Relation Indices for all of the vertices of S2 are positive, then all of the vertices are outside S1, i.e. the two building components S1 and S2 are *separate*. 3. If there are more than three vertices of S2 for which the Relation Indices are zero with respect to the face of S1 and the Relation Indices for other vertices of S2 are positive, then the two building components are *adjacent-to* each other.
Intersection	S1 and S2 as two spaces of a building where S1 ≠ S2	S1 intersects with S2	1. For every face (say F1) of S1 and for two different vertices v1 and v2 of any face (say F2) of S2, compute the Relation Indices RI1 and RI2 for the vertices v1 and v2 respectively with respect to F1. 2. If RI1 x RI2 < 0 (i.e. v1 and v2 are located in two different regions of F1) and for two different vertices v3 and v4 of the face F1, compute the Relation Indices RI3 and RI4 for the vertices v3 and v4 respectively with respect to face F2. 3. If RI3 x RI4 <0, then face F1 of S1 intersects with face F2 of S2.

3.1. Adjacency

The algorithm identifies spaces that are *next-to*, *above*, and *below* a given space. In other words, the algorithm determines whether the given space shares a common face with any other spaces and all vertices other than those defining the common face are outside the given space.

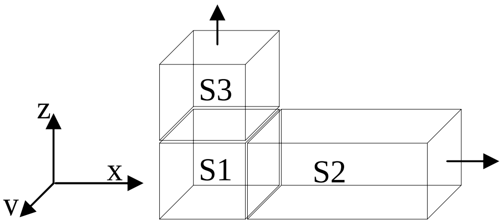
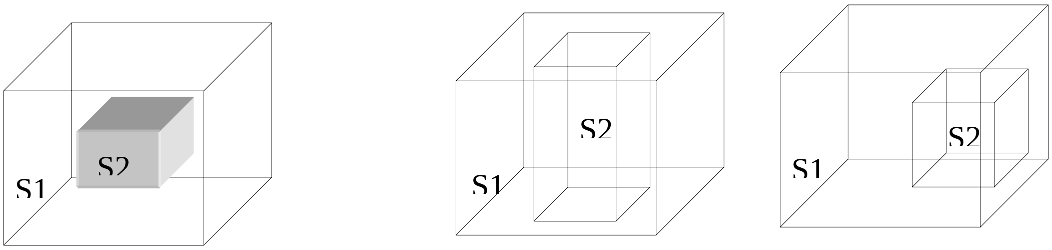


FIG.1: Adjacency Relations between Spaces

3.2. Containment

Containment of one building element by the other could exist in two cases: *fully-contained* and *face-touching*. For a building component is *fully-contained* in another component, all of its vertices should lie to the negative side of all the faces of the containing component. Therefore, the Relation Index of all of the contained component's vertices with respect to any and all the faces of the containing component should be negative. In the case that the Relation Indices for some of the vertices of the contained component are zero, the faces of the two components touch each other.



(a) Fully-Contained (b) Face-Touching

FIG. 2: Containment Relations between Building Components

3.3. Separation

Conditions to determine if the building components (e.g. spaces) are *separate* from each other are almost the converse of those for *containment*. In other words, if a face of a building component exists for which all of the vertices of the other building component lie in the positive side (i.e. the Relation Indices are equal to1), the building components are *separate* from each other.

3.4. Intersection

If conditions to determine if two spaces are *contained* and *separate* are not found then they are intersecting one another. Two different spaces intersect one another if there are two faces of the two spaces for which two vertices of the other face lie on opposite sides. In other words, the Relation Indices for two vertices of one face with respect to the other face should be opposite, i.e. 1 and –1, and vice versa.

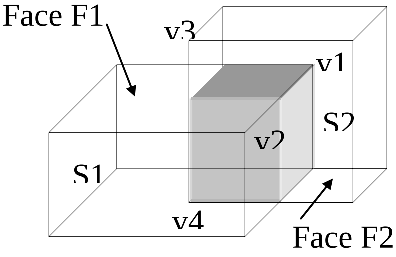
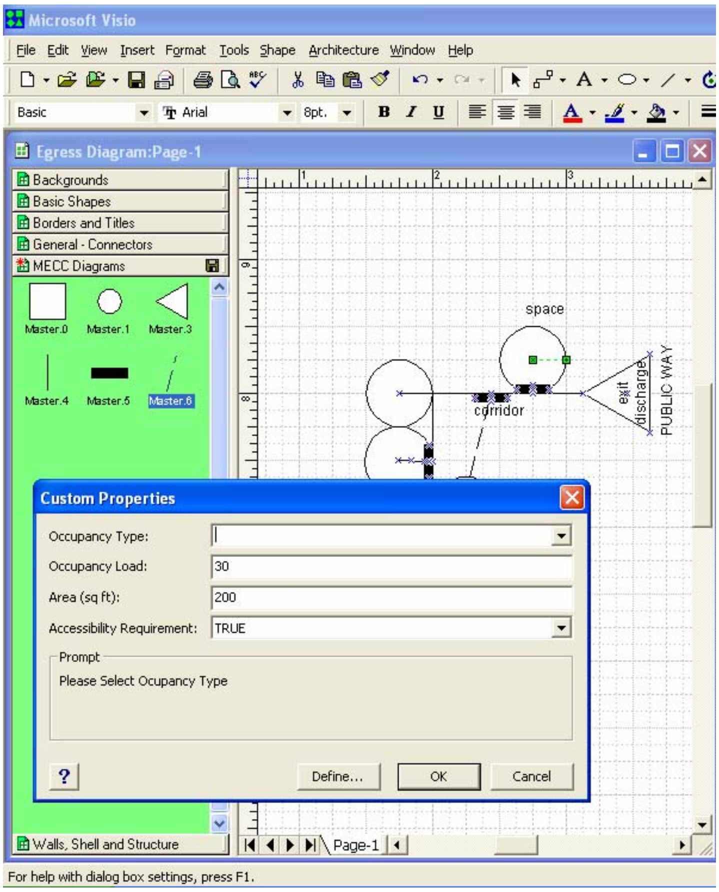


FIG.3: Intersection Relation between Spaces

Given the above algorithms for deducing the relationships between the buildings, the next step is to use and extract these relationships from an electronic building product model to automatically generate the MECC diagrams. The developed computer prototype is described below.

4. COMPUTER IMPLEMENTATION AND EXAMPLE

A prototype tool, called MEC^3 (*M*eans of *E*gress *C*ode *C*ompliance *C*hecker), was developed to automatically generate and check means of egress code issues. MEC^3 is developed as an add-on to a commercial building information modeler (BIM) namely, Architectural Desktop^{TM} (ADT^{TM}) by AutoDesk. ADT^{TM} was chosen because it supports the most current and widely acceptable standard for a building product model, Industry Foundation Classes (IFC). There have been a number of different building product models proposed over the years. Although a unified model was only a dream a few years ago, the International Alliance of Interoperability (IAI), a consortium of CAD vendors and other AEC industry partners, developed standards for a three-dimensional project model that enables interoperability between applications by different software vendors.. The IAI's effort includes defining a set of objects Industry Foundation Classes that conform to current object-oriented philosophy. A number of different vendors currently support IFC in their products. ADT^{TM} currently supports version 2.x. Although several building modeling software have incorporated IFC such as Autodesk's Architectural Desktop and Revit, the IFC is in the process of being widely accepted and used.



Occupancy Type:
Occupancy Load: 30
Area (sq ft): 200
Accessibility Requirement: TRUE
Prompt
Please Select Ocupancy Type

For help with dialog box settings, press F1.

FIG. 4: The MECC diagram analyzer

MEC^3 essentially has two main components. First, a simple ADT-to-MEC diagram converter and secondly, a MEC diagram analyzer which is used as an interface to define and analyze code compliance rules. The ADT-to-MECC converter takes as its input the ADT file and analyzes the spaces (in IFC's object model these are named IfcSpace) of the building using the topology algorithms described above to output on the other end a MECC diagram. Each space in the ADT model has a name and a solid B-rep representation. In an approach similar to that of the work by Han [Han et al 2002], we require the user to explicitly label a space (an IfcSpace) as a corridor (in our implementation, if the name of a space starts with the letter 'Cor_') and the ADT-to-MECC diagram converter examines this information to note which spaces are corridors. The software then proceeds to analyze the spaces in the building in order to identify the topological relationships between these spaces. The

MECC diagrams are then generated and are output to the second component of MEC^3, which is the MECC diagram analyzer.

The MECC diagram analyzer is the interface used to define and analyze code compliance rules, and is developed in a commercial diagramming software, VISIO. Fig. 4 shows the MECC diagram analyzer interface, which has several code issues pre-defined already. For example, allowable exit distances are defined and the tool automatically checks to make sure that they are within the acceptable limits. If exit distance exceeds allowable limits, the exit is marked in red to indicate a violation. In addition, the width and the number of required exit accesses and exit discharges are predefined and are tagged if they do not pass code requirements. Code issues that relate the topological relationships between the spaces of the building are also evaluated. An example of a topological related code issue is, *"Where the foyer is not directly connected to the public street through the main lobby, an unobstructed corridor shall be provided which leads to main entrances and exits."*

Other code rules that depend on some or a specific property of the means of egress components (e.g. the area or occupancy type of a space) are also incorporated. An example from the IBC would be "*Egress from a room or space shall not open into an adjoining or intervening room or area that is a high hazard occupancy area*". If a room or space has an occupancy property that is a high hazard, then this rule applies and is evaluated.

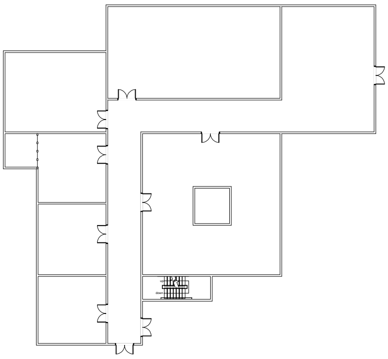


FIG. 5: Example building modeled in ADT

The MECC diagram analyzer can also be used as a stand alone tool. The user can drag and drop the different Means of Egress components and generate their own MECC diagrams. The example building shown in Fig. 5 was drawn in ADT. The various spaces have been drawn and the corridor spaces have been identified. Once this is complete the ADT-to-MEC diagram converter is initiated within the ADT drafting environment in order to generate the MEC diagram shown in Fig. 6.

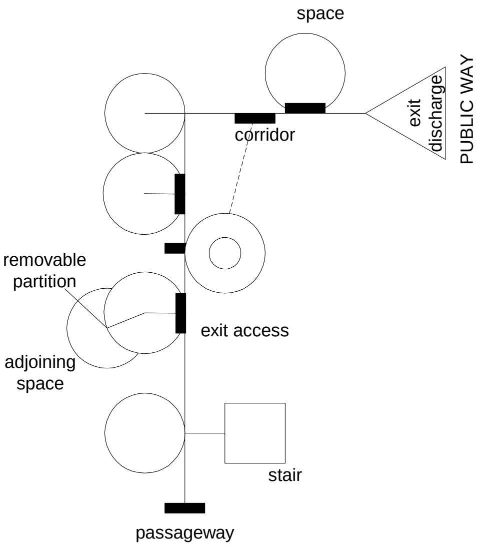


FIG.6: MECC Diagram Generated Automatically from the Building Drawing in ADT

5. LIMITATIONS AND FURTHER WORK

This paper presented an automated technique to check for means of egress code requirements. Means of egress code compliance diagrams are suggested as an abstract representation of the building design for analyzing and verifying code constraints. Topological relationships between the building spaces needed to generate MECC diagrams are suggested and algorithms for the extracting these relationships from solid b-rep models are presented. A computer prototype tool was also developed and presented here.

The scalability of the model and its application to other kinds of geometries and buildings is another important issue to be addressed. Although the example presented in this paper involves a rectilinear building the same topological information can be used for curvilinear buildings as well. However, the curvilinear buildings preferably need to be represented either as B-splines if the analysis is to be done in 2D or NURBS (Nonuniform Rational B-Splines). This is because B-Splines and NURBS provide the flexibility to design a large variety of shapes as well as being evaluated reasonably fast by numerically stable and accurate algorithms. The one drawback is the need for extra storage to define traditional shapes (e.g. circles).

Also other kinds of code checking problems can be addressed using the defined representation and algorithm as long as topological information is the primary requirement to perform the code checking. Application of the representation to structural problems that require information about the location and relationships between structural members for example is possible. In this case however, the fifth topological relationship, namely connectivity, needs to added to the system since this relationship is often used to refer to connection between two structural members (such as a column and a beam or a load bearing wall and a slab, etc…) using either an *attached-to* or *supported-by* relationship.

Suggestions for future work include adding a two way connection between MECC diagrams and building design in ADT. This allows for modifications in the MECC diagram to be automatically translated into design revisions so that the building design would be automatically updated each time a modification is made in the MECC diagram. In addition adding more properties and attributes to the different components of means of diagrams would increase the usefulness of the developed prototype.

6. REFERENCES

(ADAAG97) Access Board (U.S. Architectural and Transportation Barriers Compliance Board) (1997). *Americans with Disabilities Act Accessibility Guide*, Washington, DC.

BCA (Building and Construction Authority), Singapore Code of Practice on Building Design, (2004) http://www.bca.gov.sg/industry_programmes/buildable_design/legislation/codepractice.html

California Building Code, California Building Standards Commission, (1998), http://www.bsc.ca.gov/

De Waard, Marcel (1992). Ph.D. Thesis: Computer Aided Conformance Checking: Checking Residential Building Designs Against Building Regulations with the Aid of Computers, The Hague, The Netherlands.

Eastman, C et al (2002), Strategies for Realizing the Benefits of 3D Integrated Modeling of Buildings for the AEC Industry. Proceedings of the 19th International Symposium on Automation and Robotics in Construction, Washington DC 23-25.

Fenves, S.J., Garrett, J.H., Kiliccote, H., Law, K.H., Reed, K.A. (1995) “Computer Representations of Design Standards and Building Codes: U.S. Perspective,” 21, the International Journal of Construction Information Technology, University of Salford, Salford, U.K.

Fischer, M., et al. (2001), Geometric Representations for Construction Planning and Scheduling, Stanford University, Center for Integrated Facility Engineering, (California, U.S.A).

Garrett, J.H., Basten, J., Breslin, J., Andersen, T. (1989) “An object-oriented model for building design and construction,” Proc. Struct.Congress, ASCE pp. 332-341, New York, NY.

Griffis, F.H & Sturts, C. (2001), Three-Dimensional Computer Models and the Fully Integrated and Automated Project Process for the Management of Construction, Construction Industry Institute Report RR152-11

Han C, Kunz, J C., and Law, K (2002) "Compliance Analysis for Disabled Access," Advances in Digital Government Technology, Human Factors, and Policy, Boston Kluwer.

Han, C.S., Kunz, J.C., Law, K.H. (1997) “Making Automated Building Code Checking a Reality,” Facility Management Journal, IFMA September/October 1997 pp. 22-28, Houston, TX.

Han, J. C. Latombe, J. C. Kunz, and K. H. Law (2002. "A Performance Based Approach to Wheelchair Accessible Route Analysis," Advanced Engineering Informatics, 16:53-71.

Haymaker, J. (2001), Perspectors: Inferring Spatial Relations from Building Product Models, Stanford University, Center for Integrated Facility Engineering, (California, U.S.A).

International Alliance for Interoperability (1997). Industry Foundation Classes Release 1.0, Specifications Volumes 1-4, Washington DC.

International Alliance for Interoperability (2001). http://www.iai.org. Industry Foundation Classes IFC Release 2.X

International Standards Organization, Technical Committee 184, subcommittee 4 (1994). ISO 10303-1:1994: Industrial automation systems and integration—Product data representation and exchange—Part 1: Overview and fundamental principles, Geneva, Switzerland.

Ito, K., Ueno, Y., Levitt, R.E., Darwiche, A. (1989) “Linking knowledge-based systems to CAD design data with an object-oriented building product model,” Technical Report 17, Center for Integrated Facility Engineering, Stanford University, Stanford, CA.

Kiliccote, Han (1996). Ph.D. Thesis: A Standards Processing Framework, Carnegie Mellon University.

Lau, G., Law, K.H and. Wiederhold, G (2003), A Framework for Regulation Comparison with Application to Accessibility Codes, Proceedings of the National Conference on Digital Government Research (dg.o2003), Boston, MA, May 18-21, pp. 251-254.

Moss R (2001), Eurocode: EN1992 Approaches to the design of reinforced concrete flat slabs, BRE

Nguyen, T.H. and Oloufa A.A. (2001), Computer-Generated Building Data: Topological Information. Journal of Computting in Civil Engineering, Vol. 15, No. 4, 268-274.

ObjectARX (2002), Software development Kit for AutoCAD, Autodesk Inc

Sharpe, R, Oakes, S, Haselden, P.and Davidson, I. (1994), Building Code of Australia Expert Assistant – Putting Research into Practice, Computing in Civil Engineering, Conference Proceeding , pp. 1679-1686

Vogel Andreas, and Keith Duddy (1997). Java Programming with CORBA, John Wiley and Sons, Inc., New York, NY.

Woodbury, R. F.,.Burrow, A. L, Drogemuller, R. M. and. Datta, S, (2000) Code Checking by Representation Comparison, International Journal of Design Computing, 3(1), pp. 73-89.

Yabuki, N. and Law, K.H. (1992). “An Integrated Framework for Design Standards Processing,” Technical Report 67, Center for Integrated Facility Engineering, Stanford University, Stanford, CA.

Zeid, CAD/CAM Theory and Practice, (McGraw-Hill, 1991)