

CONFERENCE THEME: INFORMATION AND COMMUNICATION TECHNOLOGIES IMPROVING EFFICIENCIES

Full Paper

AUTOMATING CODE CHECKING FOR BUILDING DESIGNS – DESIGNCHECK

Lan Ding

CSIRO Manufacturing & Infrastructure Technology, Australia

Lan.Ding@csiro.au

Robin Drogemuller

CSIRO Manufacturing & Infrastructure Technology, Australia

Robin.Drogemuller@csiro.au

Mike Rosenman

Key Centre of Design Computing and Cognition, University of Sydney, Australia

Mike@arch.usyd.edu.au

David Marchant

Woods Bagot

David.Marchant@woodsbagot.com.au

John Gero

Key Centre of Design Computing and Cognition, University of Sydney, Australia

John@arch.usyd.edu.au

ABSTRACT

This paper presents an automated code checking system (DesignCheck) that enables quick and easy compliance assessment against building codes and assists designers in finding potential problems early. The system enables modelling of extended design information and encoding of a wider range of domain knowledge embedded in building codes. It uses Industry Foundation Classes (IFCs) as a common model to transfer 3D object-based CAD models to the DesignCheck internal model. The DesignCheck internal model allows for the definition of comprehensive design information as well as identical description mapping onto building codes. Building codes are interpreted using an object-based representation and then encoded into object-based rules using Express language. A geometry engine and semantic interpretation are used in the DesignCheck system to support design performance verification.

The system allows for checking designs at the different stages – sketch design, detailed design and documentation. It enables the checking of building models against individual clauses within a building code, or alternatively, checking individual object type or group of objects rather than the entire building model. Once the checking is completed, the interactive reporting interface offers a variety of viewing options and enables the user to input the required specifications of objects. The first code to be implemented is the disabled access code.

Keywords: design check, building codes, IFC, EDM

1 INTRODUCTION

Legislation requires the construction industry to check building designs for compliance against numerous building codes. This task is complex and failure to correctly assess designs for compliance can result in high, long-term costs. For example, in a large scale housing project in south London, the wheelchair ramps were found to be too steep and narrow and cost £800,000 in construction and design changes (Building, 2003). To enable designers to identify potential problems earlier, an automated code checking software tool is needed by the construction industry.

The study of code compliance checking has had a long history of development (Gero, 1982; Rosenman et al, 1986; Balachandran et al, 1991; Fenves et al, 1995; Drogemuller et al, 2000; Woodbury et al, 2000; Maissa et al, 2002; Ding et al, 2004). However, there are fewer applications for use in the construction industry. Barriers to more widespread use in industry lie in the lack of common models to integrate building codes with various application environments, object-based representations of building codes to support sophisticated computation and reasoning and support for use of design standards during the design process (Fenves et al, 1995).

Industry Foundation Classes (IFCs) provide a common standard for data interoperability and have been used as a common model in architecture, engineering and construction domains. Major CAD vendors have provided interfaces to IFCs, which make it easier to integrate CAD systems with external analysis tools. Express Data Manager™ (EDM) is a software integration platform that supports interoperability of models defined by IFCs. It provides object-based rule bases and is therefore an ideal platform for encoding building codes and linking them with building models.

The e-PlanCheck system (Solihin, 2004), developed in Singapore, uses the IFC model and EDM. It provides code compliance assessment and acts as an internet-based application or standalone application. However, in e-PlanCheck, support for use of design standards at various stages of design during the design process is not provided. Solibri Model Checker (Solibri, Inc.), developed in Finland, uses the IFC model and focuses on 'design-spell-check'. However, Solibri Model Checker is restricted in its application to code compliance checking due to a restricted range of objects and parameters for encoding building codes and domain knowledge.

This paper presents an advanced automated code checking system – DesignCheck, which was developed by a research team of CRC for Construction Innovation and currently on trial by the construction industry in Australia. The DesignCheck system develops an object-based rule system using EDM for encoding building codes. It defines a DesignCheck internal model based on IFCs for modelling extended design information. The advantages in the DesignCheck system beyond existing tools provide an automated code checking process, flexibility by allowing a design to be checked by selected clauses or object types and support for checking various stages of design during the design process, such as at the early stage of design, detailed stage of design and specification stage of design. The DesignCheck system is targeted more broadly. It is not only for use by certifiers but also by architects and designers.

2 BUILDING INFORMATION MODELLING AND DESIGNCHECK INTERNAL MODEL

The process of automating code checking requires an adequate building model to begin with. Information currently provided within the object-based CAD model and the

IFC model is inadequate for many building code requirements. This section illustrates the method of constructing better building models in object-based CAD systems using IFCs and the development of a DesignCheck internal model.

2.1 OBJECT-BASED CAD MODEL AND IFC MODEL

ArchiCAD supports building information modelling and distinguishes itself as an information-rich architectural CAD tool rather than a drafting tool. Therefore, the DesignCheck system uses ArchiCAD as the exemplar object-based CAD system. Figure 1 shows a 3D object-based model produced in ArchiCAD by Woods Bagot.

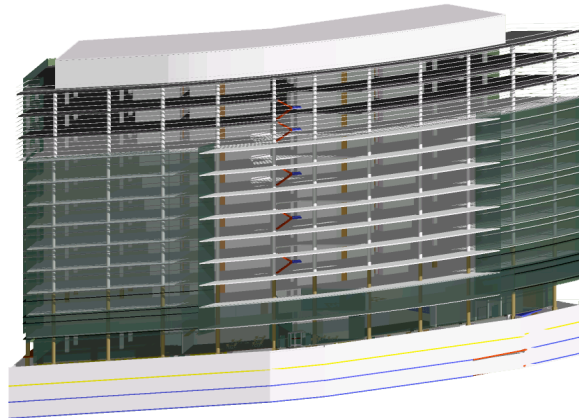


Figure 1. A 3D building model produced in ArchiCAD by Woods Bagot.

Customising GDL object properties and IFCTreeView are two existing approaches available in ArchiCAD 9 that allow extended design information associated with building codes to be inputted and modelled. IFCTreeView lists the element mappings between the CAD model and the IFC model and displays the IFC attributes and property sets of selected objects. Designers can select an element and then define the attributes and properties associated with building codes in the IFCTreeView, Figure 2. IFCTreeView is particularly useful since it allows designers to easily specify additional properties compatible with the IFC model.

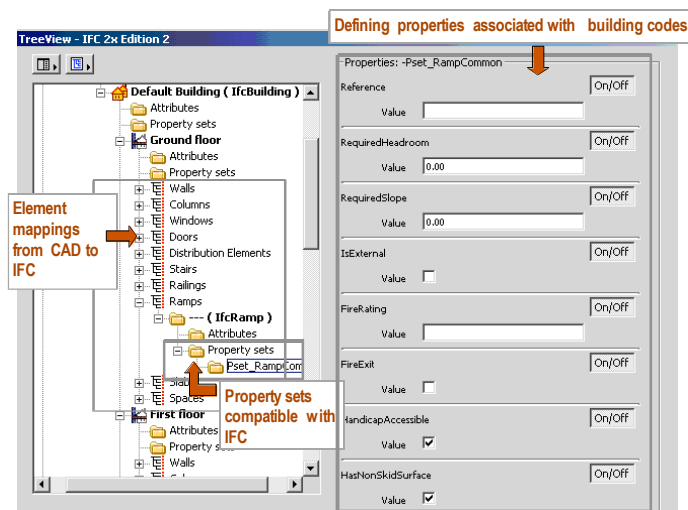


Figure 2. IFCTreeView allows users to define extended properties required by building codes.

Elements, properties and relationships of elements are the distinguishing features in the IFC model. Rich relationships of elements enable meanings between elements to be identified. For example, IfcRelSpaceBoundary provides the bound relationship between a Space and a building element such as a Door. This enables the spatial relationship between an entrance Space and an exterior Door to be identified in order to check an accessible entrance for disabled people.

Examples of the relationship mappings, supported by ArchiCAD and mostly needed by the code compliance checking, are listed as follows:

- IfcRelAggregates – aggregate relationship of a building and storeys, a storey and spaces, etc.
- IfcRelSpaceBoundary – bounds relationship of a space and a building element such as a wall, a door.
- IfcRelContainedInSpatialStructure – containment relationship of a space and objects contained in the space such as toilet objects.
- IfcRelDefinesByProperties – relationship of property sets and objects

However, the existing IFCTreeView in ArchiCAD is restricted in customising enriched element and relationship mappings onto the IFC model. The existing element mappings cover Building, Building Storey, Space, Wall, Door, Stair, etc., but mappings onto Stair Flight and Ramp Flight are not available. Table 1 lists the necessary element mappings required by Australian Standard 1428.1 (AS1428.1) while Table 2 lists a summary of the relationship mappings.

Table 1. Element mappings between AS1428.1, the CAD model and the IFC model, as supported by ArchiCAD 9. Note: element mappings in blue are not yet implemented by ArchiCAD 9 – IFC2x2 Exporting.

| AS 1428.1 ELEMENT | ArchiCAD 9 ELEMENT | IFC2x2 ELEMENT |
|--|---|--------------------------------------|
| Building | Building | IfcBuilding |
| Storey | Storey | IfcBuildingStorey |
| Space, Circulation Space | Zone | IfcSpace |
| Wall | Wall | IfcWallStandardCase |
| Window | Window | IfcWindow |
| Door | Door | IfcDoor |
| Column | Column | IfcColumn |
| Floor | Slab | IfcSlab |
| Stair | Stair | IfcStair |
| | Library object - subtype - Stair | IfcStairFlight Expected (N/A) |
| Ramp | Library object - subtype - Ramp | IfcRamp |
| | | IfcRampFlight Expected (N/A) |
| Walkway | Library object - subtype – Slab | IfcSlab / IfcBuildingElementProxy |
| Landing | Library object - subtype – Slab | IfcSlab / IfcBuildingElementProxy |
| Handrail, Balustrade, Grab Rail | Library object - subtype - Railing | IfcRailing |
| Washbasin, Bidet, Shower, Sink, Toilet, Urinal | GDL object - subtype – Flow Terminal (Basin, Bath, Bidet, Shower, Sink, Toilet, Urinal) | IfcFlowTerminal |
| | | IfcSanitaryTerminal Expected (N/A) |
| Weelchair Seating, fixed Seating | Library object - subtype - Seating | IfcFurnishingElement |
| Light, Outlet, Switch | Library object - subtype – Electrical Elements (Light, Outlet, Switch) | IfcElectricalElement |

Table 2. A summary of the relationship mappings required by the code compliance checking for AS1428.1.

| RELATIONSHIP REQUIRED BY AS1428.1 | IFC 2x2 SUPPORT |
|---|--|
| Bounds_relation (Space, Door) | IfcRelSpaceBoundary |
| Bounds_relation (Space, Wall) | IfcRelSpaceBoundary |
| Contains_relation (Space, Object) | IfcRelContainedInSpatialStructure |
| Contains_relation (Space, Column) | IfcRelContainedInSpatialStructure IfcRelSpaceBoundary |
| Adjacent_relation (Space, Space, Door) | IfcRelSpaceBoundary |
| Wall_door_relation (Wall, OpeningElement, Door) | IfcRelVoidsElements IfcRelFillsElements |
| Element_covering_relation (Element, Covering) | IfcRelCoversBldgElements |
| Space_covering_relation (Space, Covering) | IfcRelCoversBldgElements IfcRelSpaceBoundary |
| Decomposes/Assemble_relation (Ramp, Landing, Handrails) | Not provided in the existing ArchiCAD-IFC exporting |
| Connectivity_relation (Ramp, Doorway, Space) | Not provided in the existing ArchiCAD-IFC exporting |

When the required relationship mappings are not available, the DesignCheck system derives the semantics from the geometry of elements. However, to fundamentally support code compliance checking, it requires the object-based CAD systems to deliver adequate semantics of elements and map a number of elements and relationships of elements provided by the IFC model to enable to infer high level building performance.

2.2 DESIGN CHECK INTERNAL MODEL

The IFC model provides a predefined standard. It covers a large scope of interoperability including architecture, structure, fire engineering and building service domains, hence is complicated. For a domain-specific application such as code compliance checking, detailed application-specific information may be missing in the IFC model.

An internal model is developed for DesignCheck to solve this problem. The DesignCheck internal model extends the IFC model to cover enriched application-specific information, i.e. the information required by building codes.

For example, the Building Code Australia Part D Access and Egress (BCA D3) clauses require checking building classes to determine specific access requirements for disabled people. A new type definition mapping onto building classes is defined in the DesignCheck internal model as shown below:

```

TYPE Building_Type_Enum
    CLASS_1A_SINGLE_DWELLING
    CLASS_1B_BOARDING_HOUSE
    CLASS_2_SOLE_OCCUPANCY_UNITS
    CLASS_3_RESIDENTIAL_BUILDING
    CLASS_4_A_DWELLING_IN_A_BUILDING
    CLASS_5_OFFICE_BUILDING
    CLASS_6_SHOP_BUILDING
    CLASS_7A_CARPARK
    CLASS_7B_STORAGE
    CLASS_8_LABORATORY
    CLASS_9A_PUBLIC_HEALTH_CARE_BUILDING
    
```

```
CLASS_9B_PUBLIC_ASSEMBLY_BUILDING  
CLASS_9C_PUBLIC_AGED_CARE_BUILDING  
CLASS_10A_NON_HABITABLE_BUILDING  
CLASS_10B_NON_HABITABLE_STRUCTURE
```

The entities in the DesignCheck internal model are structured to include new attributes mapping onto building codes and the properties transformed from IFC property sets that are associated with building codes. An example of the Door entity with new attributes and extended properties in the DesignCheck internal model is shown below:

```
ENTITY DOOR_CRC  
  OverallHeight  
  OverallWidth  
  Door_Type  
  Door_Style  
  Is_External  
  SelfClosing  
  FireExit  
  AS1428_Compliance  
  DC_Level_Handle_Clearance  
  DC_Door_Recess_Depth  
  DC_Surface_Mounted  
  DC_Door_Handle_Operation
```

The DesignCheck internal model focuses on defining application-specific information such as defining comprehensive design information associated with building codes to facilitate automated code checking. A future development to the DesignCheck internal model lies in integrating it with a semantic model.

2.3 MAPPINGS BETWEEN IFC MODEL AND DESIGNCHECK INTERNAL MODEL

The building model produced in object-based CAD systems is exported to the IFC model and then mapped onto the DesignCheck internal model for checking against building codes. A mapping schema is required to facilitate automated translation from the IFC model to the internal model, Figure 3.

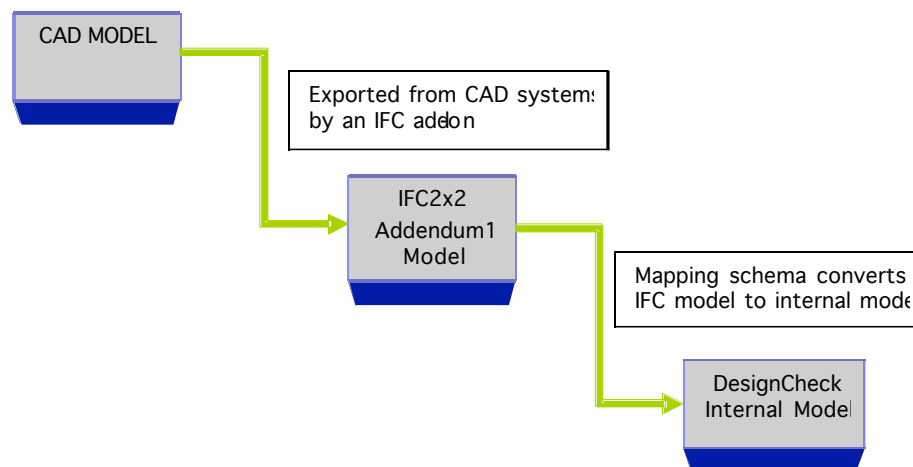


Figure 3. A process of mapping from the CAD model to the IFC2x2 model and then to the DesignCheck internal model.

The mapping schema is implemented using the ExpressX language. ExpressX contains mapping specific functions which allow users to efficiently convert models. The mapping schema for the DesignCheck system is well structured and can be readily modified and extended in future.

3 OBJECT-BASED INTERPRETATION FOR BUILDING CODES

3.1 BUILDING CODE INTERPRETATION

Building codes comprise specific definitions of terms and imply domain-specific knowledge. Researchers have been attempting to develop interpretation of building codes into computational representations in order to facilitate automated compliance checking. A general approach to the interpretation of building codes considered by this research is summarised as follows.

- Develop an object-based interpretation for building codes to facilitate their integration with object-based applications.
- Incorporate specific definitions of items in the object-based interpretation of building codes and develop strategies for encoding.
- Develop the building code interpretation for use at different stages of design.
- Consult with experts such as standards writing organisations, architects and certifiers.
- Enable the building code interpretation from different resources to be consistent.

3.2 OBJECT-BASED INTERPRETATION

The DesignCheck system represents building codes using an object-based interpretation and encodes them into the EDM rule bases. A process of encoding a building code clause to an object-based interpretation and then to the EDM rules is illustrated in Figure 4.

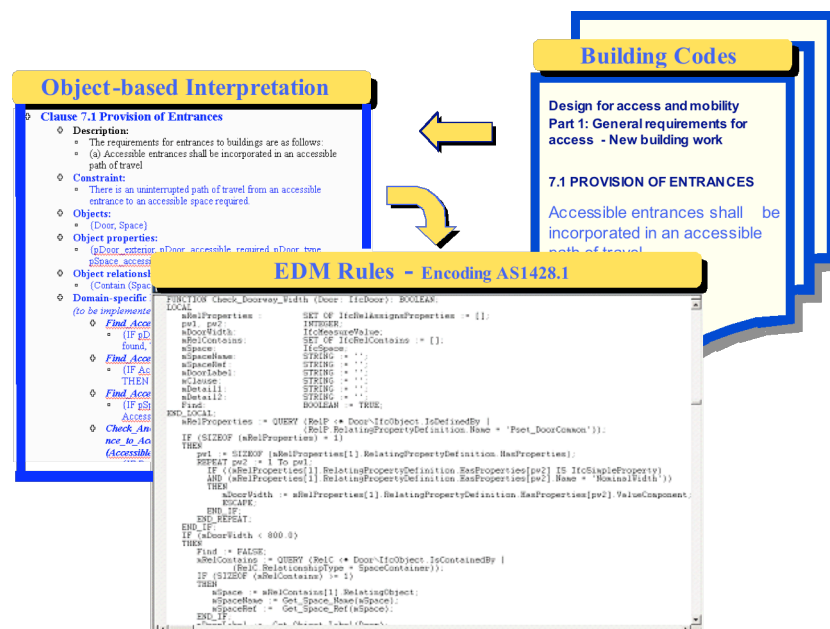


Figure 4. An illustration of the process of encoding a building code clause to an object-based interpretation and then to the EDM rules.

The object-based interpretation presents building codes with elements, properties and relationships and domain-specific knowledge embedded in building codes with

functions and procedures. Figure 5 illustrates an example of the object-based interpretation of a subclause in AS1428.1. The subclause, from Clause 7.1 Provision of Entrances, is described as:

Clause 7.1 (a)

Accessible entrances shall be incorporated in an accessible path of travel.

Compliance checking against this clause requires an object-based interpretation on 'accessible entrances' and 'an accessible path of travel'. The DesignCheck system extracts required elements, properties and relationships from this clause, e.g. 'Space', 'Door', 'Door_external', 'Door_type', 'Door_width', 'Space_accessible', 'Space_identification', 'Space_area', 'Containment relationship between space and door' and 'Adjacency relationship between two spaces'. 'Accessible entrances' are determined by 'Door', 'Door_external', 'Door_type', 'Door_width' and 'Containment relationship between space and door'.

CLAUSE 7: DOORWAYS, DOORS AND CIRCULATION SPACE AT DOORWAYS

Clause 7.1 Provision of Entrances

Description:
The requirements for entrances to buildings are as follows:
(a) Accessible entrances shall be incorporated in an accessible path of travel.

Performance Requirements:
There is an uninterrupted path of travel from an accessible entrance to an accessible space required.

Objects:
{Space, Door}

Object Properties:
{Door_external, Door_accessible, Door_type, Door_width, Space_accessible, Space_identification, Space_area}

Object Relationship:
{Contain (Space, Door)}; {Adjacent (Space, Space)}

Domain-specific knowledge for Interpretation:
(to be implemented with functions, procedures, etc.)

AssessibleExteriorDoor (Doors)
{IF Door_external and Door_accessible are found, THEN return AssessibleExteriorDoors}

AccessibleEntranceSpace (AccessibleExteriorDoors)
{IF AssessibleExteriorDoors are contained by Spaces, THEN return AccessibleEntranceSpaces}

AccessibleSpaceRequired (Spaces)
{IF Space_accessible is found, THEN return AccessibleSpacesRequired}

A_Path_from_AccessibleEntranceSpace_to_AccessibleSpaceRequired (Spaces, Doors)
{IF Spaces and Doors are located in the path from AccessibleEntranceSpace to AccessibleSpaceRequired, THEN return a set of the Spaces and a set of the Doors}

Criteria_for_anUninterruptedPath
{IF Spaces and Doors located in the path satisfy the requirement of Door_width, Door_type, Space_area, etc. THEN return TRUE}

Figure 5. An example of an object-based interpretation for a building code clause.

4 OBJECT-BASED RULE BASE

4.1 RULE BASE STRUCTURE

Different interests and concerns by designers are clarified in the DesignCheck system. This provides a basis for structuring rule bases in EDM for use at different stages of design.

At the early stage of design, designers are concerned with accessible paths to/within a proposed building, circulation space at doorway, circulation space at disabled toilet, etc. Associated clauses are mainly from BCA D3 and partly from AS1428.1. A rule base for the early stage of design is constructed in EDM, which involves functions and procedures that interpret performances at the early stage of design. Semantic interpretations for verification of high level performances are required.

At the detailed stage of design, designers may be concerned with door widths, handrail heights, etc. Associated clauses are mainly from AS1428.1. A rule base for the detailed stage of design is constructed in EDM, which involves functions and procedures that interpret performances such as door widths, handrail heights, etc. Semantic interpretations derived from geometrical descriptions of objects are required.

At the specification stage of design, designers are interested in specification requirements for certain objects such as floor surfaces, handrail materials, signs, etc. Associated clauses are from both AS1428.1 and BCA D3. A rule base for the specification stage of design is constructed in EDM to encode specification requirements of objects for designers to check.

The rule base structure for the DesignCheck system is presented in Figure 6. It consists of the early stage design rule base, detailed stage design rule base and specification stage design rule base, which are developed using EDM Rule Schema. Each EDM Rule Schema comprises a number of global rules, which enable building models to be validated against the selected rules. The check results are stored in the results model in EDM. The intermediate results model is used to store interim data from validation of rules when necessary. For example, the data from validation of a rule for an early stage of design may be of use by a rule for a detailed stage of design.

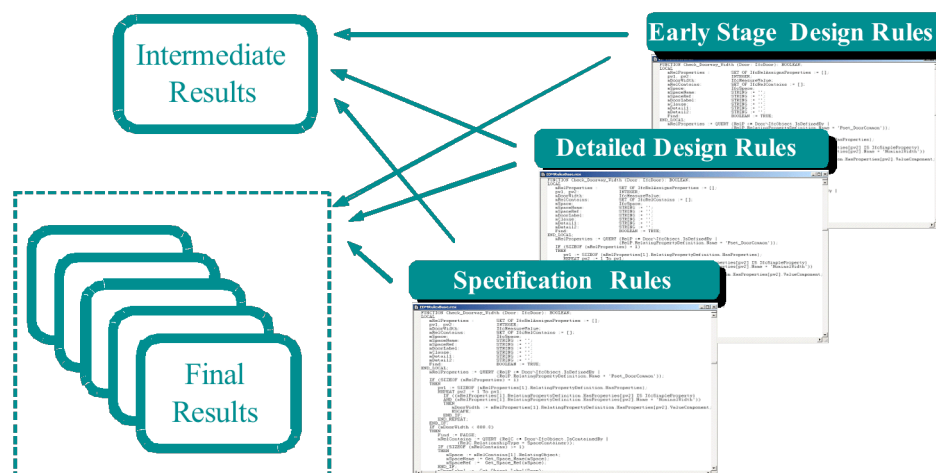


Figure 6. The rule base structure in EDM for the DesignCheck system.

4.2 ENCODING OF EDM RULES

The rules of encoding building codes are built in the Express language. This section illustrates the strategy and methodology of encoding building codes with an example, Figure 7.

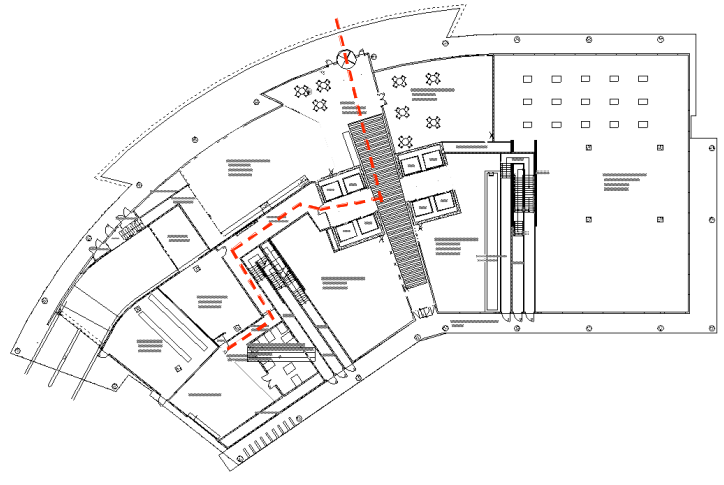


Figure 7. An example of checking an accessible entrance to be incorporated in an accessible path of travel.

Clause 7.1 in AS1428.1 refers to the requirements for entrances to buildings, where Clause 7.1 (a) says that

Clause 7.1 Provision of Entrances

(a) Accessible entrances shall be incorporated in an accessible path of travel.

The DesignCheck system allows designers to check a design against the entire clause, or alternatively by object types of interest, e.g.

Checking an accessible path between disabled toilets and accessible entrances;
Checking an accessible path between lifts and accessible entrances;
Checking an accessible path between public spaces (e.g. conference room) and accessible entrances;

Strategies for this clause lie in finding all Space and Door objects on the path and checking for satisfaction of accessibility. Verification for the containment relationship between Space and Door and the adjacent relationship of accessible Spaces are two critical parts for determining a path. A graph developed for inferring adjacent accessible spaces is presented in Figure 9 (Boulaire, 2005) and described as follows.

The nodes in the graph in Figure 8 represent spaces, the line between two nodes indicates that the spaces are adjacent and accessible, and the dashed line indicates that the spaces are adjacent but not accessible by disabled people. For example, Office 1 and Office 2 in Figure 9 are adjacent but fail to comply with a subclause 'opening at doorways', hence, they are not adjacent accessible spaces and there is no accessible path between them.

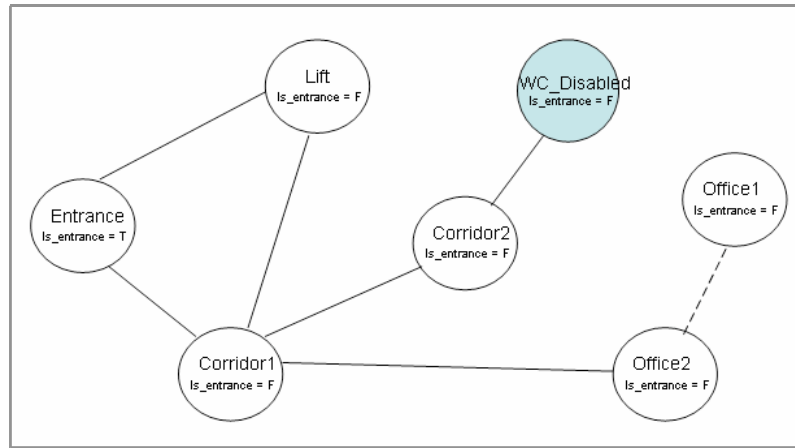


Figure 8. An illustration of the adjacency graph, where F means false and T means true.

If searching for an accessible path between disabled toilets and accessible entrances as shown in Figure 9, the algorithm starts with finding a node of interest, e.g. WC_Disabled and then identifies adjacent spaces and checks for accessibility. Since Corridor2 is the only accessible adjacent space to WC_Disabled, it searches from Corridor2 and finds Corridor1. From Corridor1, it finds four accessible adjacent spaces and one of them is identified as an accessible entrance. An accessible path is then determined between WC_Disabled and Entrance.

5 DESIGNCHECK SYSTEM

5.1 SYSTEM ARCHITECTURE

The architecture of the DesignCheck system is illustrated in Figure 9. It consists of three main components: main user interface, EDM database and the report system.

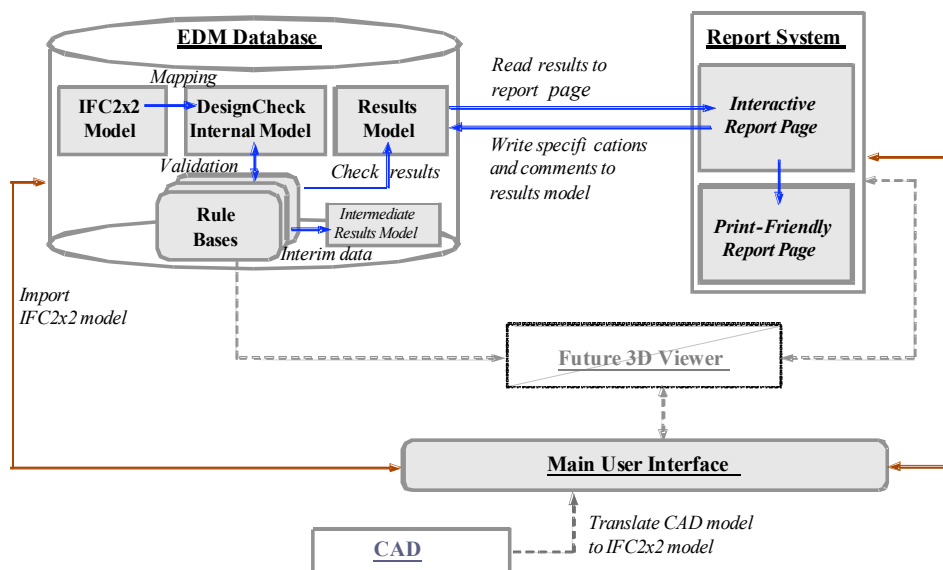


Figure 9. Architecture of the DesignCheck system.

The main user interface allows designers to: monitor the checking of designs at various stages, select a specific clause or object type, view check results, and input specifications and comments. The DesignCheck system runs as a standalone software. The building model created in object-oriented CAD systems is exported as an IFC2x2 model and then imported to the DesignCheck system for compliance checking. If it is required, a direct interface to object-oriented CAD systems could be developed in future.

The EDM database is the core component of the DesignCheck system. The EDM database has been developed to contain building models, rules bases and the check results. Two building model schemata are defined in the EDM database: the IFC2x2 model schema and the DesignCheck internal model schema. The IFC2x2 model schema allows the building model to be imported to the EDM database in IFC2x2 format. The DesignCheck internal model schema enables application-specific information, i.e. the information required by building codes. A mapping schema is developed to allow the IFC2x2 model to be mapped onto the DesignCheck internal model automatically. The DesignCheck internal model is validated against rules in the rule bases. The rules encode object-based interpretations and performance requirements from building codes. The results model is defined in the EDM database to store the check results.

The report system has a direct interface to the EDM database. It reads the check results from and writes the specifications/comments to the results model in the EDM database. The report system provides both an interactive report page and a print-friendly report page. Once checking is completed an interactive report page appears to the user, which offers a variety of viewing options and enables the user to view results by 'All', 'Compliance', 'Non-compliance', 'Specification required', and input the required specifications of objects and comments. The interactive report page links to a print-friendly report page that allows designers to list all details in the report and print it out.

A 3D model viewer, shown in Figure 10 by dashed lines, will be integrated with the DesignCheck system in future. It will provide a 3D visualisation of the building model and allows problem elements to be highlighted.

5.2 SYSTEM IMPLEMENTATION

The implementation of the DesignCheck system is illustrated in Figure 10. The main user interface is a Java application. It allows users to monitor the information flow commencing from importing building models to reporting check results. The interactive report page and print-friendly report page are implemented in Java and html.

The main user interface allows users to select a building code for checking. The following two building codes are available for users to choose in the current implementation of the DesignCheck system:

1. Australian Standard
Design for access and mobility
Part 1: General requirements for access – new building work (i.e. AS1428.1);
2. Building Code Australia
New draft access code for buildings (currently released for public comments)
Part D – Access and egress (D3)

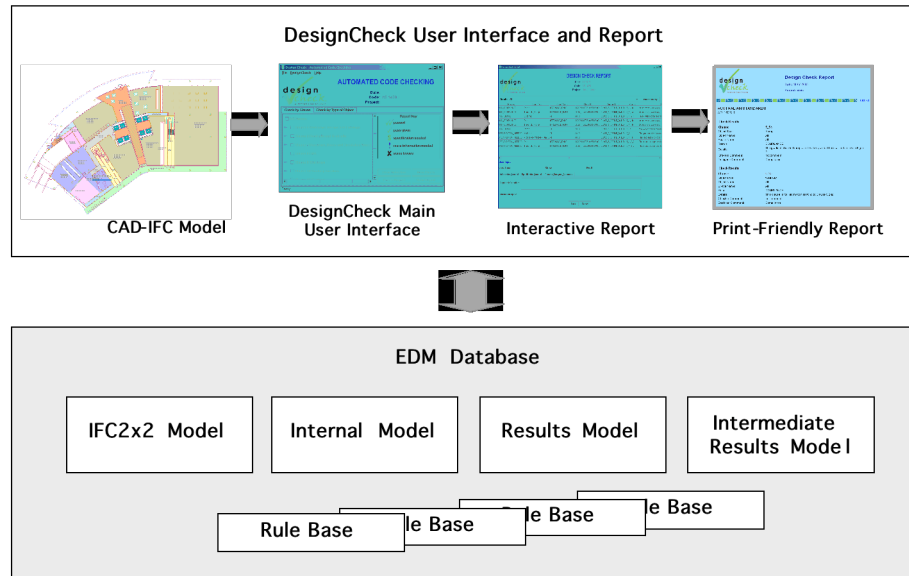


Figure 10. Implementation of the DesignCheck system.

The option of checking design by clauses provides a selection tree of all clauses and subclauses, Figure 11. Selecting a main clause triggers EDM to validate a rule schema corresponding to the selected clause, whereas, the selection of a set of subclauses, triggers EDM to validate individual global rules within a rule schema.

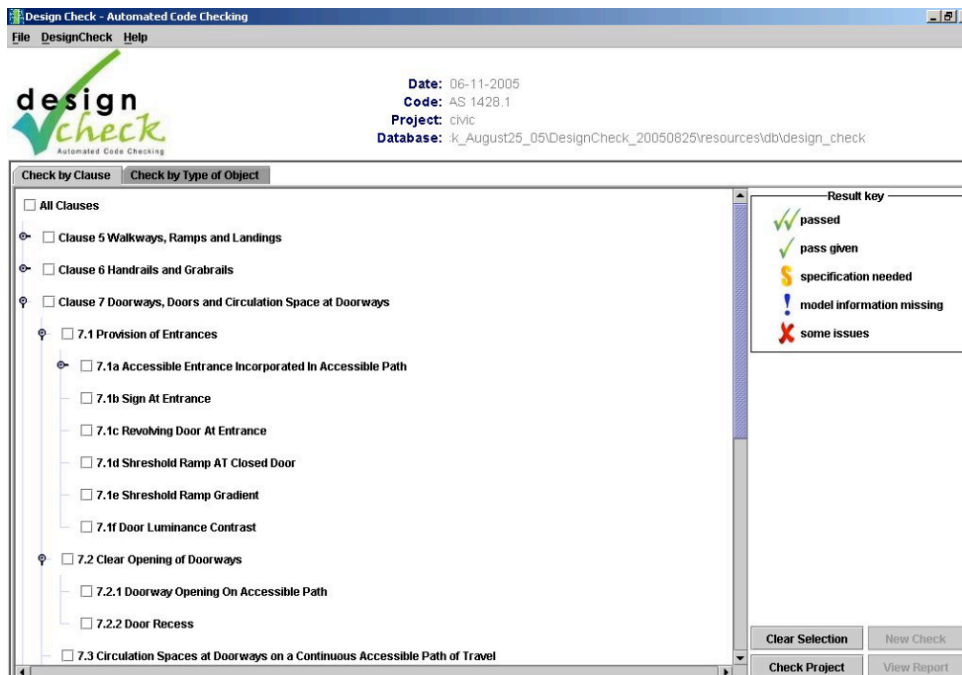


Figure 11. Checking design by selected clauses.

The option of checking design by object types provides a selection tree of object types, Figure 12. Users are allowed to select an object type of interest for checking at the early stage of design, detailed stage of design or specification stage of design.

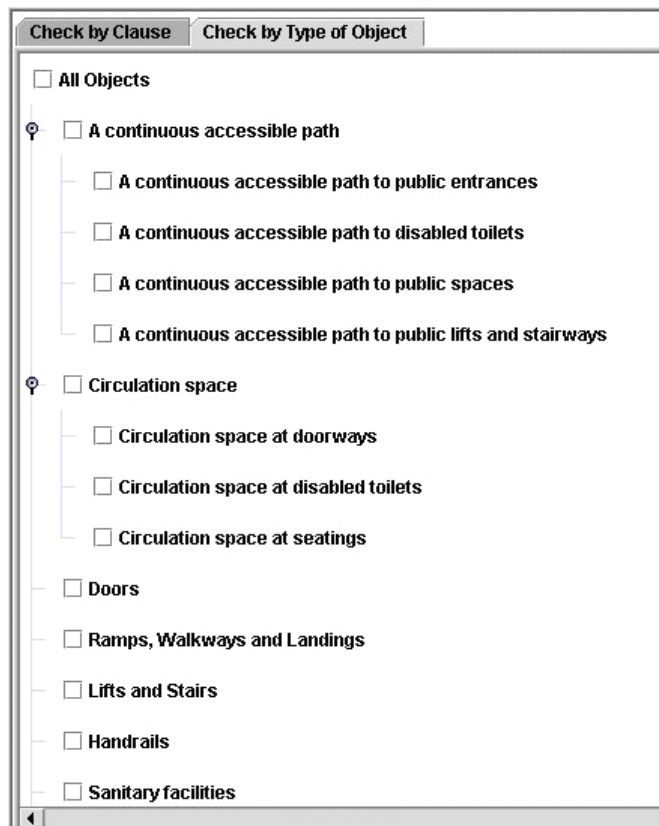


Figure 12. Checking design by selected object types.

The results of rule validation are stored in the results model in the EDM database. Once validation is completed, graphic display of the results is provided for each clause or object type selected. The Report Key panel shows details of meanings of the result icon, Figure 13.

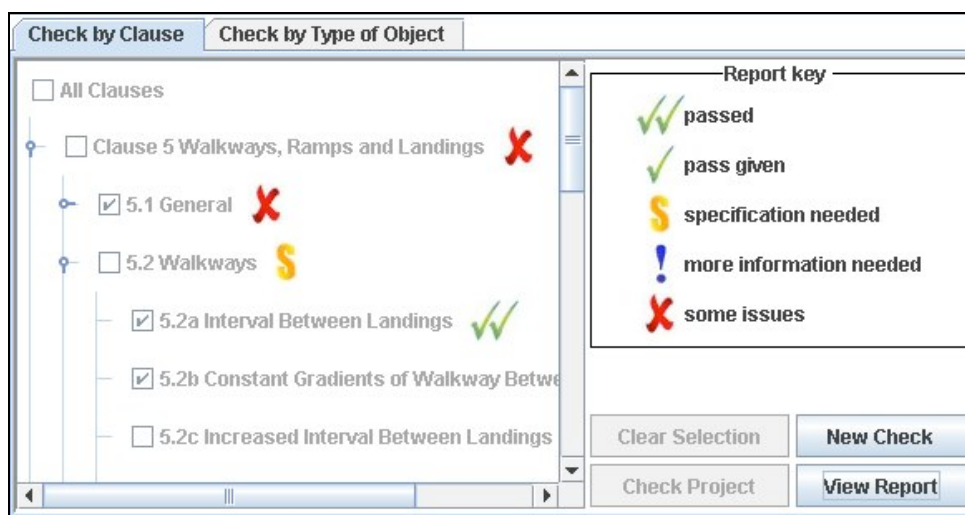


Figure 13. The graphic display of the check results.

The report page is designed as an interactive user interface, so that users can select a result type that they intend to view and update the result model by adding specifications of objects and comments. The interactive report page consists of four major areas: (1) the top panel to display project information; (2) the selection panel to

provide option of viewing results; (3) the table panel to display detailed check results including object name, object type, space name where the object is located, failed feature of the object, clause name and check result; and (4) the bottom tabbed panel that allows users to input specifications and comments, Figure 14.

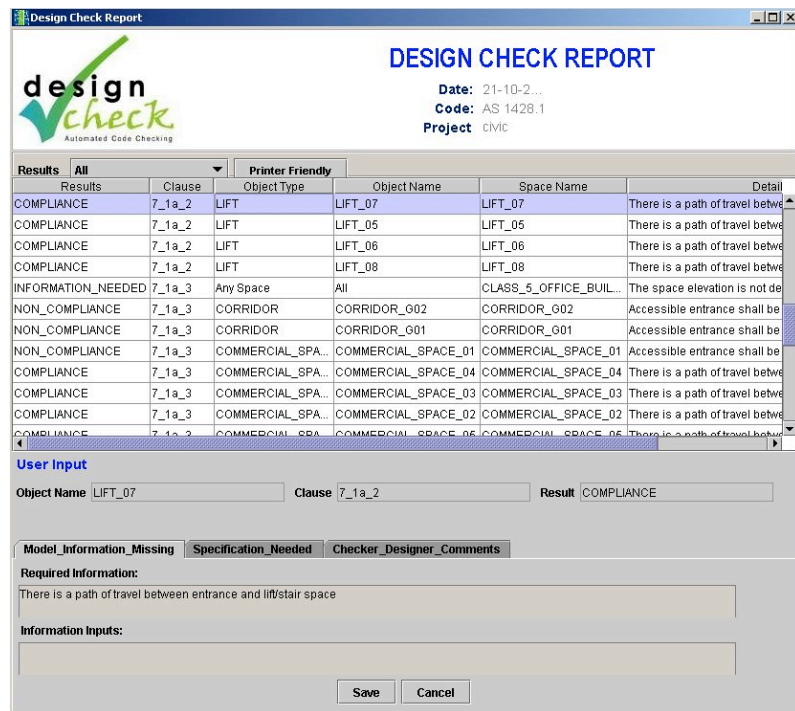


Figure 14. An example of the interactive report page.

A print-friendly version of the report is linked to the interactive report page. It opens Microsoft Internet Explorer, showing a formatted print-friendly report ready for previewing and printing, Figure 15. This report page can also be saved into archives for later backup and further reference or comparison.



Figure 15. An example of the print-friendly report page.

6 CONCLUSION AND FUTURE DEVELOPMENT

The development of the DesignCheck system uses an efficient platform and provides functionalities towards industry needs. As an advanced software tool, the DesignCheck system will reduce the risk of non-compliance with its associated rectification costs and significantly improve the efficiency in the building code checking process. Direct benefits to architects, designers, building consultants and engineers can be gained from DesignCheck. These benefits include:

- Automating the design checking process for compliance with building codes;
- Providing more reliable assessment with less errors;
- The ability to interrogate 3D object-based CAD systems;
- Allowing the checking at various stages - sketch design, detailed design and specification;
- Allowing the checking of a design by selected building code clauses;
- Allowing the checking of a design by selected building object type;
- Providing a friendly and interactive reporting system;
- The ability to check 'on-the-fly' the compliance of the design to building codes, and to reduce the lead-time of a design process.

DesignCheck is currently being tested by private and public design organisations for validation and feedback.

Future development of the DesignCheck system relating to the interest areas in both research and practice includes: the development of a consistent manner for building code interpretation such as using decision tables, the development of semantic models and expert knowledge, and system improvement, including the development of structured specification to allow users to input specification easily and the integration with a 3D model viewer.

Collaboration with CAD vendors is required to enable the CAD-IFC interface to be improved to support automated code checking application.

ACKNOWLEDGEMENTS

The authors acknowledge Moshe Gilovitz (Building Commission) for his valuable advice and coordination of input from others, Cheryl McNamara, Kevin McDonald, Dr. John Mashford and Fanny Boulaire of CSIRO and Julie Jupp, Wei Peng, JiSoo Yoon and Nicholas Preema of University of Sydney for significant contributions to the DesignCheck system.

REFERENCES

- Balachandran M., Rosenman M. A. and Gero J. S. (1991) A Knowledge-based Approach to the Automatic Verification of Designs from CAD Databases, in Gero J. S. (ed.), *Artificial Intelligence in Design '91*, Butterworth-Heinemann, Oxford, pp. 757-781.
- Boulaire B. (2005) Code checking phase-2 internal report, CRC for Construction Innovation.
- Building (2003) http://www.building.co.uk/magazine/html/2003_issue_20.html
- Ding L., Drogemuller R., Jupp J., Rosenman M. A. and Gero J. S. (2004) Automated code checking, *CRC CI International Conference 2004*, Gold Coast.
- Drogemuller R., Woodbury R. and Crawford J. (2000) Extracting representation from structured text: initial steps, w78-2000-302.

- Eastman C. M. (1999) *Building Product Models: Computer Environments Supporting Design and Construction*, CRC Press.
- EPM Technology: (2002) *EDMAssist4.5*, vols 1-5, Norway: EPM Technology.
- Fenves S. J., Garrett J. H., Kiliccote H. Law, K. H. and Reed K. A. (1995) Computer representations of design standards and building codes: U.S. perspective, *The International Journal of Construction Information Technology*, Vol 3, No 1, pp. 13-34.
- Gero J. S. (1982) A self-checking database for the Australian Building Code, *CAD82*, Butterworths, Guildford, pp. 119-125.
- Graphisoft (2001) *ArchiCAD IFC Reference Guide, Version 1.0*, Graphisoft.
- IAI (2004) *IFC 2x2 Edition 2 Addendum 1*, IAI.
- Maissa S., Frachet J. P., Lombardo J. C., Bourdeau M. and Soubra S. (2002) Regulation Checking in a Virtual Building, *CIB w78 conference 2002*.
- Rosenman M. A., Gero J. S. and Oxman R. (1986) An expert system for design codes and design rules, in Sriram D. and Adey R. (eds), *Applications of Artificial Intelligence to Engineering Problems*, Springer-Verlag, Berlin, pp. 745-758.
- Solibri, Inc. <http://www.solibri.com>
- Solihin W. (2004) Achieving Automated Code Checking with Ease, a white paper, novaCITYNETS pte ltd, Singapore.
- Stouffs R. and Krishnamurti R. (2001) On the Road to Standardization, in *Proceedings of CAAD Futures 2001*, Netherlands, 75-88.
- Woodbury R., Burrow A., Drogemuller R. and Datta S. (2000) Code Checking by Representation Comparison, *CAADRIA: Proceedings of the 5th Conference on Computer Aided Architectural Design Reseach in Asia*, pp. 235-244, CASA, Singapore.