ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

AUTOMATED CODE COMPLIANCE CHECKING: A SYSTEM FOR CHECKING FIRE CODES

M.Sc. THESIS

Özgün BALABAN

Department of Informatics

Architectural Design Computing Programme

JULY 2012

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

AUTOMATED CODE COMPLIANCE CHECKING: A SYSTEM FOR CHECKING FIRE CODES

M.Sc. THESIS

Özgün BALABAN
523091025

Department of Informatics

Architectural Design Computing Programme

Thesis Advisors: Prof. Dr. Gülen ÇAĞDAŞ
Dr. Elif Sezen YAĞMUR KİLİMCİ

JULY 2012

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

YÖNETMELİKLERİN OTOMATİK UYGUNLUK KONTROLÜ: YANGIN YÖNETMELİKLERİ KONTROLÜ İÇİN BİR SİSTEM

YÜKSEK LİSANS TEZİ

Özgün BALABAN
523091025

Bilişim Anabilim Dalı

Mimari Tasarımda Bilişim Programı

Tez Danışmanı: Prof. Dr. Gülen ÇAĞDAŞ
Öğr. Gör. Dr. Elif Sezen YAĞMUR KİLİMCİ

TEMMUZ 2012

Özgün Balaban, a M.Sc. student of ITU Graduate School of Science, Engineering and Technology student ID 523091025, successfully defended the thesis entitled “Automated Code Compliance Checking: A System For Checking Fire Codes”, which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Gülen ÇAĞDAŞ** İstanbul Technical University	....
Co-advisor :	**Dr. Elif Sezen Yağmur KİLİMCİ** İstanbul Technical University	....
Jury Members :	**Prof. Dr. Salih Ofluoğlu** Mimar Sinan Fine Arts University	....
	Doç. Dr. Mine ÖZKAR Istanbul Technical University	....
	Yrd. Doç. Dr. Hakan YAMAN İstanbul Technical University	....

Date of Submission : 24 July 2012
Date of Defense : 26 July 2012

In the memory of my mother,

FOREWORD

I would like to thank my family for their support during my graduate studies. My advisors Prof. Dr. Gülen Çağdaş and Dr. Elif Sezen Yağmur Kilimci gave great support during my thesis studies, without them it was impossible to build this work.

July 2012 Özgün BALABAN

TABLE OF CONTENTS

ABBREVIATIONS

2D	: Two Dimensional
ACCC	: Automated Code Compliance Checking
ACCCS	: Automated Code Compliance Checking System
AEC	: Architecture, Engineering, Construction
AHUS	: Akershus University Hospital
AI	: Artificial Intelligence
ANSI	: American National Standards Institute
API	: Application Programming Interface
AS	: Australian Standards
BCA	: Building Code Australia
BCA	: Singapore Building Construction Authority
BOMA	: Building Owners and Managers Association
BIM	: Building Information Modeling
CAD	: Computer Aided Design
CCOM	: Code Checking Object Model
CSI	: Construction Specification Institute
CORENET	: Construction and Real Estate Network
CSIRO	: Commonwealth Scientific and Industrial Research Organisation
GIS	: Geographic Information System
GSA	: General Services Administration
EDM	: Express Data Manager
ICC	: International Code Council
IFC	: Industry Foundation Classes
IFD	: International Framework for Dictionaries
IFG	: IFC for GIS
IT	: Information Technology
HITOS	: Tromso University College
HTML	: Hyper Text Markup Language
MCS	: Model Checking Software
MEP	: Mechanical Electrical and Plumbing
OCA	: Office of Chief Architect
PBS	: Public Building Safety
PDF	: Portable Document Format
RDBMS	: Relational Database Management System
RTF	: Rich Text File
SMC	: Solibri Model Checker
XML	: Extensible Markup Language

LIST OF TABLES

LIST OF FIGURES

AUTOMATED CODE COMPLIANCE CHECKING: A SYSTEM FOR CHECKING FIRE CODES

SUMMARY

Architecture today has evolved into its most complex form. There are many criteria such as fire safety, acoustics, sustainability etc… that must be controlled by architects or engineers. To accomplish a standard on these criteria, governments publish different codes. For a building to get a permit and to be constructed it must be checked against those codes and pass this checking. Until today, regulation checks have being done manually by people. This process requires extensive manual work and time and is prone to errors. The automatic compliance checking software is a requirement to avoid costly mistakes or at least minimize it.

Two recent advancements in the field can change the human reliant nature of the rule checking process. The first one is the development of BIM (Building Information Models), which is a digital building model that defines buildings with various parameters. The other one is the development of expert systems that evolved in parallel with the progresses in Artificial Intelligence. Rule checking systems can be regarded as specialized expert systems.

There are ongoing studies to accomplish a fully automated rule compliance system. The most successful one is CORENET, which is in use in Singapore. With the help of the CORENET, in Singapore all code checks are done digitally. Apart from CORENET there are other efforts in Norway, Australia, and USA but they are concentrated on some special issues such as accessibility.

From the early works and the general structure of all previous code checking examples the code checking process is divided into four stages. These are; 1) “Rule Interpretation” where the written rules are translated into computer recognizable forms, 2) “Building Model Preparation” where the design are transferred into digital world via BIM software, 3) “Rule Execution” where the rules are applied to the building models and 4) “Rule Reporting” where the results and the errors (if any) are displayed to the user.

In this thesis, we present our model for automated code compliance checking. Our motivation for this study is that there is no previous effort to develop automated code checking in our country. Our work will be foundation to build a local code check engine. Accomplishing a fully automatized code checking system is a huge undertaking thus we have restricted our area to egress clauses in fire codes. This is because we have limited resources in terms of time, budget, work force etc… But according to the results of our current work, we will continue with a broader spectrum of code.

Our model consists of two interworking modules. First one, translates human written codes into computer recognizable entities. This is done manually by the programmer. Every entity defined in the codes whether it is a spatial entity such as egress route, circulation or an architectural element such as doors, ramps; it is regarded as an object. Objects have parameters like width of a door and methods for extracting the needed information such as distance between two doors.

In second module, rule checking is performed. This module uses IFC to retrieve data needed for rule checking. IFC is a neutral data format, which ensures most of the BIM programs if not all of them, are usable with our model. The retrieved data is stored in a database and this module processes this data.

After rule checking module finishes checking the building model, the results are given to the user. The results can be PASS if all of the codes are satisfied, FAIL if one or more clauses fail and N/A if there is some missing information, which prevents the program to run. If the result is, FAIL then the program reports, which clauses of the code fail, and shows the cause of the problem. The report page will be interactive so that users can check each clause individually. It is possible to export this information in pdf, doc or xml format or to take print outs of the report.
The implementation of the model is done in Java and for database part, MySQL is used.

We discussed on the advantages and drawbacks of our model according to the results of this study. If we find this model as viable, we will continue our study and broaden our scope in the area we cover.

OTOMATİK YÖNETMELİK UYGUNLUK KONTROLÜ: YANGIN YÖNETMELİKLERİ KONTROLÜ İÇİN BİR SİSTEM

ÖZET

Mimarlık, günümüzde daha önceki dönemlerde olduğundan çok daha karmaşık bir hale gelmiştir. Mimarlık, Mühendislik, İnşaat (AEC) firmalarının, tasarım süreci boyunca ve tasarım sonrasında kontrol etmeleri gereken yangın güvenliği, akustik, sürdürülebilirlik gibi birçok kriter vardır. Kamu için inşa edilen tüm binaların yasalara uymaları gerekir ve yasalara uygunluklarını kontrol etmek amacıyla inşa edilecek bina işlevine ve ele alacakları kıstaslara göre birçok farklı yönetmelik yayınlanır. Bir firmanın bir projeyi gerçekleştirebilmek için ruhsat alması gerekir ve ruhsat alımı sürecinde inşa edilen ya da edilecek binanın onlarca yönetmeliğe uygun olup olmadığı kontrol edilir. Bu güne kadar yönetmelik kontrolleri insan uzmanlar tarafından ve elle yapılmaktaydı. Fakat bu işlem çok fazla iş gücü ve zaman gerektirir ve ayrıca hatalara açıktır. 1998 yılında İngiltere'de yapılan bir toplu konut projesinde inşa edilen tekerlekli sandalye rampalarının çok dik ve dar oldukları ortaya çıkmıştır, bu rampaların yeniden düzenlenmesi 800.000 Sterline mal olmuştur. Bu gibi hataların ortaya çıkmasını engellemek ya da en azından azaltmak için yönetmelik ve şartname kontrollerini otomatik hale getiren yazılımlara ihtiyaç duyulmaktadır.

Günümüzde yönetmelik kontrolünün insana dayalı olmasını değiştirebilecek iki gelişme yaşanmaktadır. Bunlardan ilki Yapı Bilgi Sistemi (BIM) adı verilen ve yapıların 3B obje modelleri şeklinde çizildikleri bilgisayar uygulamalarının gelişmesi ve bu sayede çizilen projelerin bilgisayar tarafından yorumlanabilen objeler haline gelmesidir. İkinci gelişme de 20. yüzyılın ikinci yarısında, yapay zekânın bir kolu olarak ortaya çıkan, bir insan uzmanı taklit eden, onun verdiği uzmanlık hizmetine yakın hizmetler veren uzman sistemler geliştirilmeye başlanmasıdır. Uzman sistemlerle, Yapı Bilgi Sistemlerinin gelişip beraber çalıştırılmaları sayesinde yakın bir gelecekte herhangi bir Yapı Bilgi Sistemi uygulamasında çizilen bir yapının istenen bir kural tabanı dâhilinde incelemesi yapılabilecek ve çıkan sonuçlar tasarımcıya geri beslenebilecektir.

Otomatik yönetmelik kontrol sistemi geliştirmek için devam eden çalışmalar vardır. Bunlardan en başarılısı, Singapur'da kullanımda olan CORENET'tir. Singapur'da, CORENET yardımı ile tüm yönetmelik kontrolleri dijital olarak yapılmaktadır. CORENET dışında Norveç, Avustralya ve ABD'de de başka çalışmalar vardır ama bunlar erişilebilirlik gibi bazı özel konular üzerinde yoğunlaşmıştır.

Bu tezde, geliştirilen otomatik yönetmelik kontrolü sunulmuştur. Bu çalışmayı yapmaktaki temel amaç daha önce Türkiye için geliştirilmiş otomatik yönetmelik kontrolü çalışmasının bulunmamasıdır. Bu çalışma sonrasında çıkan modelin, ileride geliştirilebilecek tamamen otomatik yönetmelik kontrolü sistemine temel oluşturması

hedeflenmektedir. Tek bir yönetmelik için bile olsa otomatik yönetmelik kontrolü sistemi oluşturulması işi çok büyük bir iştir. Bu çalışmada böyle bir girişim için yetecek bütçe, zaman, iş gücü vs… bulunmamaktadır. O yüzden bu modelde, yangın yönetmeliklerinden kaçış rotalarının düzenleyen bölüm ele alınmıştır. Bu çalışmada çıkacak modelin daha sonra tüm yönetmeliğe uygulanması planlanmaktadır.

Daha önceki çalışmalardan ve şu ana kadar geliştirilmiş otomatik yönetmelik kontrolü modellerinin yapılarından yola çıkarak, otomatik yönetmelik kontrolü süreci dört etaba ayırlır. Birincisi yönetmeliklerde yazılı olan kuralların yorumlanması ve makine tarafından işlenebilir mantık kuralları haline getirilmesi, ikincisi yapının Bina Enformasyon Sistemi programı içerisinde oluşturulması ve bu sayede yapının yönetmelik kontrolüne uygun hale getirilmesi, üçüncüsü yönetmelik kontrolünün uygulanması ve son olarak da çıkan sonucun tasarımcıya geri bildirilmesidir.

Çalışan bir otomatik yönetmelik kontrolü modeli yapabilmek için bu dört etabın programda olması gerekir. Kural yorumlama etabında, kullanılabilecek iki yöntem vardır. Bunların birincisi, kuralın dönüştürülmesinde programcı kullanılmasıdır. Bu yöntemin eksiği, yönetmelikte olacak her değişimde, programcının programı tekrar düzenlemesi gerekir. Ayrıca bu işlemde kuralların dönüşümünde insan kullanıldığı için insan faktöründen tamamen bağımsız bir sistem geliştirilememiştir ve kuralların çevrilmesinde programcının yapacağı herhangi bir hata sonrasında program hatalı çalışacaktır. İkinci metotta kuralların çevrilmesi işinde Doğal Dil İşleme (NLP) programları kullanılarak yazılı kurallar dijital hale otomatik olarak dönüştürülür. Bu yöntemin eksiği ise, insan dilinin bilgisayar tarafından işlenmesinin çok karmaşık olmasıdır. Bu modelde birinci yöntem uygulanacaktır.

Yapı Modeli oluşturulması etabında, yönetmelik kontrolünün yapılacağı yapı bir Bina Enformasyon Sistemi programı kullanılarak çizilmelidir. Bu sayede yapı hakkındaki gerekli tüm bilgilere program tarafından ulaşılabilir.

Kural kontrolünü yapıldığı etapta ise, dijital hale getirilmiş kuralların kontrolü gerçekleştirilir. Bu etapta önemli olan hiçbir kuralın atlanmamasıdır. Son olarak sonucu bildirildiği etapta kontrolün sonucu ve eğer varsa hatalar kullanıcı bildirilir.

Bu modelde de aynı yaklaşım kullanılmıştır.

Geliştirdiğimiz model iki modülden oluşmaktadır. Birincisi, kullanıcıların yönetmeliklere kolay erişimini sağlamak için yapılmış olan Bilgi modülüdür. Bilgi modülün kullanıcı arayüzden seçtiği maddenin içeriğine kolaylıkla erişebilir. Bu modülün çalışabilmesi için yönetmeliklerin bilgisayar ortamına aktarılmaları gerekmektedir. Bu iş için Genişletilebilir İşaretleme Dili (Extensible Markup Language - XML) kullanılmıştır. Türkiye Yangından Korunma Yönetmeliği XML kullanılarak modele aktarılmıştır ve model bu XML dosyasında istenen maddelerin içeriğini kullanıcıya gösterir.

İkinci modülde, yönetmeliklerin kontrolü gerçekleştirilir. Bu modül yönetmelik kontrolü için gereken veriyi elde etmek için IFC kullanır. IFC, bağımsız bir veri formatıdır. Modelde IFC kullanılması, BIM programlarının hepsi olmasa da çok büyük çoğunluğunun modelle birlikte çalışabileceği anlamına gelir. IFC dosyası sisteme yüklendikten sonra bu IFC dosyasının içerdiği bilgiler mySQL'le hazırlanmış bir veritabanında saklanır. Sonrasın kullanıcı kontrolü yapmak istediği maddeleri seçer. Bu maddelere göre oluşturulmuş algoritmalar sayesinde veritabanındaki bilgiler kontrol edillir. Eğer bu bilgiler, algoritmalarda saklı olan koşulları sağlıyorlarsa o maddeyi GEÇTİ sayılır. BIM'de olan modelden erişilen veri, bir veritabanında saklanır ve yönetmelik kontrolü yapan modül bu veriyi işler.

Yönetmelik kontrolü modülü yapı modelini kontrol ettikten sonra, çıkan sonuçlar kullanıcılara iletilir. Sonuç, yapı eğer tüm yönetmeliklere uygunsa GEÇTİ, eğer bir ya da birkaç yönetmelik maddesine uygun değilse KALDI olur. Eğer programın çalışmasını engelleyecek bir bilgi eksiği varsa sonuç N/A olur. Eğer sonuç KALDI çıkarsa, program binanın uymadığı yönetmelik maddelerini bildirir. Rapor sayfası, kullanıcılar isterlerse herhangi bir yönetmelik maddesine göre kontrol yapabilecekleri şekilde ayarlanmıştır. Çıkan sonuçlar pdf, doc veya xml formatında kaydedilebilir ve ayrıca sonuçların çıktısı da alınabilmektedir.

Programın uygulanmasında Java programlama dili ve mySQL veritabanı kullanılmıştır.

Bu çalışma sonrasında ortaya çıkan model henüz geliştirilme aşamasındadır. Modelde henüz birkaç madde için yönetmelik kontrolü yapılabilmektedir. Ayrıca henüz bir grafik raporlama seçeneği bulunmamaktadır. Ancak üretilen modelin, daha sonraki çalışmalara bir çerçeve oluşturarak ilerideki çalışmaları kolaylaştıracağı düşünülmektedir. Bu yüzden önemli bir girişimdir.

1. **INTRODUCTION**

Architecture today has evolved into its most complex form. Architecture, Engineering, Construction (AEC) companies have many criteria to check such as fire safety, acoustics, sustainability etc… during or after the design process. All buildings that are constructed have to obey the legislation; and for this purpose, there are different codes published depending on the type of the building or the criterion they are referring. Thus, generally for a building project to get its approval, a design firm must satisfy dozens of building codes. If we do not count some unfinished systems that are present in some countries, code checking is a manual process done by experts. This process requires extensive manual work and time and is prone to errors. For instance, in a mass housing project done in England in 1998, the ramps for the wheel-chaired users are found to be too steep and narrow. The required slope and width for the ramps are published in the codes but designers failed to check this information both during the early design phase and after the design check phase. The reconstruction efforts cost GBP 800,000 and it took more than eight months to solve the problem and deliver the project. (Nikkhah, 2003). If automated code compliance checking systems (ACCCS) are developed, it will be possible to minimize if not avoid mistakes like these.

Two advancements are progressing that can change the human reliant nature of rule checking process. The first one is the development of BIM (Building Information Modeling), which is a digital building model that defines buildings with various parameters. The other one is the development of expert systems that evolved parallel with the progresses in the Artificial Intelligence (AI). The rule checking systems can be regarded as specialized expert systems.

1.1 Problem Definition - Motivation

Codes, regulations, and specifications are important assets of construction industry. They draw the bottom line to the performance requirement of a building. Without these requirements performance of buildings would vary greatly, which would ruin the general accessibility of public spaces. Therefore, abiding to the general requirements of a project is vital for construction industry.

The problems of codes are that there are vast volumes of different codes, in these codes, there is constant referencing between clauses, which makes following and locating the needed information from these codes difficult. In addition, most of the time codes are written in an old, legal language, which also makes it difficult for a designer to understand the clauses. Moreover, there are many different subjects of codes and these codes "create a massive volume of semi-structured documents with possible differences in formatting, terminology and context" (Lau, 2004). Because of these reasons, AEC companies treat these regulations as if they are a liability they should obey, instead of an asset they can take advantage.

The act of abiding to the regulations become more complex, if a firm operates in many countries and the rate of multinational firms is big enough to give importance to this issue. A survey on the difficulties of obeying to the regulations of different countries, found this result: "Widely divergent legal restrictions present a growing obstacle to multinational companies … The more prudent multinationals want to comply with data protection laws in an efficient and coordinated manner. It's just not obvious to them how to do it. The laws vary from jurisdiction to jurisdiction, they are constantly changing, and sometimes difficult to understand … a surprisingly large amount of companies are still "solving" this problem by ignoring it" (Raskopf and Bender, 2003). The emergence of automated code compliance checking systems will help in checking designs against the regulations from all over the world. Designers will be able to change the rule schema according to the country they are working to, with "a click of a button".

Complexity of regulations causes AEC firms to devote considerable amount of time in code compliance checking. In a survey about code compliance checking, it is founded that in a project architects typically spend more than 50 hours per discipline and if you think about the four main discipline (structure, architecture, mechanical electrical and plumbing (MEP), contractor), it takes more than 200 hours for code checking (Young, Jones, and Bernstein, 2007). The minimum time spent is 30 hours, and it takes 3-4% of the design time. This amount can rise up to one third of the design time spent for the project. In the same survey, 85% of the architects responded that they are interested in automated code compliance checking. These values prove that there is a need for ACCCS by the architects. With the appearance of ACCCSs, designers will invest more time in their designs instead of spending hours on checking their designs against the regulations.

Checking a project against code compliance in the early design stage is nearly impossible in the traditional manual way, as the regulations are not separated for design stages. Generally, in an AEC firm architects design a structure relying on their expertise and common sense. The final product is tested in the late design stage or even in the documentation stage. In this stage if the building does not comply with the regulations, it is redesigned. This is iteratively performed until it is approved by the authorities. In addition, any changes by the clients will result in rechecking of the project. ACCCSs will allow the designers to check the projects in any design stage, as the clauses will be arranged accordingly. Moreover, constant changes in design will not infer any additional difficulty or cost about the code checking of design because with the ACCCSs designs can be checked instantaneously.

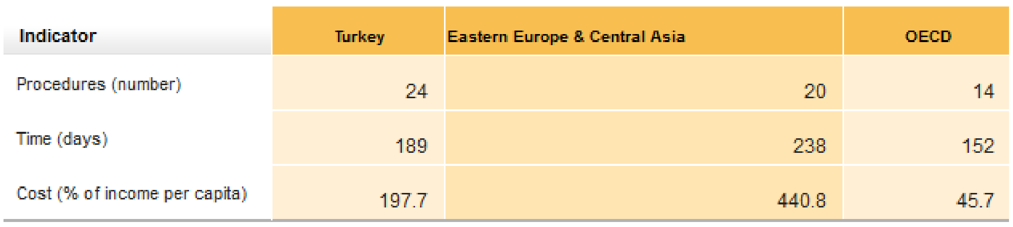
21st century dictates its own criteria on AEC industry like any other industries. These criteria are: "globalization, continued innovation, technology uptake, digitalization, the need for interoperability, standardization of the information and exchange methods etc…" (Yurchyshyna and Zarli, 2009). Globalization changes the way AEC companies work. Today most of the big AEC companies works in more than one country or at least it operates in coordination with foreign firms. As mentioned before they must obey to the rules of the country they work in. Continued innovation and technology uptake must be followed and taken into the process, to be competitive and to lessen the costs. Digitalization allows working faster and more accurately and it makes exchange of information easier. The need for interoperability is more prevalent as there are more criteria in the design that needs to be addressed by different actors. Standardization of the information and exchange methods allows faster, more accurate exchange of information, which in turn allows simultaneous working and interoperability. These values are embraced in the industries like automotive, electronics etc… and in turn, these industries made a leap in productivity, profitability and less error in production. However, AEC industry is more conservative in applying new values into its work cycle, thus cannot profit from the new criteria of the century, as much as other industries. This situation can change with the introduction of new technological tools into the AEC industry and making these tools fundamental to the work cycle of AEC industry. ACCCSs and BIM are the two tools that will help AEC industry, in their quest for modernizing the tools and processes.

All construction projects must take approval before it is started to be build. This approval is given by governmental organizations. Code checkers, check the projects from its building model and these models can be a two dimensional (2D) Computer Aided Design (CAD) drawing or a BIM model. The code checkers do this process manually. Project owners, apply to several branches like fire safety, structure etc… for a project to get the permit. As a result, it takes more time to get an approval, which in turn increases the costs of a project. In addition, this process is open to errors as the code checkers are generally overwhelmed with lots of projects to approve. The development of ACCCSs will take the burden of code compliance checking against dozens of different regulations from the governments. Governments will require much less officials and this will save money. The approval times will considerably get shorter which will make the economies of these governments more competitive and in turn, this can encourage commerce and industry.

Finally yet importantly, the manual code approval process is not transparent enough, as it is not easy to inspect the process to avoid corruption. “Corruption around the world is believed to be endemic and pervasive, and a significant contributor to the low economic growth, inhibition of the provision of public services and increase in the inequality” (Zou, 2006).Therefore, corruption is seen as an obstacle in a healthy country and it needs to be battled. Corruption in the approval stage of design is a serious problem in developing countries. Often in these countries, the code checkers are bribed to get an approval on a project that cannot otherwise receive an approval, or at least they are bribed to shorten the process. There are thousands of different project approvals generally stored in files and folders, thus it is extremely difficult to recheck these approvals. With the introduction of ACCCSs, the code compliance checking process will be completely transparent. Any stakeholder of a project or an official can view the results of the checking. The results will be stored in the systems, and they can be rechecked in the case of a dispute.

When we inspect Turkey with the above-mentioned topics, the current situation is not promising. Firstly, AEC companies do not pay enough attention to the regulations. This fact is prevalent in most of the criteria of the buildings like accessibility, fire safety, earthquake safety etc… For example, most of the public spaces in Turkey lack properly designed accessible structures. In addition, Turkey is a country constantly under threat from earthquakes and several major earthquakes happened in the last fifteen years. The high mortality rate after those earthquakes is found to be because of poorly designed and poorly checked structures. A number of reasons for this poor design and poor checking maybe lack of experts, corruption and lack of proper methods to inspect the designs. All of the three problems can be remedied by introducing ACCCS aimed to work with Turkish codes.

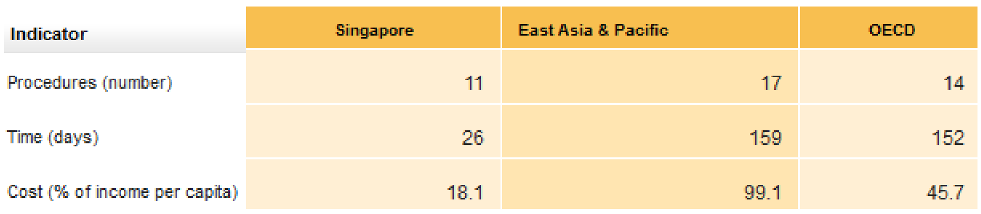
In addition, regarding the easiness and quickness of having one project checked against the required building codes and getting an approval for one’s project, Turkey is one of the worst countries in the World. This fact is reported by Doing Business organization, which assesses the countries all around the World regarding some set of criterion to find the easiness rating of doing business at those countries. In that report, it is shown that one must complete 24 procedures, pay costs worth 197.7% of its income per capita value and it takes 189 days to get a construction permit (Url-1) (Figure 1.1). Turkey is ranked 155th out of 183 economies for the easiness of dealing with construction permits. What is worse is that the current trend is downwards, as Turkey was 153th in ranking the year before which shows it had dropped two ranks. Most of the 189 days that take during the approval stage goes to the code compliance checking. We can compare these values with Singapore’s values to see the effect of ACCCSs. Even though Singapore has a partially working ACCCSs, it is 3rd in rank of easiness in getting an approval (Url-2). It takes only 26 days to get an approval for a project, which is 163 days less than Turkey (Figure 1.2).



Indicator	Turkey	Eastern Europe & Central Asia	OECD
Procedures (number)	24	20	14
Time (days)	189	238	152
Cost (% of income per capita)	197.7	440.8	45.7

Figure 1.1 : Overview of easiness of dealing with construction permit in Turkey report published by Doing Business organization (Url-1).

As can be seen the reasons for developing ACCCS is multifold. It is obvious that ACCCSs will shorten the code checking time, lessen the errors during the checking, make the projects more cost effective because both the checking time will reduce and there will be less errors that must be fixed after the completion of the project. In an industry where even a tiny cost saving will make AEC companies more competitive among the fierce competition in the market, implementing an ACCCS is critical. In addition, ACCCSs will add transparency in the approval process. Because of these, there is an immense interest on developing ACCCSs both in the industry and in the governments. However, although there is an ongoing hard work all over the world, we could not find any study occurring within Turkey for Turkish regulations. That is our motivation for this study.



Indicator	Singapore	East Asia & Pacific	OECD
Procedures (number)	11	17	14
Time (days)	26	159	152
Cost (% of income per capita)	18.1	99.1	45.7

Figure 1.2 : Overview of easiness of dealing with construction permit in Singapore report published by Doing Business organization(Url-2).

1.2 Scope - Contents

Our aim in this study is, to learn the structure of ACCCSs, inspect examples of them, and finally implement a demonstration system that checks a building model against some clauses from Turkish Fire Codes. Implementing an ACCCS is a huge undertaking, there are many different technologies used and the different rules require different expertise. In addition, there are many clauses to address. In order to suit into a master’s thesis, the scope of the implementation is limited to couple of clauses. As a source of the rules, we selected egress rules part from Turkish Fire Codes. The reason in this choice is that egress route codes are complex in nature as it deals mostly with circulation, which is difficult to inspect with computers. This complexity can demonstrate general complexities of automatic code checking algorithms. In addition, there is no system dealing with Turkish egress route rules, in the end of this work there will be a system some clauses of the fire codes of Turkey, which is intended to be evolved into automatic code check system of Turkish Fire Codes in the future.

This thesis is organized into five chapters:

Chapter 2, discusses about ACCCSs and their four stages. These are; 1) “Rule Interpretation” where the written rules are translated into computer recognizable forms, 2) “Building Model Preparation” where the design are transferred into digital world via BIM software, 3) “Rule Execution” where the rules are applied to the building models and 4) “Rule Reporting” where the results and the errors (if any) are displayed to the user.

Chapter 3, is about the examples of ACCCSs. The selected examples are; CORENET – Singapore, DesignCheck – Australia, SmartCodes – USA, General Services Administration – USA and finally Statsbygg – Norway. The chapter gives detailed information about these examples, their scope, contribution and features.

Chapter 4, is about the system we have developed, that is Fire Codes Checker. We have discussed the features, the implementation process and the software that is used in the process. Finally, the system is tested with a building model according to two clauses from the Turkish Fire Codes and results are discussed.

Chapter 5, concludes the thesis. The final remarks and future prospects of the system is discussed in this chapter.

2. **AUTOMATED CODE COMPLIANCE CHECKING**

Automated Code Compliance Checking is the act of checking a building model against regulations, using computerized processes. There is an ongoing effort to automate the code compliance checking process. The earliest efforts started in 1960s with Fenves' (1966) effort on structuring of the codes in decision tables so that they can be resolved easily. These tables are structured such that one can find related clauses by following the branches of a logic tree (Figure 2.1). This work was manual. Later computers started to take part in the studies, one example by Fenves and Wright (1977) was about software tools to manage regulations. With expert systems coming into the scene, these efforts in the structuring of the codes and regulations shifted into developing systems that can automatically assess some clauses from regulations. These systems used 2D CAD drawings as the source of building information. 2D CAD drawings could not accommodate vast numbers of properties that building elements have. As a result, the studies were restricted to some aspects relating to subjects like fire safety, accessibility etc…With the advance of Building Information Modeling, which is "a digital representation of physical and functional characteristics of a facility", the information content of building models improved drastically. It covers three geometric axes (X, Y, Z) along with time and cost as fourth and fifth dimensions. Moreover, it includes spatial relationships, quantities and properties of the building elements and physical information such as light etc… The added complexity of the projects meant that the projects turned to be developed by teams often assigned by governmental institutions instead of small group of researchers developing the projects in the earlier efforts. Today there are ongoing studies about automated code compliance checking systems in the countries like Singapore, USA, Scandinavian countries. Further details on ongoing research will be presented in details in the next chapter.

Tan et. al. (2010) mentions about some important cases to be considered in ACCCSs, these are; 1) rule checking software must point out which object does not obey to the code, 2) most of the codes apply different rules for different situations so rule checking software must consider all of these situations, 3) codes can be changed frequently and the program must be adjusted accordingly, 4) rules are different on each region or country and these changes must be applied in the program, 5) finally if there is not enough conversation between the developers and the rule makers, the software can work in an erroneous way (Han, Kunz, & Law, 1997).

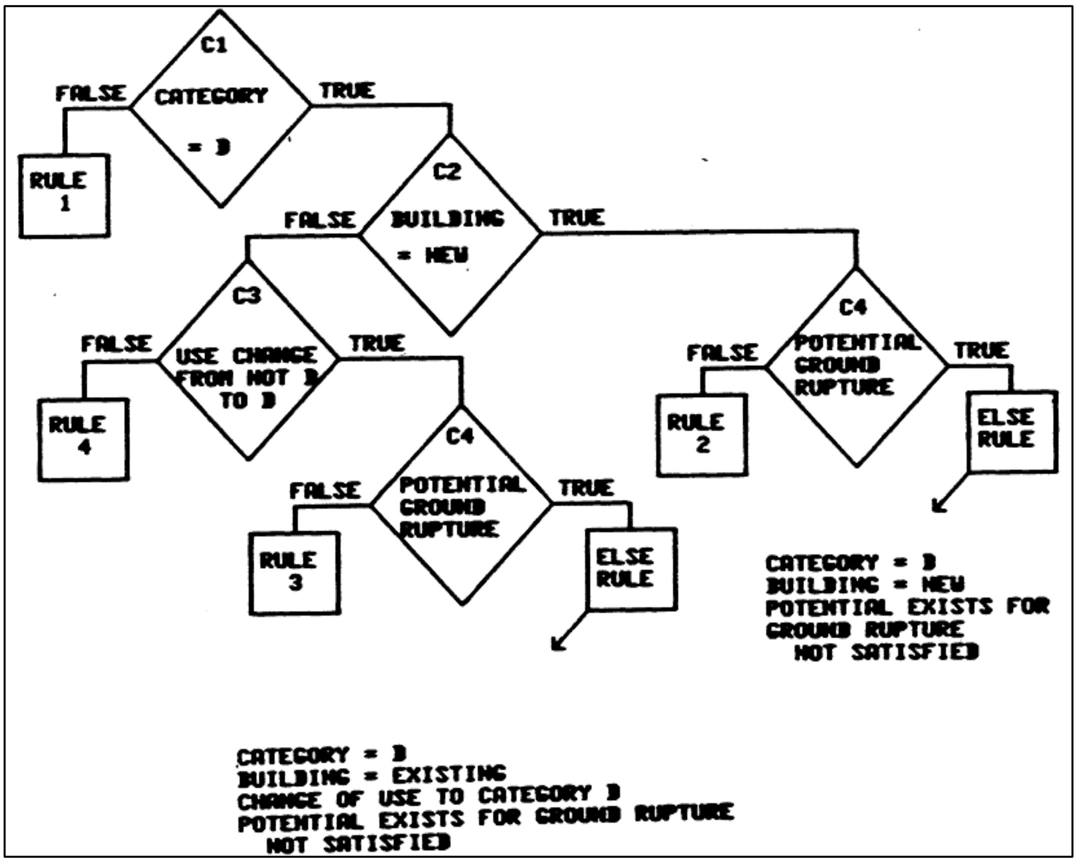


Figure 2.1 : Decision table example taken from Fenves' work (Fenves, 1982).

From the early works and the general structure of all previous code checking examples Eastman et al. (2009) as it will be discussed divides the process into four stages. These are; 1) "Rule Interpretation" where the written rules are translated into computer recognizable forms, 2) "Building Model Preparation" where the design are transferred into digital world via BIM software, 3) "Rule Execution" where the rules are applied to the building models and 4) "Rule Reporting" where the results and the errors (if any) are displayed to the user.

2.1 Rule Interpretation

Building codes and specifications are written by people. They are presented in the local language as written form. They are not always complete; they lack some definitions and some particular cases, even some codes clash. In those cases, common sense of the architects and engineers apply. However, computers do not have any common sense, so every case must be defined without leaving any gap, to have a successful automated code compliance checker.

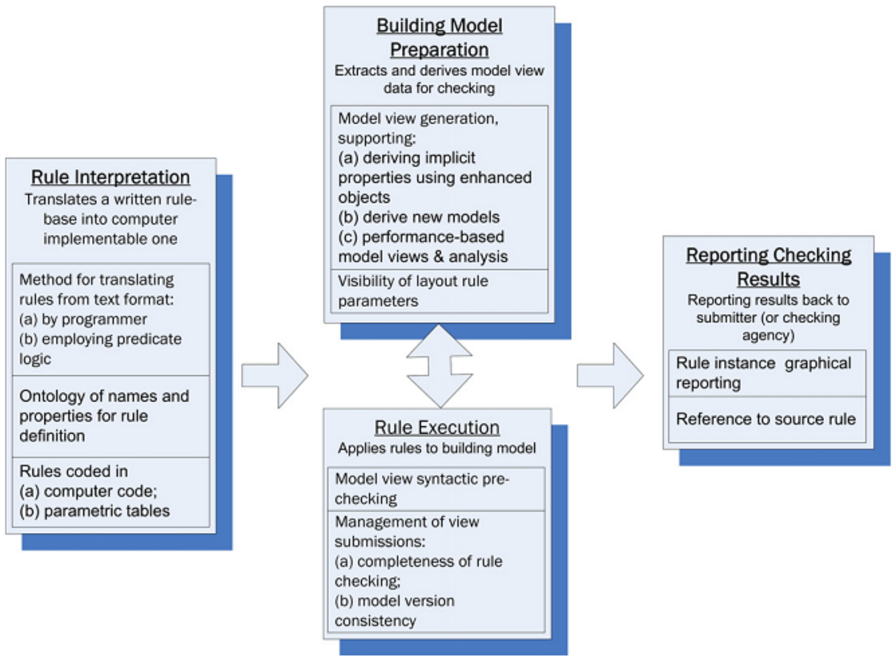

Building Model Preparation
Extracts and derives model view data for checking
Model view generation, supporting:
(a) deriving implicit properties using enhanced objects
(b) derive new models
(c) performance-based model views & analysis
Visibility of layout rule parameters
Rule Interpretation
Translates a written rule-base into computer implementable one
Method for translating rules from text format:
(a) by programmer
(b) employing predicate logic
Ontology of names and properties for rule definition
Rules coded in
(a) computer code;
(b) parametric tables
Rule Execution
Applies rules to building model
Model view syntactic pre-checking
Management of view submissions:
(a) completeness of rule checking;
(b) model version consistency
Reporting Checking Results
Reporting results back to submitter (or checking agency)
Rule instance graphical reporting
Reference to source rule

Figure 2.2 : Overview of four stages of ACCCS (Eastman, 2009).

The organization of the written codes is also complex, which makes it difficult to locate the rules that apply for a domain. This is mainly because the codes are fragmented along different chapters. For example, a rule about egress route for a wheel-chaired user in a hospital can be found in chapters about egress route, hospitals or in “Disabilities Act” section. For manual code checking, this is a time consuming problem as the code checker must review all chapters and apply the ones that is related with the subject. Likewise, this is also a problem for automatic rule checking as well as computer must be aware of the domain it is working in. (This as well becomes a challenge in establishing)

2.2 Building Model Preparation

In the traditional manual checking of building codes, project managers submit their building models as 2D drafts. These 2D representations are aimed for human perception, so their first aim is to represent the entire information available as geometrically correct for humans to check. Eastman et. al. (2009) gives an example for this concept. A stair is a stair when drawn like a stair for a human code checker. However, an object drawn like a stair, is not a stair if it is not defined as a stair object in the building model for an ACCCS. Therefore, today there are more restrictions on how to model the structures. Designers need to select the appropriate building elements while modeling to accurately represent the building, which may be difficult as different modeling tools handle these elements in a different manner. However, this disadvantage is evened out when it comes to added capabilities of object based modeling tools like clash detection, automated code compliance checking etc…

Today, all the efforts in developing an ACCCS use BIM capable modeling tools, when modeling their structures. After generating the building models, they are exported in IFC file format. IFC is a neutral data format that is accepted as a standard file format in BIM application especially on ACCCS. It is designed to help interoperability between different tools.

2.3 Rule Execution

After the building model is setup, the codes that are translated into computer workable algorithms functions, are applied into the building model. However, before these rules are checked, the building model is pre-checked to see if the building model has the required objects, properties etc… required for the checking. This checking can be a full checking, which makes the checking in, one go, or most frequently the check is separated into separate model views. These separate model views are reserved for one part of the rule set.

According to Eastman et. al. (2009) the rule checking is straightforward if these two conditions are met: 1) the rules are translated into computable forms, which involve functions, 2) the functions are prepared with respect to the information of the building model.

Of the when rule checking is executed, the process is partitioned into rule sets. These rule sets are checked separately but to give a complete result of the checking these sub parts must be managed. Management has two important aspects. One is that it must ensure that no rule set is skipped or missed. Second issue is that because there is partial checking and during these checking of the parts the building model may change the results may be inconsistent. This must be avoided.

Due to developing an automated code checking system is a complicated task, which will take many years, during this process of the development there will be some mixture of manual and automatic code checks.

2.4 Rule Reporting

After the checking process finishes, there comes the reporting of the results. In this phase it is critical to include every entity that is checked against a certain rule in the report. By this way, one can check if the rule checking included the all entities from the building model. From thousands of different object groups like doors, windows etc… it is important to check and report every other entity separately. For example, if there is a rule about door width every door must be checked accordingly and reported.

While the rules are implemented in the rule base they are partitioned according to the object type, design stage etc… they refer to. It is of great use for designers or approval committees to select the criteria that they want to see. For example, one architect checking its project in the early design stage, do not need reports about the clauses referring to the specification design stage.

While reporting thousands of different entities, the problem of property referencing these objects arises. For example, for spaces, floors, doors alike there is a naming or numbering schema that is already in use by architects. However, most entities lack proper naming scheme. Therefore, these entities must have a referencing scheme for properly indicating which object is the faulty one. For example, some error in wall thirty-five is very hard to locate within a building model. Instead of this, a graphical reporting methodology is devised. In this report, faulty object needs to be shown with its coordinate in the project and it needs to be shown with a camera focus on it.

In the reports, it is also important to show which clause of the code the object violates. This time the process is reversed. For the object, the program must find the clause it violates and preferably show the definition with the current situation. It is also best way to show improvements in the code.

3. AUTOMATED CODE COMPLIANCE CHECKING SYSTEM EXAMPLES

In this section, we will give examples of rule compliance systems that are developed until today. All of these examples are still in development phase and they are subject to changes. In addition, most projects have commercial value, as it is obvious that companies that develop fully automated code checking systems will have a great asset in their name. Therefore, information to these projects is often limited and it may be possible that some changes occurred in the projects. The sources of the information about these projects come mostly from papers of authors who have been participating in these project groups or informed by attending project presentations, conferences and by personal communication with the project team members.

3.1 CORENET – Singapore

CORENET is an acronym for Construction and Real Estate Network, and it is a project started by Singapore Ministry of National Development in 1995 to "propel the construction and real estate sector into the new millennium by re-engineering the business processes with state-of-the-art information technology to achieve a quantum leap in turnaround time, productivity and quality" (Khemlani, 2005). Singapore Building and Construction Authority (BCA) builds and maintains the CORENET. It is the first working system that became operational.

CORENET is an e-government project and it is a huge undertaking. It is not a coincidence that first example of a working ACCCS appear in Singapore. Singapore has started developing its well thought master plans from the time they got independent in 1965. From those days, there were constant reforms to develop physical development regulations and starting from early 80's they started using information technology aggressively to succeed in better control of physical development (novaCITYNETS, 2012). This willingness to use IT along with strong leadership that force the citizens' to enforce development plans and coordination and communication between agencies and organizations, made the development of CORENET, possible.

CORENET is a "comprehensive network system consisting of a series of IT systems and services that allows seamless and expedient communication and exchange of information between relevant government agencies and parties involved in construction and real estate industry" (Sing and Zhong, 2001). It is developed to cover wide variety of processes in project life cycle (Figure 3.1).

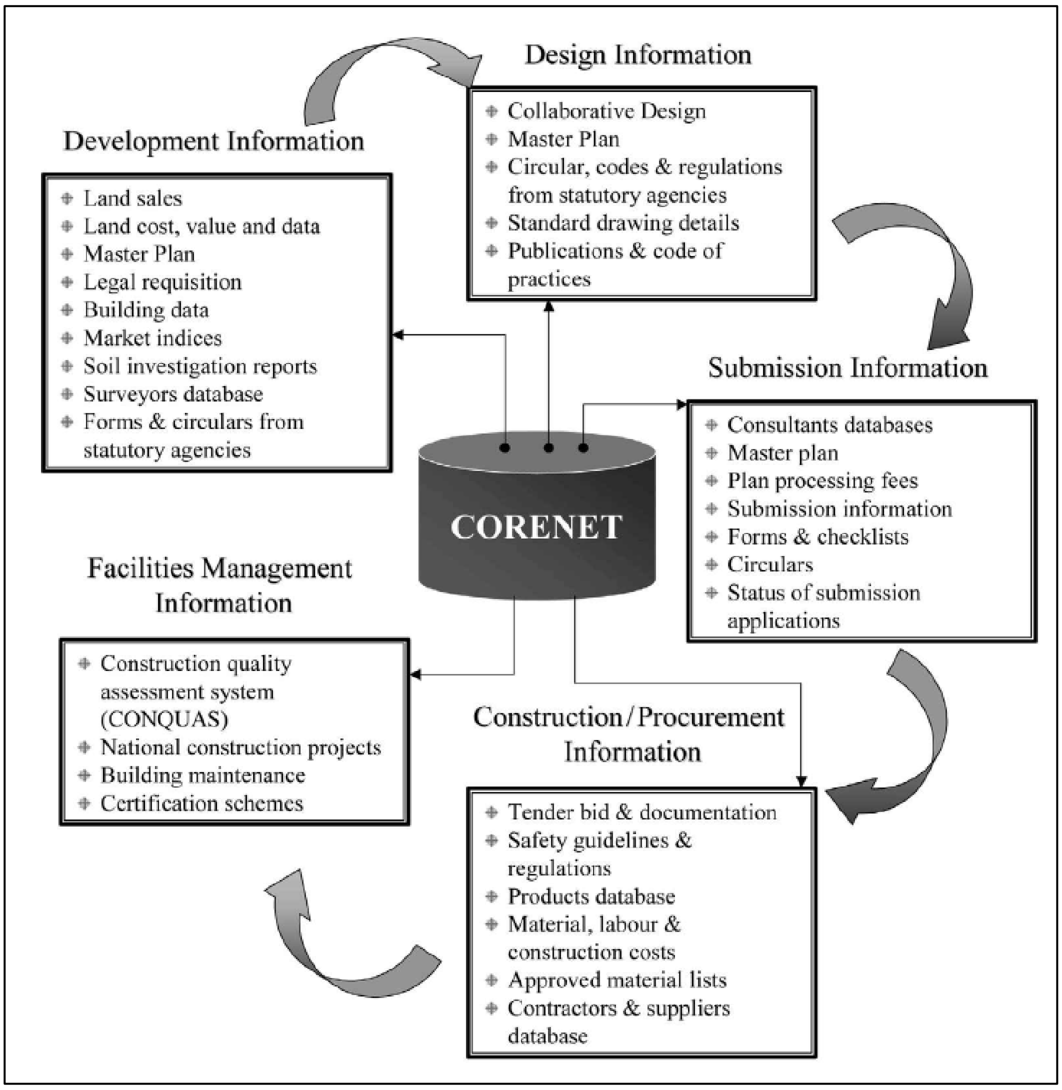

Design Information
Collaborative Design
Master Plan
Circular, codes & regulations from statutory agencies
Standard drawing details
Publications & code of practices
Development Information
Land sales
Land cost, value and data
Master Plan
Legal requisition
Building data
Market indices
Soil investigation reports
Surveyors database
Forms & circulars from statutory agencies
Submission Information
Consultants databases
Master plan
Plan processing fees
Submission information
Forms & checklists
Circulars
Status of submission applications
CORENET
Facilities Management Information
Construction quality assessment system (CONQUAS)
National construction projects
Building maintenance
Certification schemes
Construction/Procurement Information
Tender bid & documentation
Safety guidelines & regulations
Products database
Material, labour & construction costs
Approved material lists
Contractors & suppliers database

Figure 3.1 : The processes from the project lifecycle covered by CORENET (Sing and Zhong, 2001).

CORENET consists of three modules; these are CORENET e-Submission, CORENET e-PlanCheck and CORENET e-Info (Figure 3.2).

CORENET e-Submission is a web-based system and it aims to collect the entire project related documents and drawings needed for the code checking process against variety of different areas in one place. This system has many benefits compared to

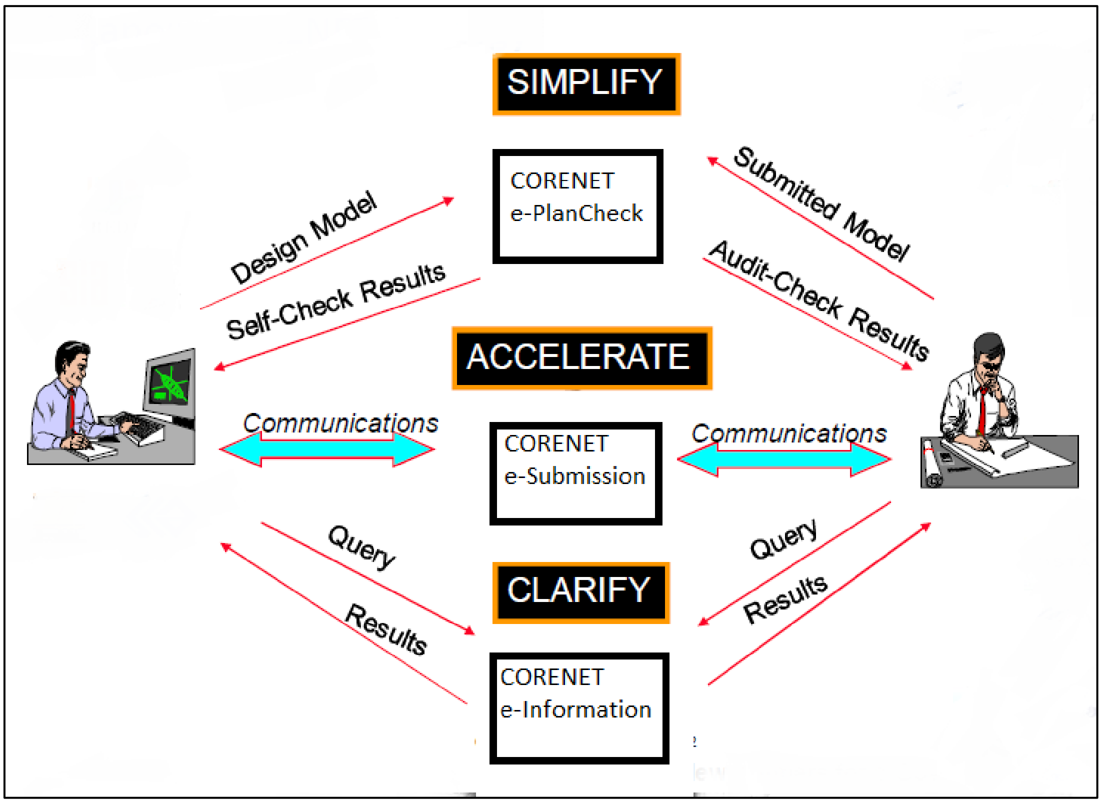
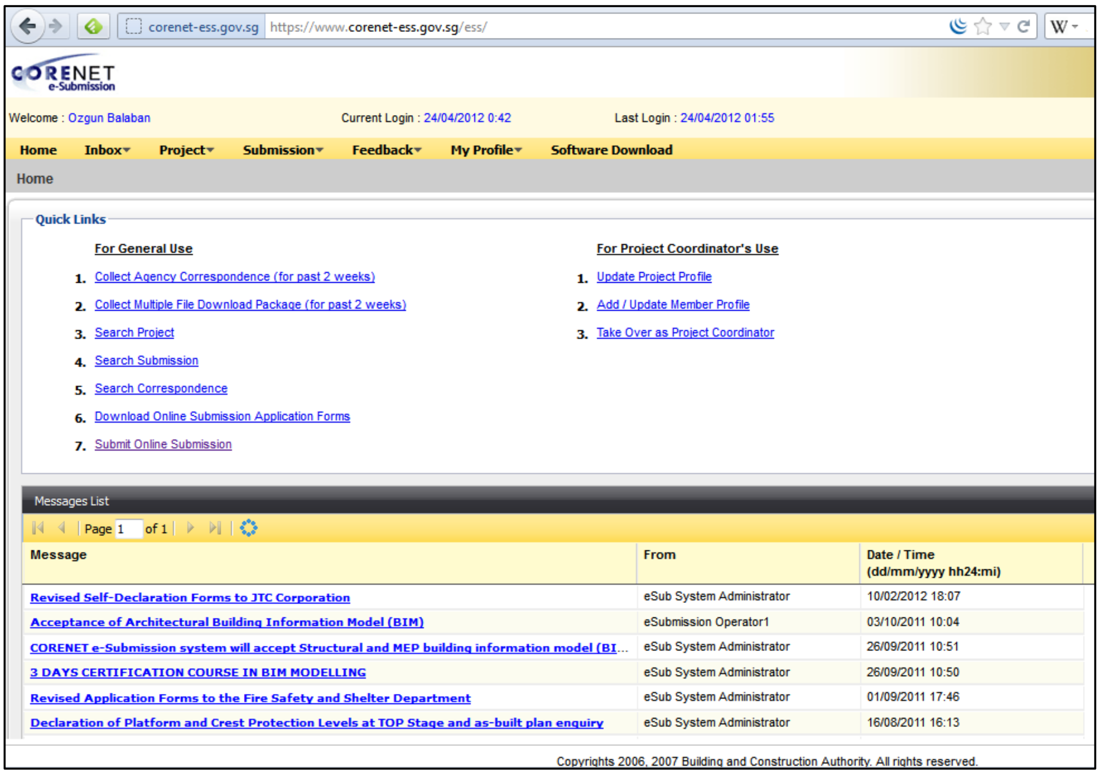


Figure 3.2 : Three modules of CORENET: CORENET e-Submission, CORENET e-PlanCheck and CORENET e-Info (Solihin, 2005).

the traditional building approval process. Firstly, the documents or drawings need not to be printed as they are delivered digitally. This makes e-Submission greener and there will be less space required for storing these digital files. Furthermore, project owners do not need to visit dozens of different officials at different places in the working time, as entire documents about the project can be uploaded from anywhere and at any time. In addition, project owners can check the status of their project approvals and officials can post these status changes from one place. Moreover, the application forms and fees are collected from one place. Besides, transparency of the approval process is improved as every participant can check the progress of the project online. Lastly, since all of the different agencies working in the AEC field are situated in one place their rules and forms become homogenous.

E-Info, is a website for presenting the entire official documents about construction and real estate in one data format online. Project developers can access these documents anywhere, anytime. It is a fast, easy and reliable way of accessing reference materials, thus the importance of the hard copy regulations or other reference materials diminished (Figure 3.4).



Welcome : Ozgun Balaban Current Login : 24/04/2012 0:42 Last Login : 24/04/2012 01:55

Home

For General Use

1. Collect Agency Correspondence (for past 2 weeks)
2. Collect Multiple File Download Package (for past 2 weeks)
3. Search Project
4. Search Submission
5. Search Correspondence
6. Download Online Submission Application Forms
7. Submit Online Submission

For Project Coordinator's Use

1. Update Project Profile
2. Add / Update Member Profile
3. Take Over as Project Coordinator

Message	From	Date / Time (dd/mm/yyyy hh24:mi)
Revised Self-Declaration Forms to JTC Corporation	eSub System Administrator	10/02/2012 18:07
Acceptance of Architectural Building Information Model (BIM)	eSubmission Operator1	03/10/2011 10:04
CORENET e-Submission system will accept Structural and MEP building information model (BI...	eSub System Administrator	26/09/2011 10:51
3 DAYS CERTIFICATION COURSE IN BIM MODELLING	eSub System Administrator	26/09/2011 10:50
Revised Application Forms to the Fire Safety and Shelter Department	eSub System Administrator	01/09/2011 17:46
Declaration of Platform and Crest Protection Levels at TOP Stage and as-built plan enquiry	eSub System Administrator	16/08/2011 16:13

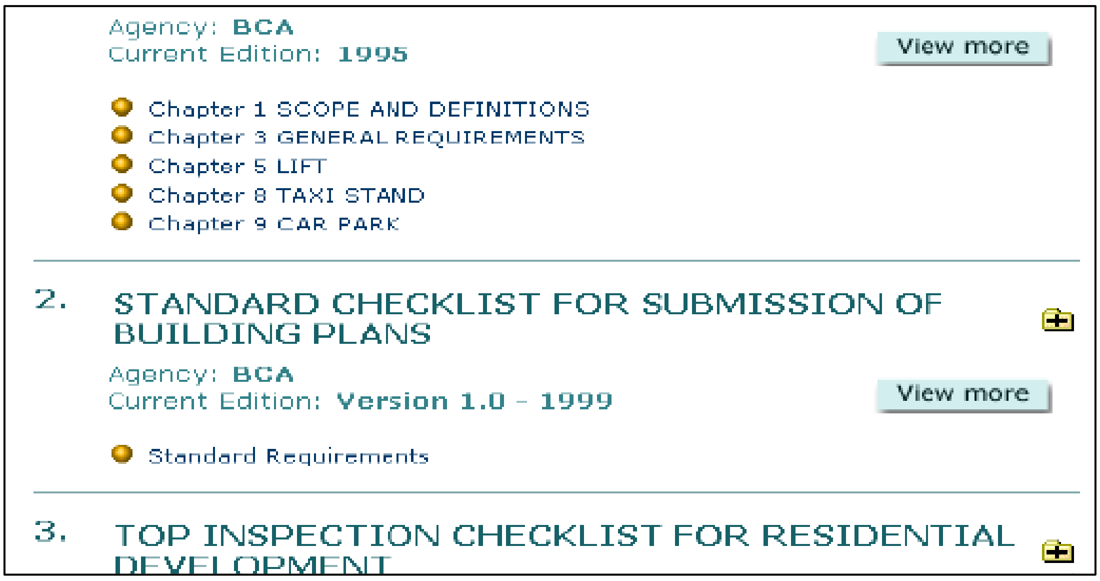
Copyrights 2006, 2007 Building and Construction Authority. All rights reserved.

Figure 3.3: CORENET e-Submission (Url-8).

e-Info uses Extensible Markup Language (XML) for transforming written documents into machine recognizable format without losing its human readability. It is used for transferring documents for use in different applications.

E-PlanCheck module is the most ambitious part of CORENET and its aim is to allow "designs for new buildings to be digitally checked against building codes, using automated procedures, rather than manual paper based processes" (buildingSMART, 2002).

It has a long history of development. It first started with the idea of checking CAD plans with the help of Artificial Intelligence (AI) in 1982 (Fatt, 2005). In 1984 and 1988, attempts to develop a successful plan checking system failed. In 1991, a master plan for "transforming Singapore into an intelligent island" (Sing and Zhong, 2001) is developed and its result in construction and real estate was checking of the feasibility of the AI plan checking idea one more time.


Agency: BCA
Current Edition: 1995

View more

- Chapter 1 SCOPE AND DEFINITIONS
- Chapter 3 GENERAL REQUIREMENTS
- Chapter 5 LIFT
- Chapter 8 TAXI STAND
- Chapter 9 CAR PARK

2. STANDARD CHECKLIST FOR SUBMISSION OF BUILDING PLANS

Agency: BCA
Current Edition: Version 1.0 - 1999

View more

- Standard Requirements

3. TOP INSPECTION CHECKLIST FOR RESIDENTIAL DEVELOPMENT

Figure 3.4 : CORENET e-Info (Fatt, 2005).

As a result, of this feasibility study, the idea was found doable, but it is also discovered that a new standard on intelligent CAD data format is needed for this effort.

In the time from 1994 and 1997, a neutral CAD data format for Singapore and an AI plan checking system is developed. It is called Building Plan Expert or BP-Expert. The experimentations with this system found some serious shortcomings. First of all, it was very costly to develop and maintain a local, proprietary CAD data format valid only for Singapore. In addition, it was not flexible enough for a complex system like this. As a result, it did not operate well with inconsistent or bad data, which was very frequently encountered. In addition, it only covered some building codes clauses. Added to these were performance issues which in the end make this system unusable.

Although the system was not usable, the effort for developing it proved vital for the general effort. The developers noted two key issues required for the success of the project. First, it is noted that it was critical to work with an international 3D CAD data. Secondly, it is also important to make other actors like software vendors or people from the construction industry participate the process.

With these points in mind developers started to work for the system which is still used today. The project team found the building model they are looking for and it was IFC. IFC provided them the intelligent data format, which they require to access building elements and their properties.

Some information required for clauses are easy to retrieve from the building model using IFC. For example, the clause 2.2.7 from "means of egress" requires "No exit, exit staircase or other exit facilities shall be narrower than the maximum width requirement as specified under table 2.2A. The minimum clear width of an exit door opening shall be not less than 850mm". When checking for this clause first thing to do for a programmer or codes official is to be sure about what does "minimum clear width" refers to. They checked it and found that it is the measurement of the nominal width minus the width of the door jamb (Figure 3.5.a). Next, they match this information from the IFC. There is no IFCDoor.ClearWidth attribute so it is not possible to extract "minimum clear width" directly. Instead, there are IFCDoor.OverallWidth for nominal width of the door and IFCDoorLiningProperties.LiningThickness for width of the door jamb, attributes available in IFC (Figure 3.5.b). Therefore, to find the value of clear width programmers subtract the value of IFCDoorLiningProperties.LiningThickness from IFCDoor.ClearWidth. This is an easy and straightforward operation.

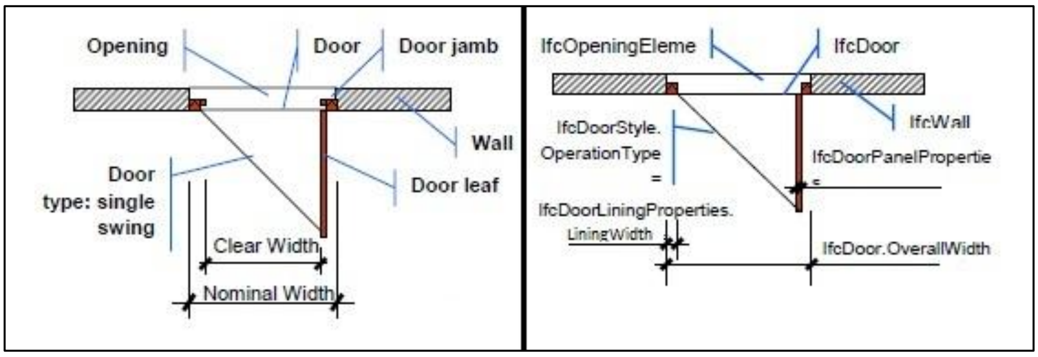
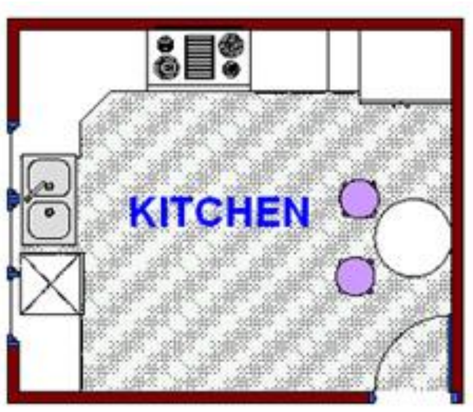
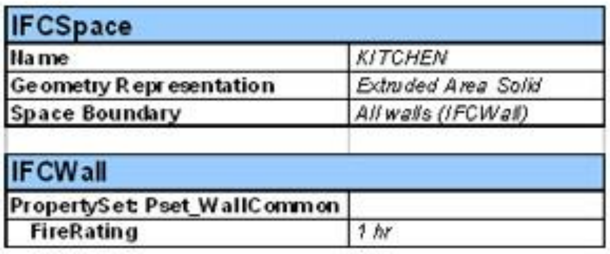


Figure 3.5 : a) Normal definitions about door, b) IFC definitions about door (Liebich et. al., 2002).

However, IFC itself is not enough for a successful code-checking system. It is because "IFC only represents basic building information model that can be captured with BIM application during design stage" (Khemlani, 2005). In IFC, building objects are basic and their properties carry limited and static information about these objects. Development of an automated code checking system from these basic objects is a tedious and nearly impossible work. The inability of IFC to represent the required information is showed in a presentation of novaCITYNETS (Solihin, 2005). The first example is a kitchen space example. A kitchen is represented with a space object (IFCSpace) which takes its boundaries from IFCWall object. (Figure 3.6) The

Singapore fire code requires kitchen must be compartmentalized with minimum 1hr fire rating and it must have enough ventilation if mechanical ventilation is not provided. So kitchen space needs to satisfy some special requirements but in IFC it has no distinction from other spaces, in other words there is no IFCKitchen object. Therefore when checking against fire codes the system cannot differentiate a kitchen space from another space and as a result that kitchen space cannot be checked if it satisfies the special requirements for kitchens or not.





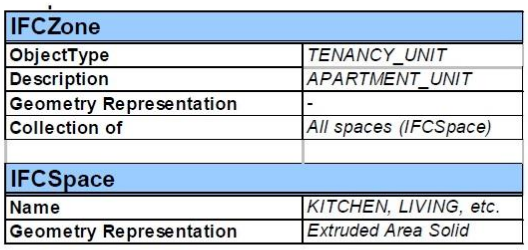
IFCSpace	
Name	KITCHEN
Geometry Representation	Extruded Area Solid
Space Boundary	All walls (IFCWall)
IFCWall	
PropertySet: Pset_WallCommon	
FireRating	1 hr

Figure 3.6: Kitchen and its representation in IFC (Solihin, 2005).

The kitchen example is a relatively easy one case to find a work around to work with solely on IFC. However, there are more complex issues like in the case of apartment unit zone example. The apartment unit is a collection of spaces (Figure 3.7) which can be found in an apartment dwelling. The Singapore fire codes requires an apartment unit to satisfy "most remote distance within the apartment to the exit door must be with in 20 m". By only using IFC it is a tedious task to check for this clause, which needs lots of calculations from IFC geometries that are difficult to manage.

To complement the limited capabilities of the IFC, novaCITYNETS built a Code Checking Object Model (CCOM) which is named as FORNAX. FORNAX is implemented to extend the information found in the IFC. It is a "model representing both the building geometry models in 3D and the semantics information such as the relationships and the behaviors of the building elements" (Xu, Solihin and Huang 2004).

The FORNAX objects is encapsulation of simple building components (Figure 3.8), by this way the programmers do not need to develop separate algorithms for all the



IFCZone	
ObjectType	*TENANCY_UNIT*
Description	*APARTMENT_UNIT*
Geometry Representation	-
Collection of	*All spaces (IFCSpace)*
IFCSpace	
Name	*KITCHEN, LIVING, etc.*
Geometry Representation	*Extruded Area Solid*

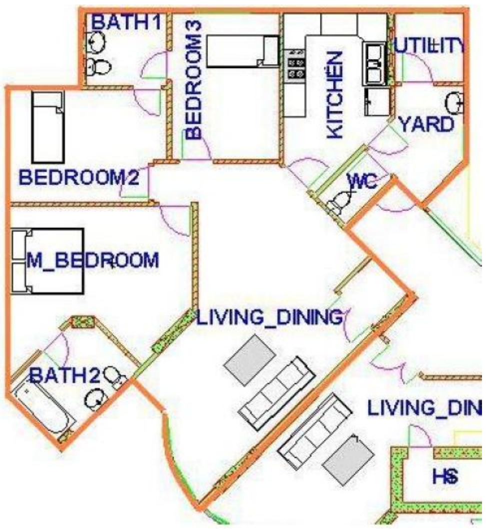


Figure 3.7: Apartment Unit and its representation in IFC (Solihin, 2005).

required calculation that is needed. Instead, it is possible to use FORNAX objects and their supplemented functions and attributes to check the requirements of several codes. As a result translating a written code into computer process is straightforward.

As an example, take the apartment unit we mentioned before. It was difficult to calculate "the most remote distance to exit" by using IFC. In the case of FORNAX developers produced FXApartmentUnit object which has its functions related with the fire safety listed in Figure 3.9. By using these functions, programmers can generate a procedure to find the required distance easily. In addition, a snapshot of checking the distance in the example apartment unit is provided in Figure 3.10. It can be noted that as a result of the algorithm, the distance between the remote point (blue circle in the figure) and exit door (red circle) is founded to more than 20 meters which is more than the allowed distance in Singapore fire codes, thus the system gave an error.

FORNAX does not replace IFC. It takes basic object information and its associated geometry from IFC model and using some geometric operations finds out information such as spatial information, network information and design constraints and adds it into its repository. Spatial information gives relative place of other objects, network information makes drawing of paths and assessing the connection of spaces possible and design constraints is about how a certain object is defined. The FORNAX system is composed of four parts; database for storing information, ACIS and Open Cascade as geometry engines and lastly IFC. It is developed and maintained by novaCITYNETS.

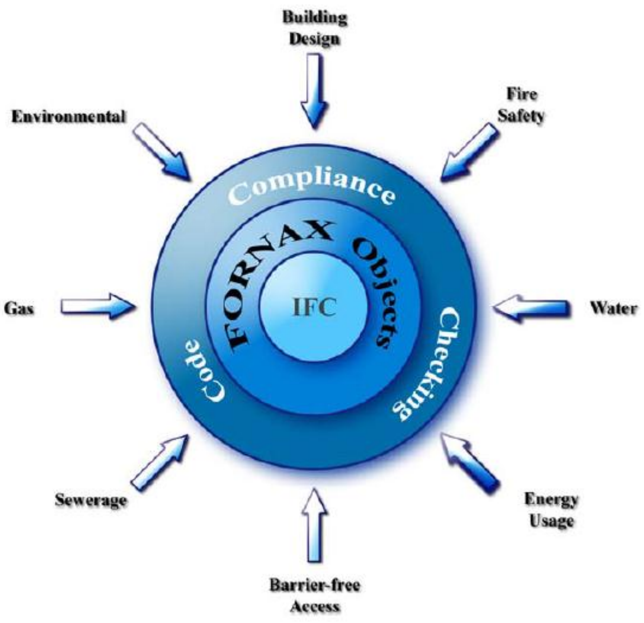
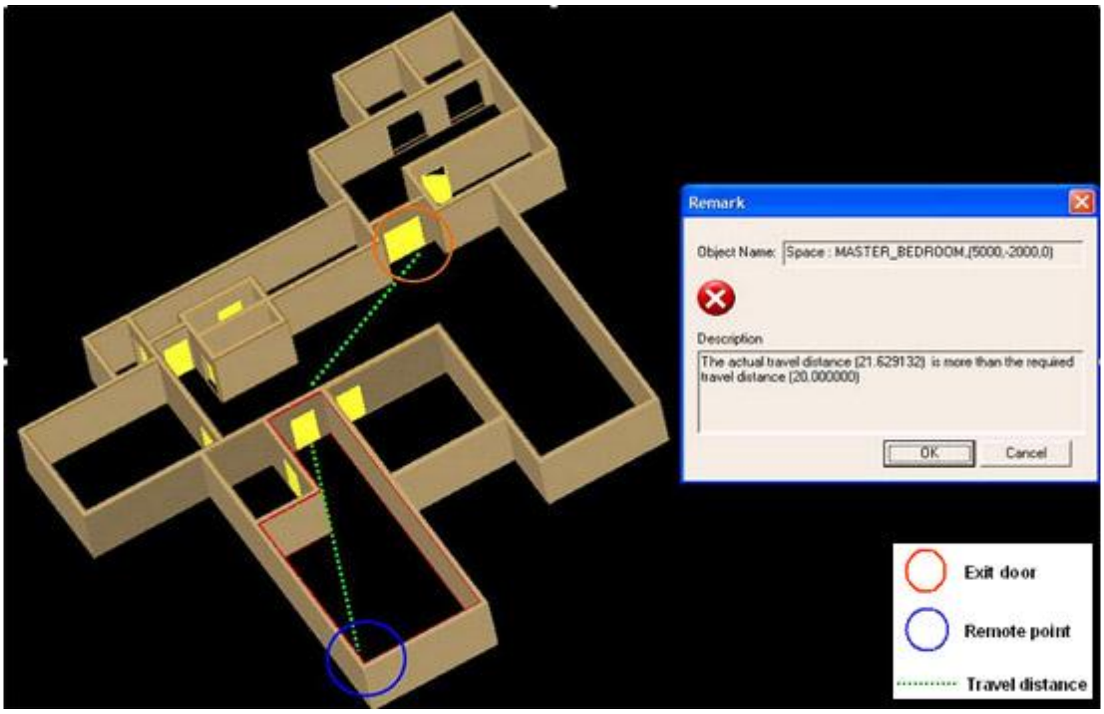


Figure 3.8 : Encapsulation of IFC into FORNAX objects (Khemlani, 2005).

FXApartmentUnit	
Methods	Description
GetSpaces	Get the spaces that make the apartment unit.
GetExit	Finds the exit of the apartment.
CalculateRemotePoint	Calculate the remote point in a space from its doors.
CalculateTravelDistance	Calculate the travel distance from one point to nearest exit door.
CalculateArea	Calculate the area of the apartment.
CalculateVolume	Calculate the volume of the apartment.

Figure 3.9 : Example of methods used by FXApartmentUnit.

E-PlanCheck can output the results in the popular document formats such as PDF, DOC or HTML. They are presented within a website. It supports giving reference to the written code clause while listing the elements that do not comply with the codes. The reporting module has graphical presenting capabilities.



Object Name: Space : MASTER_BEDROOM,(5000,-2000,0)

Description

The actual travel distance (21.629132) is more than the required travel distance (20.000000)

Figure 3.10 : An example of travel distance calculation by using FORNAX (Khemlani, 2005).

CORENET is used efficiently for automated code compliance checking in Singapore. It is much more mature than other examples we will review here. Today thousands of engineers and architects use CORENET successfully. However, CORENET is not aimed for use during design stage; it is only used by the governmental agencies. In this aspect it differs from the other systems. We have adapted the information module of CORENET into our system to be used for giving information to the users.

3.2 DesignCheck – Australia

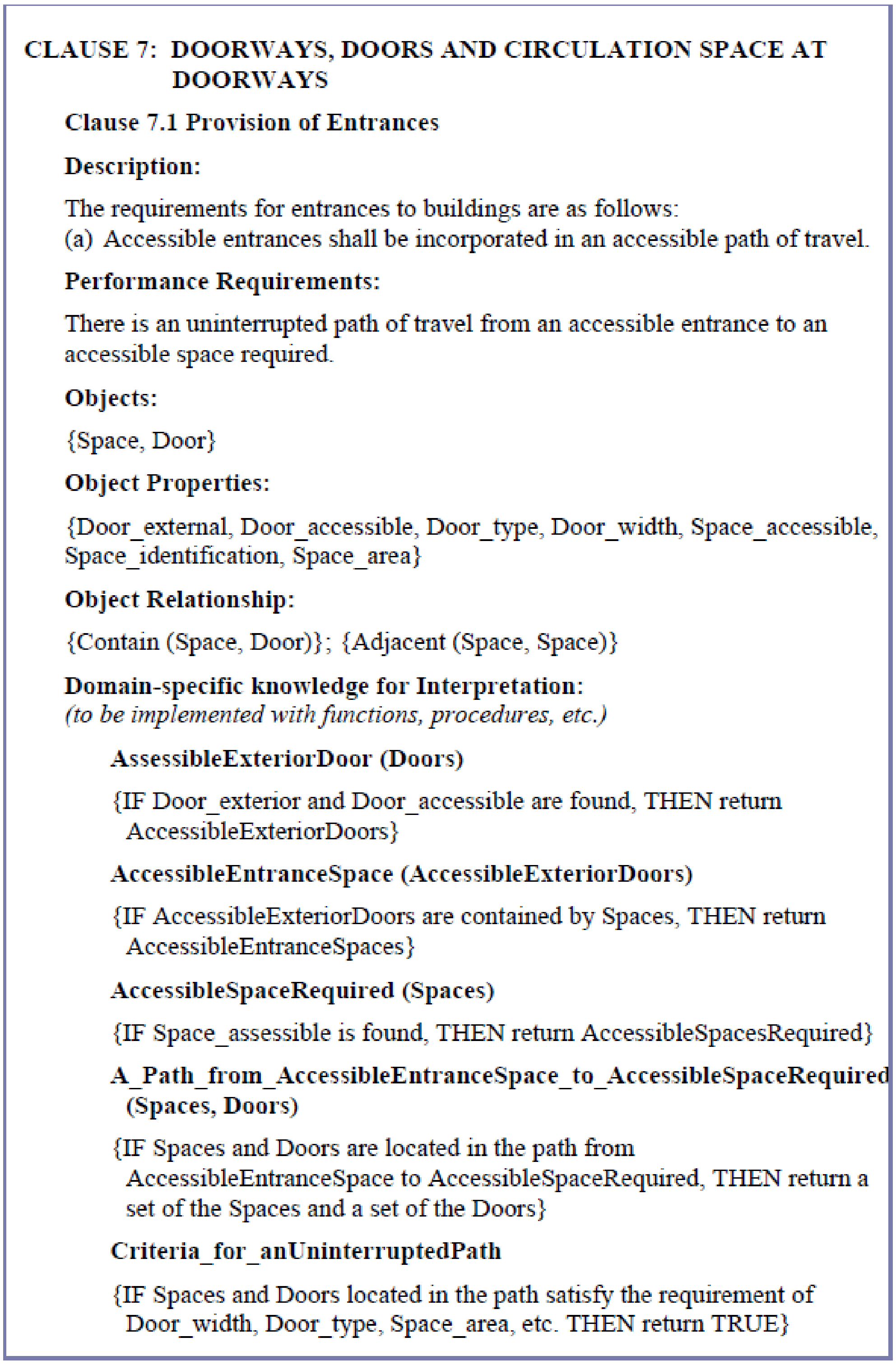
The Cooperative Research Center for Construction Innovation funded the DesignCheck project and it was undertaken by University of Sydney and Commonwealth Scientific and Industrial Research Organisation (CSIRO). The aim was to develop an automated code checking system for Australia which will "enable quick and easy compliance assessment against building codes and assist designers in finding potential problems early" (Ding et al, 2006). Design for access and mobility, Part 1: General requirements for access – new building work (AS1428.1) from Australian Standards (AS) and New draft access code for buildings Part D – Access and Egress (D3) from Building Code Australia (BCA) are selected as exemplar codes and these codes are about accessibility.

Rather than starting from scratch, DesignCheck team aimed to use the existing rule based systems and develop them further according to the requirements of the project. Thus, as a starting point, DesignCheck project group started with a review of two existing commercial rule based systems; Express Data Manager (EDM) and Solibri Model Checker (SMC). In this review SMC proved to have advantages like directly interfacing to BIM systems and excellent performance of its reporting system with the option of showing graphically the noncomplying building elements. However, it was not flexible enough and it was not possible to encode the design requirements for different stages of design as it did not offer modifications to its rule base directly. Instead, EDM allowed modifications in the rule schema so it allowed the DesignCheck team to modify the rule base according to their needs. Also EDM showed its automated code checking capabilities which consists importing of the building model into the database, checking of the imported building model against the constructed EDM rule schema and finally reporting the building elements that failed to comply with the clauses. Nevertheless, EDM lacked the user friendly interface of the SMC especially, it did not offer graphical reporting capability of the SMC so the reports of the checks were only in text form (Ding, 2004). After this review, they have decided to continue on the project using EDM because it had offered more flexibility and capability.

In the interpretation of written building codes, DesignCheck design team considered following general strategy: (Ding et al, 2006)

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

DesignCheck uses object oriented techniques for transforming written statements from the building codes into computer interpretable structures. DesignCheck team has prepared a pre-implementation specification structure that is used before the translation process begins. This structure consists of: Description, Performance requirements, Object, Properties, Relationships, and Domain specific knowledge for interpretation. Ding et al. gives an example of how this structure is used (2006). The subclause of Clause 7.1 Provision of entrances from the AS1428.1, states "Accessible entrances shall be incorporated in an accessible path of travel" (Figure 3.11). This statement becomes the description of the clause, and a programmer or a code official derives the other elements used in the structure by interpreting this description using IFC object-based interpretation. In this case, the performance requirement for this description is derived as "There is an uninterrupted path of travel from an accessible entrance to an accessible space required". Next, the objects required for these clauses are determined to be Space and Door. After this the properties and relationships of these objects that will be needed for this clause is founded. The relationships are: Door_external, Door_accessible, Door_type, Door_width, Space_accessible, Space_identification, Space_area and the relationships are "Space contains Door". From using all these objects and properties and relationships of objects, pseudecodes are developed by the programmers to help in the implementation of the rule schema and these are named as domain-specific knowledge for interpretation in the pre-implementation specification structure. For instance, for a door object to be an exterior accessible door, it needs to be specified as exterior and accessible in its properties. In the pseudecode structure, it is evaluated with the following conditional statement: "IF Door_exterior and Door_accessible are found, THEN return AccessibleExteriorDoors. This becomes a function to find all of the accessible exterior doors. All of the related functions are listed in the domain-specific knowledge for interpretation part and after this step; the pre-implementation specification structure for clause 7.1 becomes complete. All of the clauses from AS 1428.1 and BCA D3 were encoded in the same structure to help in the translation process.



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_exterior and Door_accessible are found, THEN return AccessibleExteriorDoors}

AccessibleEntranceSpace (AccessibleExteriorDoors)

{IF AccessibleExteriorDoors are contained by Spaces, THEN return AccessibleEntranceSpaces}

AccessibleSpaceRequired (Spaces)

{IF Space_assessible 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 3.11 : DesignCheck uses pre-implementation specification structure (Ding et al., 2006).

DesignCheck is primarily designed to help architects and engineers during the design process. For this reason, the rules are separated according to the design stages they apply. At the early design stage, designers are interested in the connection of spaces like accessible paths to WCs etc… At the detailed stage of the design, designers are occupied with measurements like door width, handrail height etc… At the specification stage, designers are interested with specifications of building elements like material of a door etc… To accommodate this structure every rule is separated into the design stage they apply. A designer when checking his model, can select the design stage to evaluate the building model according to the applicable clauses. The results of checking different design stages are stored in the database to be used in the next checking (Figure 3.12).

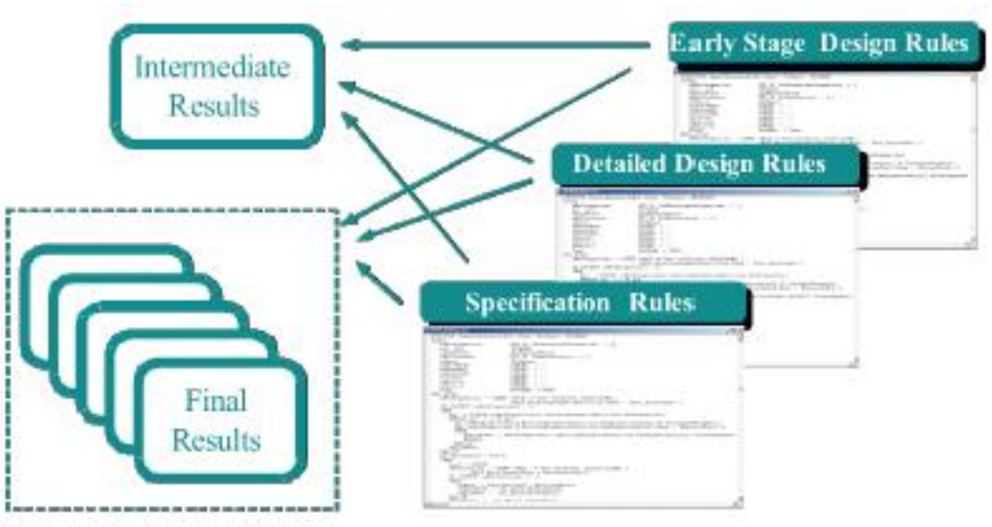
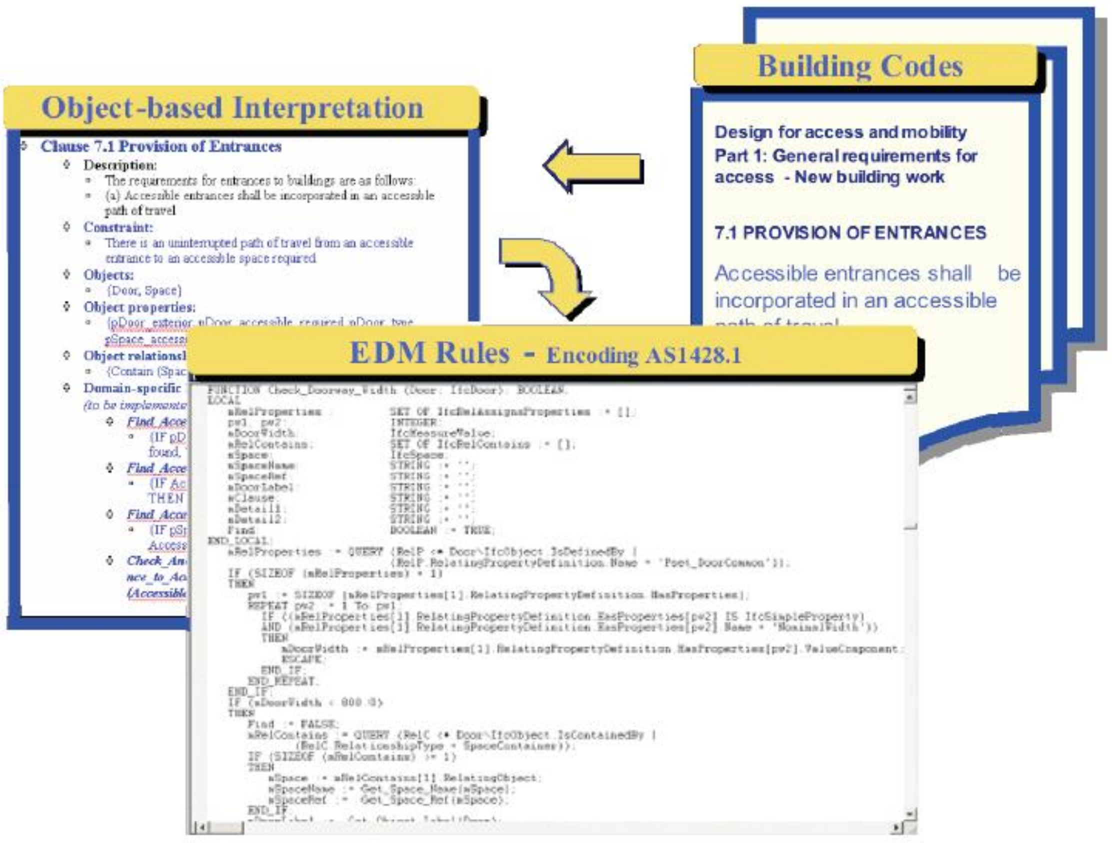


Figure 3.12 : Checking of the model in different design stages (Ding et al., 2006).

After the encoding of the pre-implementation specification structure is done, the extracted functions are mapped into EDModelChecker using ExpressX language to define the rule schema that will be used. This step is rather easy as the objects, object properties and object relationships along with the functions are extracted in the prior stage. One more reason for this stage to be easy is that ExpressX has ready mapping function, which efficiently and easily maps IFC to ExpressX. With this step, the translation of the written codes into a rule schema is finished and the rule schema is ready to use. To recap, written codes is first encoded into pre-implementation structure and then it is turned into rule schema (Figure 3.13).



Design for access and mobility
Part 1: General requirements for access - New building work

7.1 PROVISION OF ENTRANCES

Accessible entrances shall be incorporated in an accessible

- 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
  - Constraint:
    - There is an uninterrupted path of travel from an accessible entrance to an accessible space required
  - Objects:
    - {Door, Space}
  - Object properties:
    - {pDoor_exterior, pDoor_accessible_required, pDoor_type
  - Object relationsl
    - {Contain (Spac
  - Domain-specific
  - *(to be implemente*
    - *Find_Acce*
      - (IF pD found,
    - *Find_Acce*
      - (IF Ac THEN
    - *Find_Acce*
      - (IF pSp Access
    - *Check_An nce_to_Ac (Accessible*

```
FUNCTION Check_Doorway_Width (Door: IfcDoor): BOOLEAN;
LOCAL
   aRelProperties :          SET OF IfcRelAssignsProperties := [];
   pw1, pw2:                 INTEGER;
   aDoorWidth:               IfcMeasureValue;
   aRelContains:             SET OF IfcRelContains := [];
   aSpace:                   IfcSpace;
   aSpaceName:               STRING := '';
   aSpaceRef:                STRING := '';
   aDoorLabel:               STRING := '';
   aClause:                  STRING := '';
   aDetail1:                 STRING := '';
   aDetail2:                 STRING := '';
   Find:                     BOOLEAN := TRUE;
END_LOCAL;
   aRelProperties := QUERY (RelP <* Door\IfcObject.IsDefinedBy |
                            (RelP.RelatingPropertyDefinition.Name = 'Pset_DoorCommon'));
   IF (SIZEOF (aRelProperties) = 1)
   THEN
      pw1 := SIZEOF (aRelProperties[1].RelatingPropertyDefinition.HasProperties);
      REPEAT pw2 := 1 To pw1;
        IF ((aRelProperties[1].RelatingPropertyDefinition.HasProperties[pw2] IS IfcSimpleProperty)
        AND (aRelProperties[1].RelatingPropertyDefinition.HasProperties[pw2].Name = 'NominalWidth'))
        THEN
           aDoorWidth := aRelProperties[1].RelatingPropertyDefinition.HasProperties[pw2].ValueComponent;
           ESCAPE;
        END_IF;
      END_REPEAT;
   END_IF;
   IF (aDoorWidth < 800.0)
   THEN
      Find := FALSE;
      aRelContains := QUERY (RelC <* Door\IfcObject.IsContainedBy |
              (RelC.RelationshipType = SpaceContainer));
      IF (SIZEOF (aRelContains) >= 1)
      THEN
         aSpace := aRelContains[1].RelatingObject;
         aSpaceName := Get_Space_Name(aSpace);
         aSpaceRef := Get_Space_Ref(aSpace);
      END_IF
```

Figure 3.13 : Written codes is first encoded into pre-implementation structure and then it is turned into rule schema (Ding et al., 2006)

This structure of rule translation is successful to derive the rule schema that will be used for code checking of the most the clauses. However, in the clauses that are about the configuration criteria like in the example clause 7.1, it is difficult to use the same approach (Eastman, 2009). Clause 7.1 is about access rather than some calculation of paths, thus to remedy the problem a graph approach has been devised. "Accessible entrances" and "accessible path of travel" are defined (Figure 3.14). Starting from the "accessible entrances", all neighboring spaces that have "accessible path of travel" between itself and the "accessible entrance" are regarded as "accessible". In the figure 3.14, all the spaces except Office1 has "accessible path of travel" which is shown as straight line, thus they are "accessible". However, the Office1 space has only a non-accessible path of travel between itself and Office2, which is shown with dashed line and it, has no other connection to another space. Therefore, Office1 is a non-accessible space.

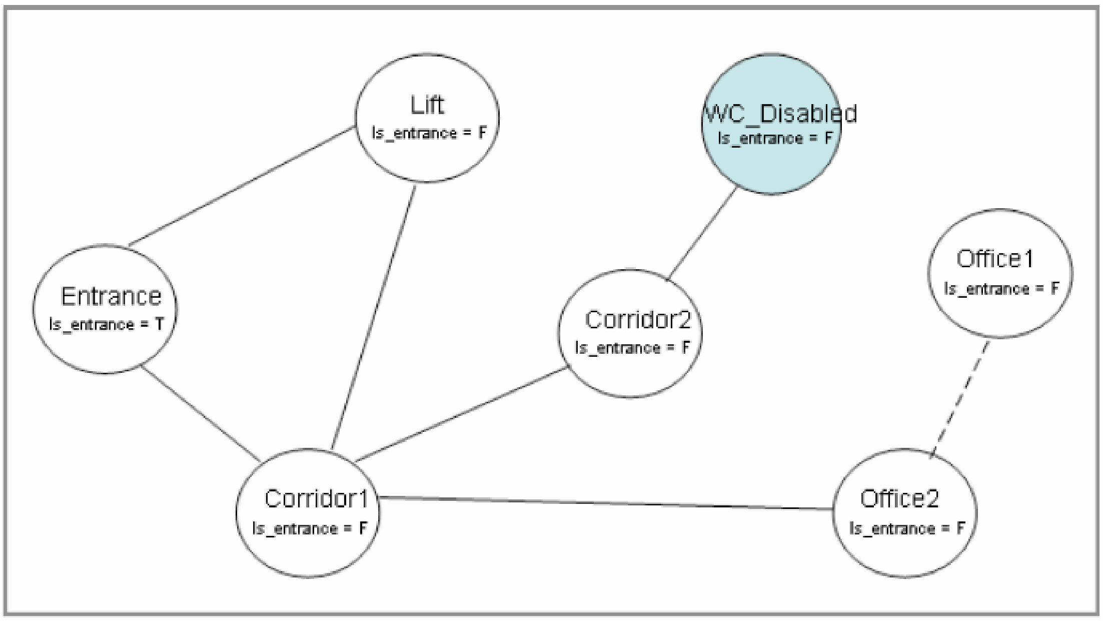
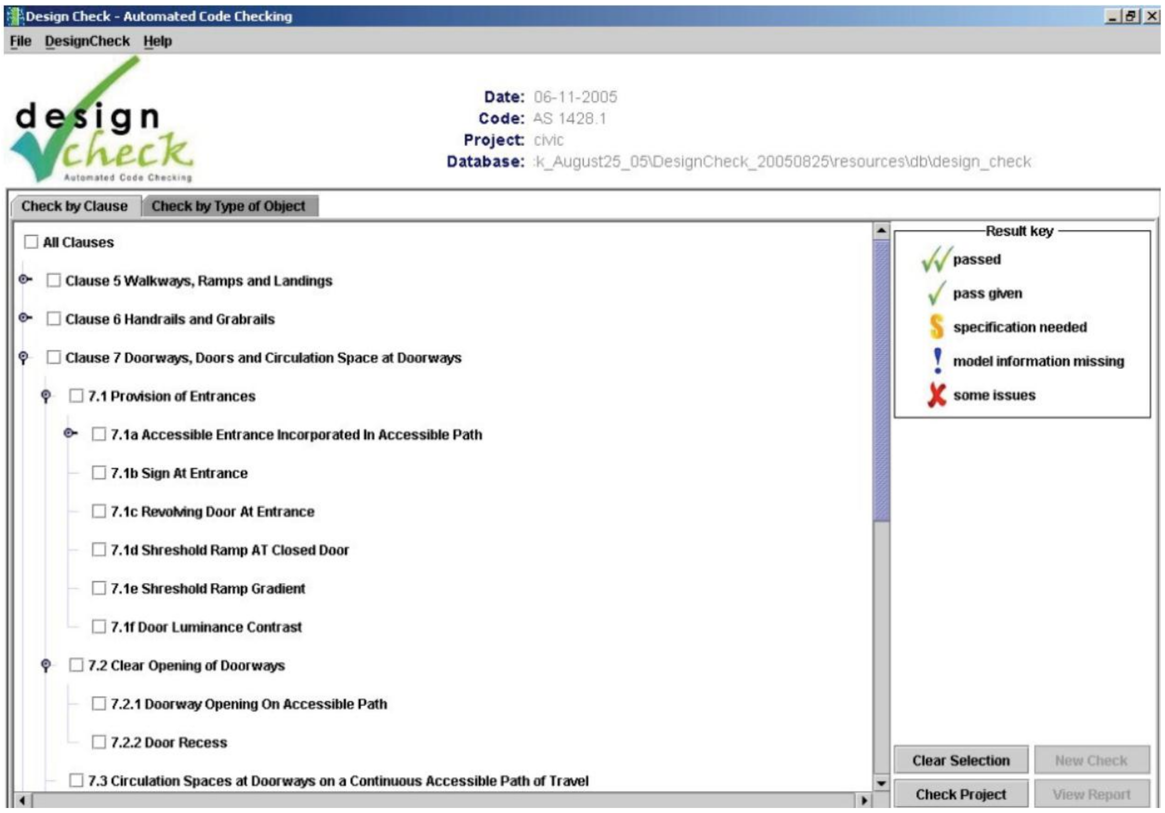


Figure 3.14 : Path finding algorithm (Ding et al., 2006).

Automatic code compliance checking requires information rich building model for successfully inspecting the needed requirements from the building codes. At the time DesignCheck was prepared, object-based CAD tools – the CAD tools that support building information modeling and IFC was beginning to appear into the scene. However, the information they provide was not enough to cover all of the building code requirements. As a result DesignCheck started by devising better building model by adding DesignCheck Internal model with the tools they have in hand.

They started by selecting BIM capable CAD tool. As a result ArchiCAD 9 was selected. Then, before any checking against compliance for building codes takes place, the building model prepared in ArchiCAD is exported to IFC model by ArchiCAD's own exporting capability. Afterwards this IFC model is converted into DesignCheck internal model. For automated translation of IFC model to DesignCheck internal model, a mapping schema is produced using ExpressX language. Because ExpressX has ready mapping functions, the mapping between IFC to DesignCheck internal model was easier. This mapping structure for DesignCheck is open to further modification and extension if any requirement arises.

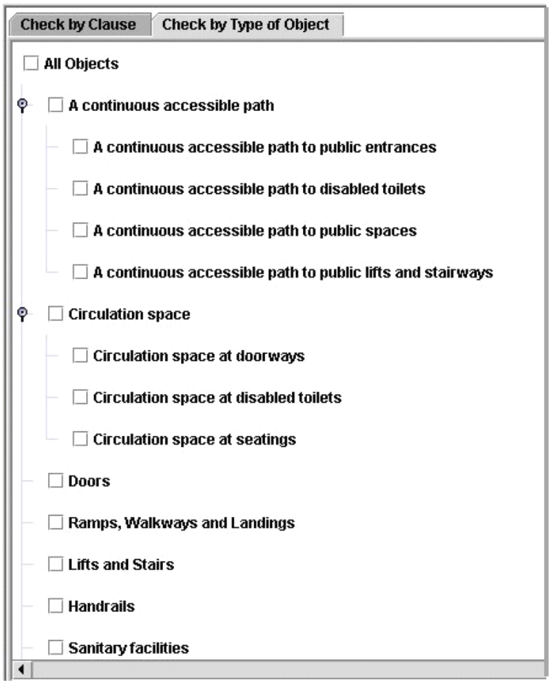
After the building model is ready, it can be checked against the codes in DesignCheck. The results of the checking are shown in an interface that is implemented in Java with HTML. Users can check their model by selecting the clauses they want to apply to the model (Figure 3.15), or they can check the object types and see if these object types comply with the rules (Figure 3.16). The results of these checks are shown in a graphical report page (Figure 3.17).



Date: 06-11-2005
Code: AS 1428.1
Project: civic
Database: :k_August25_05\DesignCheck_20050825\resources\db\design_check

- All Clauses
- Clause 5 Walkways, Ramps and Landings
- Clause 6 Handrails and Grabrails
- Clause 7 Doorways, Doors and Circulation Space at Doorways
  - 7.1 Provision of Entrances
    - 7.1a Accessible Entrance Incorporated In Accessible Path
    - 7.1b Sign At Entrance
    - 7.1c Revolving Door At Entrance
    - 7.1d Shreshold Ramp AT Closed Door
    - 7.1e Shreshold Ramp Gradient
    - 7.1f Door Luminance Contrast
  - 7.2 Clear Opening of Doorways
    - 7.2.1 Doorway Opening On Accessible Path
    - 7.2.2 Door Recess
  - 7.3 Circulation Spaces at Doorways on a Continuous Accessible Path of Travel

Figure 3.15 : Interface for selecting clauses (Ding et al., 2006).



- All Objects
- A continuous accessible path
  - A continuous accessible path to public entrances
  - A continuous accessible path to disabled toilets
  - A continuous accessible path to public spaces
  - A continuous accessible path to public lifts and stairways
- Circulation space
  - Circulation space at doorways
  - Circulation space at disabled toilets
  - Circulation space at seatings
- Doors
- Ramps, Walkways and Landings
- Lifts and Stairs
- Handrails
- Sanitary facilities

Figure 3.16 : Interface for selecting the object types (Ding et al., 2006).

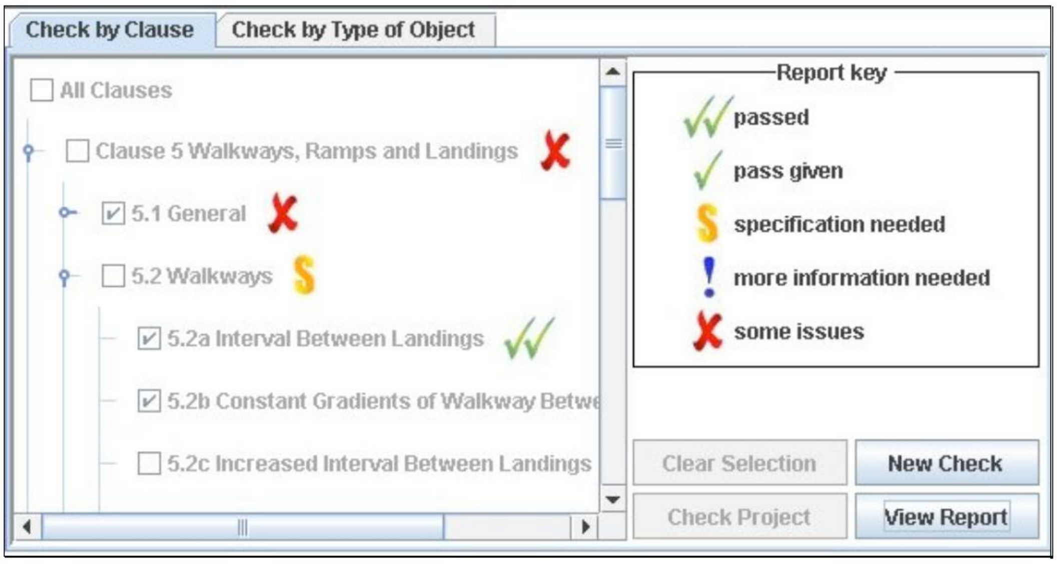
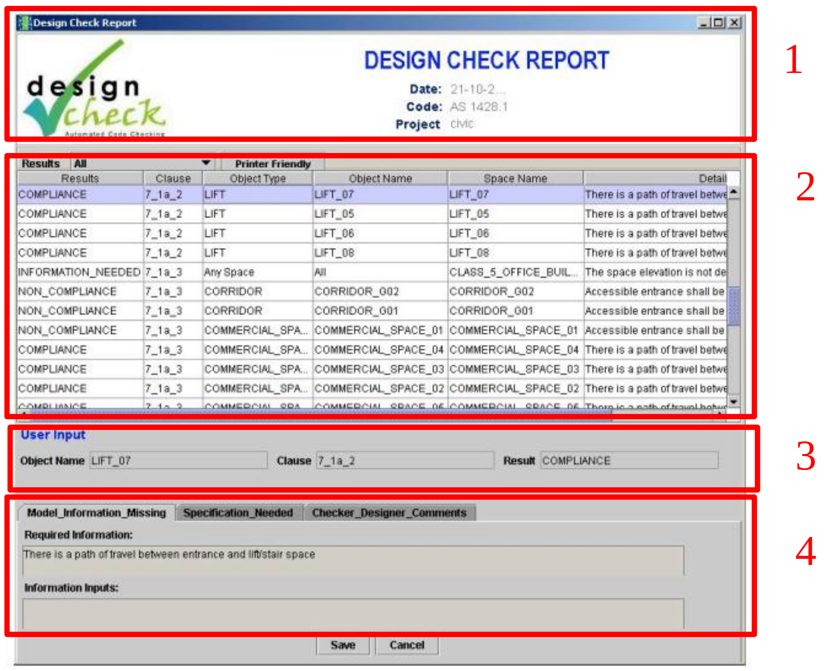


Figure 3.17: Graphical result page (Ding et al., 2006).

The textual report page acts like an additional interactive user interface. Users can update the results by further comments or specification of objects. This feature allows communication between different teams working on the same project. In addition, for the clauses that are not yet translated into the rule schema can be inspected manually and the result can be published here. The report page is composed from four main parts. First one is the top panel that shows the project information, date and code; second one is the selection panel where the results can be inspected; third one is the panel that shows detailed information like object name, object type, the space which the object belongs to, the reason of the failure, clause name and number and the result of the checking; and the last one makes it possible for the user to input further comments or specifications to the results (Figure 3.18).

If the user wants to print the interactive report page, it is formatted for printing and converted into html document. Than user can view it from any web browser, and print it. The formatted document cane be saved into the archives and can be used in comparisons or future reviews.

DesignCheck is an important example of ACCCS. It has some innovative approaches in the implementation of the system. Firstly, it accommodates the users to select the design stage when applying rules to a building model. This allows flexibility in the use of the code checking system which other examples cannot offer. In addition, the pre-implementation specification structure helps in the translation of the written codes into computer interprettable rule schemas. Lack of 3D graphical representation


Design Check Report

DESIGN CHECK REPORT

Date: 21-10-2...
Code: AS 1428.1
Project civic

Results All | Printer Friendly

Results	Clause	Object Type	Object Name	Space Name	Detail
COMPLIANCE	7_1a_2	LIFT	LIFT_07	LIFT_07	There is a path of travel betwe
COMPLIANCE	7_1a_2	LIFT	LIFT_05	LIFT_05	There is a path of travel betwe
COMPLIANCE	7_1a_2	LIFT	LIFT_06	LIFT_06	There is a path of travel betwe
COMPLIANCE	7_1a_2	LIFT	LIFT_08	LIFT_08	There is a path of travel betwe
INFORMATION_NEEDED	7_1a_3	Any Space	All	CLASS_5_OFFICE_BUIL...	The space elevation is not de
NON_COMPLIANCE	7_1a_3	CORRIDOR	CORRIDOR_G02	CORRIDOR_G02	Accessible entrance shall be
NON_COMPLIANCE	7_1a_3	CORRIDOR	CORRIDOR_G01	CORRIDOR_G01	Accessible entrance shall be
NON_COMPLIANCE	7_1a_3	COMMERCIAL_SPA...	COMMERCIAL_SPACE_01	COMMERCIAL_SPACE_01	Accessible entrance shall be
COMPLIANCE	7_1a_3	COMMERCIAL_SPA...	COMMERCIAL_SPACE_04	COMMERCIAL_SPACE_04	There is a path of travel betwe
COMPLIANCE	7_1a_3	COMMERCIAL_SPA...	COMMERCIAL_SPACE_03	COMMERCIAL_SPACE_03	There is a path of travel betwe
COMPLIANCE	7_1a_3	COMMERCIAL_SPA...	COMMERCIAL_SPACE_02	COMMERCIAL_SPACE_02	There is a path of travel betwe
COMPLIANCE	7_1a_3	COMMERCIAL_SPA...	COMMERCIAL_SPACE_06	COMMERCIAL_SPACE_06	There is a path of travel betw

User Input

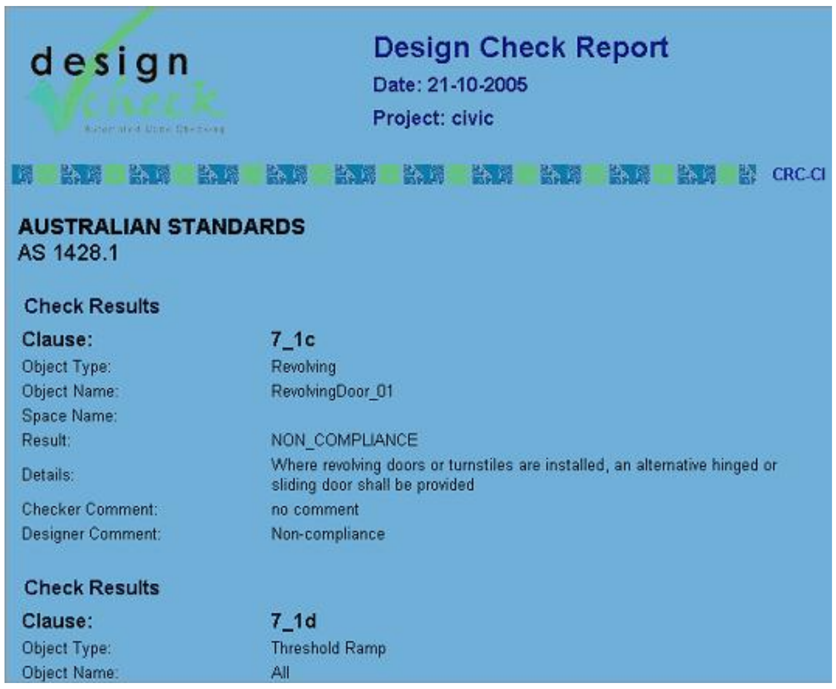
Object Name LIFT_07 Clause 7_1a_2 Result COMPLIANCE

Model_Information_Missing | Specification_Needed | Checker_Designer_Comments

Required Information:
There is a path of travel between entrance and lift/stair space

Information Inputs:

Figure 3.18: Interactive report page (Eastman et al., 2009).


Design Check Report
Date: 21-10-2005
Project: civic

CRC-CI

AUSTRALIAN STANDARDS
AS 1428.1

Check Results

Clause:	7_1c
Object Type:	Revolving
Object Name:	RevolvingDoor_01
Space Name:	
Result:	NON_COMPLIANCE
Details:	Where revolving doors or turnstiles are installed, an alternative hinged or sliding door shall be provided
Checker Comment:	no comment
Designer Comment:	Non-compliance

Check Results

Clause:	7_1d
Object Type:	Threshold Ramp
Object Name:	All

Figure 3.19: Printer friendly report of DesignCheck (Eastman et al., 2009).

ability, in the reporting of the building elements that does not comply with the codes is an important shortcoming. Without 3D graphical display, it is difficult to locate and repair the faulty element from the building model especially if the project is big.

The developers mentions this limitation and aims to add it in the future editions. The final stage of the project is unknown as there is no notice about it, most probably it was a demonstration project to test the capabilities of automated code checking, and after they reached some conclusions, they aborted the project. We have adapted the pre-implementation specification structure to our development process to make the implementation easier as this structure allows to find out required pseudecodes to be implemented in the system.

3.3 SMARTCodes – USA

International Code Council is an United States based association “dedicated to helping the building safety community and construction industry provide safe, sustainable, and affordable construction through the development of codes and standards used in the design, build, and compliance process” (ICC). ICC prepares variety of building codes to be used for residential and commercial buildings. Fifty states and District of Columbia along with many other countries have adopted building codes that are developed by ICC – or I-codes as they named them – at the state or jurisdictional level.

I-codes are model codes. Cities or any other governing bodies are not bounded to the I-codes; in fact, they can change the requirements according to their needs. However, most of the time they prefer not to modify the I-codes and conform to them because it makes the building codes uniform all over the country, which in turn reduces the construction costs.

In 2004, directors of ICC started a new initiative aiming to use object based representation techniques in their building codes and build a successful code compliance checking against these building codes. They have researched the past studies, especially the accomplishments of similar systems implemented in Singapore, Norway etc… The project started in late 2005 and the Board of Directors of ICC determined the following goals for this project;

- Enhanced communication, knowledge and collaboration
- More complete design submittals that are code compliant
- More timely acceptance and approval of plans

- Improved building safety and performance
- Reduced construction costs (ICC, 2007)

Two companies, AEC3 and Digital Alchemy, take part in the development process of SMARTcodes.

International Energy Conservation Code 2006 has been selected as an exemplar building code. When the work for rule interpretation started, ICC already had the codes in XML format, which provided the required framework for further development. At the time of the study, researchers in ICC have noted text processing research which was taking place in other industries like legal, had improvements in document scanning strategies. These strategies were studied and similar approach was devised.

The improvement to this strategy came with the process of transforming ordinary written codes to ‘SMARTcodes’ which is accomplished by using “electronic equivalent of a highlighting pen with different colors used for each concept.” (AEC) This approach is straightforward which takes most of the rule translation burden that most of the other efforts face. In addition, it is reported to be verified that code officials grasp this technique quickly and they do not make mark-up errors. Furthermore, in this method during the rule interpretation, the interpretations are more uniform than in other code compliance checking efforts where they are deeply dependent on the programmer or the code official interpreting the written code. This ensures many code official can work simultaneously on the code interpretation work. From these examples, one can speculate ICC mark-up strategy makes rule interpretation easier, quicker and lower in cost.

SMARTcodes builder, which is a web-based software, is devised to implement above mentioned ICC mark-up strategy to help the translation of written codes to SMARTcodes. It is a specialized XML editor, which is developed to mark-up the written codes with the SMARTcodes strategy.

The mark-up tags in SMARTcodes are limited whereas the properties mentioned within the building codes that must be addressed are very large in numbers. These properties are repeated throughout the building codes and it is critical to have every property “always assigned the same meaning and unit of measurement” (AEC). The accomplish this, all of the properties that can be found in the written building codes from all over the world is collected into a dictionary. International Framework for Dictionaries Library Group develops and maintains this dictionary.

IFD Library Group is an organization that is founded by four organizations, these are; buildingSMART Norway, Construction Specifications Canada, Constructions Specifications Institute (CSI) of USA and STABU from Netherlands. CSI works in cooperation with ICC in this project. IFD is described as "a mechanism that allows for creation of multilingual dictionaries or ontologies" by the IFD Library Group. (IFD Library Group) The aim of the IFD is to provide a reference library to improve interoperability in the AEC industry. "IFD Library provides a flexible and robust method of linking existing databases with construction information to an IFC based Building Information Model". (IFD Library Group)

The advantage of the adoptance of IFD into SMARTcodes is many folds. First, it enables to work side-by-side with Omniclass classification system that has been developed by CSI. It is possible to select a specific property and find it from the matching tables within Omniclass classification system. Secondly, it simplifies working in a multilingual project as IFD links terms and properties independent of any language. Lastly, by using IFD and classification systems it is possible to select codes that are relevant to the issue, which the check is about and filters out any unrelated codes. This can reduce the number of building codes that must be inspected for code checking. The reduction in number of codes results in more efficient code check and this can be applied to not only automatic code compliance checking but also manual code checking. In the case of manual code checking, ICC provides the SMARTcodes system to filter out the check related code criteria and support materials to the users that do not have their building model in BIM. This in turn reduces manual check time as the number of codes that must be inspected reduces.

In the report of AEC (AEC), it is proposed that by converting the written codes into smart codes, automatic derivation of required constraints that are needed for the code checking from the code mark-up became possible. These constraints are implemented according to the IFC constraint model and they called it "requirements model". Requirements model is a "standardized representation" of the rules.

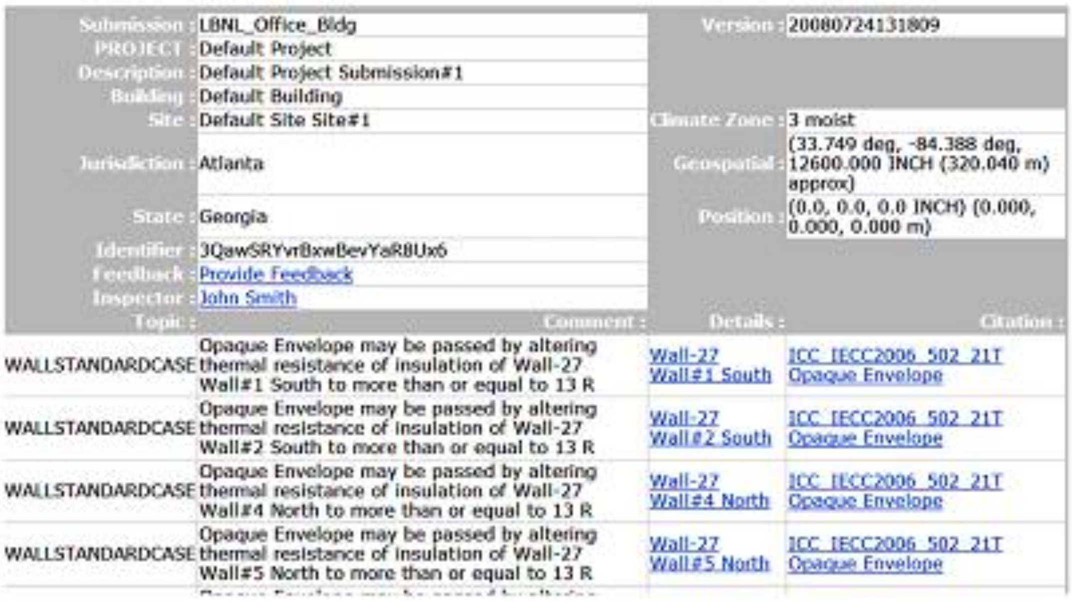
This model makes SMARTcodes approach independent of rule checking system. Any code set that is translated into smart codes can be imported into any rule check software. This is tested for multiple rule-based code checking software and it is verified to be working.

In the SMARTcodes world, building models that are constructed in a BIM software and converted as IFC is called ‘solution model’. The automatic code checking process is actually a process of comparing solutions model with requirements model.

ICC has prepared a demonstration website for user to try out the features of SMARTcodes. The website required the building code, which the compliance checking will be done, a building model, the location of the building model and model checking system. The rule base of the model checking system is consisting of the users were limited to the building models supplied by ICC. These building models were consisted of four US Coast Guard buildings, a prototype office building, Lawrence Berkeley National Laboratory and the National Association of Realtor Headquarters. The code compliance check can only be done regarding some parts of the 2006 IECC. In the future, it is planned for users to load their own building model and select the codes they want to inspect the building models they supplied.

Model Checking Software (MCS) module is responsible for the reporting of the result of the code compliance checking in SMARTcodes. It has various exporting options such as HTML, PDF, RTF, XLS and XML. It has both text based table reporting and graphic based reporting capability. The text based report lists all the outcomes, so users can easily see in which clauses the building model fails the test. Example of table based error report generated by SMC is in Figure 3.20. The text-based report is complemented with a graphical report, which shows the elements from the building model. Graphical report from SMC is in Figure 3.21 and graphical report from AEC3 XABIO is in Figure 3.22.

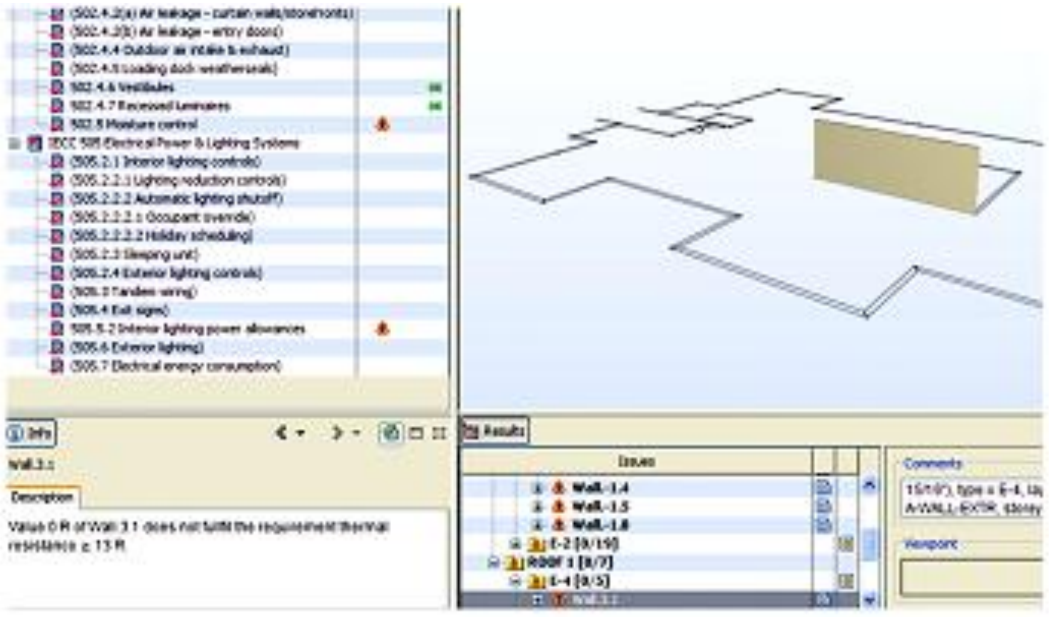
Eastman et. al. (2009) provides an example to error reporting capability of MCS. MCS finds out which elements does not comply with the required codes. MCS retrieves identifier, location, property and geometric shape of a building element from the SMC. The noncomplying building element in this example is Wall 3.1. It is notified in the graphical report of the SMC (Figure 3.21) with a yellow exlamation mark icon, and at the info tab in the description panel, it is informed that “value 0R of Wall 3.1 does not fullfill the requirement thermal resistance ≥ 13R”. In addition to this explanation, in the graphical panel user can see the location of the wall within the building. AEC3 XABIO is reported to include trace back feature which provides a logical step explanation of the noncompliance.



Submission :	LBNL_Office_Bldg	Version :	20080724131809
PROJECT :	Default Project		
Description :	Default Project Submission#1		
Building :	Default Building		
Site :	Default Site Site#1	Climate Zone :	3 moist
Jurisdiction :	Atlanta	Geospatial :	(33.749 deg, -84.388 deg, 12600.000 INCH (320.040 m) approx)
State :	Georgia	Position :	(0.0, 0.0, 0.0 INCH) (0.000, 0.000, 0.000 m)
Identifier :	3QawSRYvrBxwBevYaR8Ux6		
Feedback :	Provide Feedback		
Inspector :	John Smith		

Topic :	Comment :	Details :	Citation :
WALLSTANDARDCASE	Opaque Envelope may be passed by altering thermal resistance of insulation of Wall-27 Wall#1 South to more than or equal to 13 R	Wall-27 Wall#1 South	ICC IECC2006 502 21T Opaque Envelope
WALLSTANDARDCASE	Opaque Envelope may be passed by altering thermal resistance of insulation of Wall-27 Wall#2 South to more than or equal to 13 R	Wall-27 Wall#2 South	ICC IECC2006 502 21T Opaque Envelope
WALLSTANDARDCASE	Opaque Envelope may be passed by altering thermal resistance of insulation of Wall-27 Wall#4 North to more than or equal to 13 R	Wall-27 Wall#4 North	ICC IECC2006 502 21T Opaque Envelope
WALLSTANDARDCASE	Opaque Envelope may be passed by altering thermal resistance of insulation of Wall-27 Wall#5 North to more than or equal to 13 R	Wall-27 Wall#5 North	ICC IECC2006 502 21T Opaque Envelope

Figure 3.20: Text based reporting of SMC (Eastman et al., 2009).



(502.4.2(a) Air leakage – curtain walls/storefronts)
(502.4.2(b) Air leakage – entry doors)
(502.4.4 Outdoor air intake & exhaust)
(502.4.5 Loading dock weatherseals)
502.4.6 Vestibules
502.4.7 Recessed luminaires
502.5 Moisture control
ICC 505 Electrical Power & Lighting Systems
(505.2.1 Interior lighting controls)
(505.2.2.1 Lighting reduction controls)
(505.2.2.2 Automatic lighting shutoff)
(505.2.2.2.1 Occupant override)
(505.2.2.2.2 Holiday scheduling)
(505.2.3 Sleeping unit)
(505.2.4 Exterior lighting controls)
(505.3 Tandem wiring)
(505.4 Exit signs)
505.5.2 Interior lighting power allowances
(505.6 Exterior lighting)
(505.7 Electrical energy consumption)

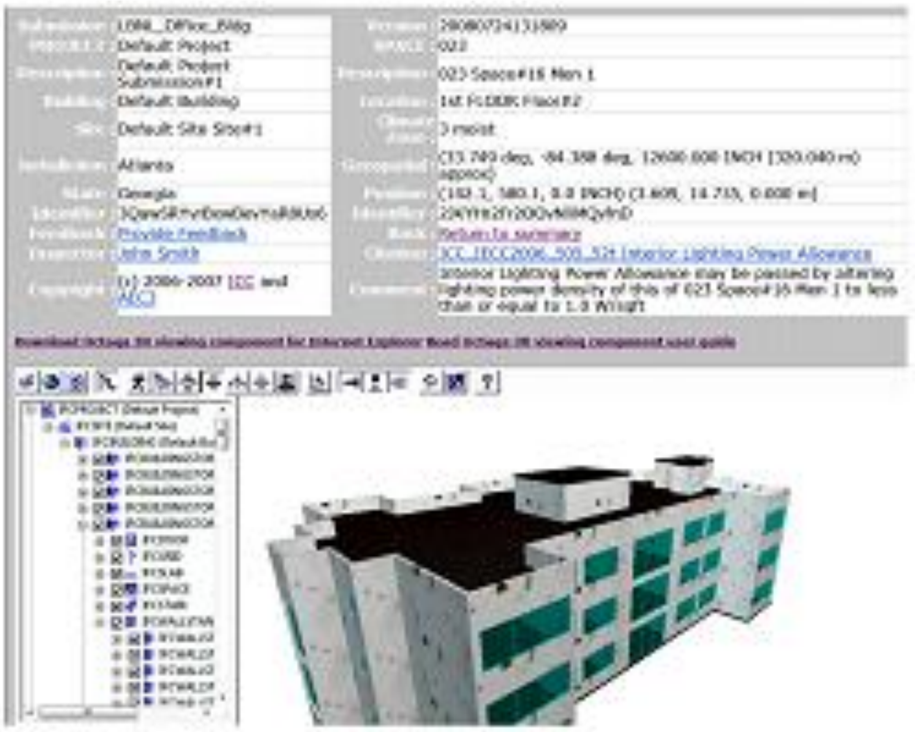
Wall.3.1

Value 0 R of Wall 3.1 does not fulfill the requirement thermal resistance ≥ 13 R

Issues
Wall-1.4
Wall-1.5
Wall-1.8
E-2 [9/19]
ROOF 1 [8/7]
E-4 [8/5]

Figure 3.21: Graphical report of SMC (Eastman et al., 2009).

The ICC SMARTcodes is different from the other automatic code checking examples, as they focused on the rule translation phase more than any other effort. They devised an easy rather more automatic method for rule translation, which makes rule translation effort faster and less costly. One more advantage of their method is that it is also applicable to manual code checking as the codes gets "smarter" it is easier and faster to locate the required information from the building codes. Therefore, while inspecting a building for code compliance, one can filter out the irrelevant clauses and focus only to the properties of the building elements that is required by the codes.



[illegible]	L086_Office_Bldg	[illegible]	20080724131589
[illegible]	Default Project	[illegible]	023
[illegible]	Default Project Submission#1	[illegible]	023 Space#16 Men 1
[illegible]	Default Building	[illegible]	1st FLOOR Floor#2
[illegible]	Default Site Site#1	[illegible]	3 moist
[illegible]	Atlanta	[illegible]	(33.749 deg, -84.388 deg, 12600.000 INCH [320.040 m] approx)
[illegible]	Georgia	[illegible]	(182.1, 580.1, 0.0 INCH) (4.605, 14.735, 0.000 m)
[illegible]	3QaeSRmHxDowDevmsRdUs6	[illegible]	2AYHh2fr20QvN8#QvfnD
[illegible]	Provide Feedback	[illegible]	Return to summary
[illegible]	John Smith	[illegible]	ICC_IECC2006_505_524 Interior Lighting Power Allowance
[illegible]	(c) 2006-2007 ICC and AEC3	[illegible]	Interior Lighting Power Allowance may be passed by altering lighting power density of this of 023 Space#16 Men 1 to less than or equal to 1.0 W/sqft

Download IfcXpg 3D viewing component for Internet Explorer Read IfcXpg 3D viewing component user guide

Figure 3.22: Graphical report of AEC3 (Eastman et al., 2009).

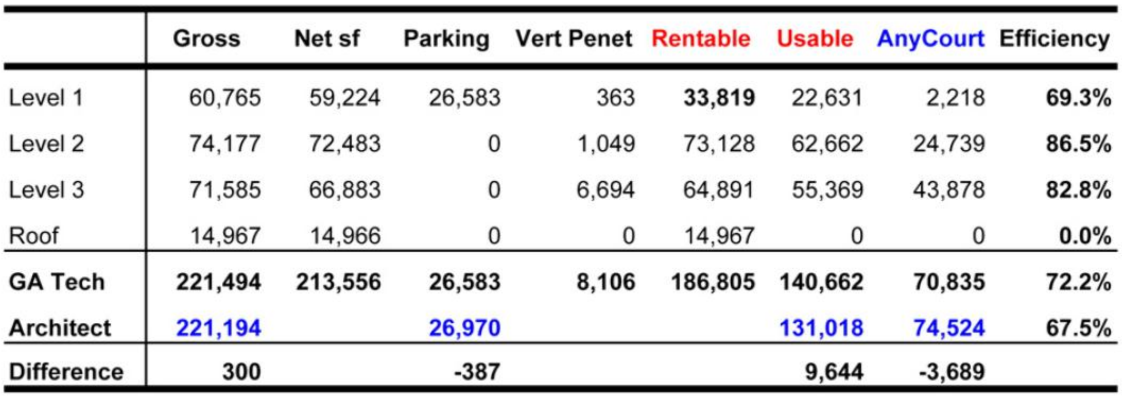
3.4 General Services Administration – USA

General Services Administration (GSA) is an independent agency from United States, and one of its organizational branch, Public Building Safety (PBS) “acquires space on behalf of the federal government through new construction and leasing, and acts as a caretaker for federal properties across the country” (Url-3). Thus, PBS manages huge amount of space all around the country with a budget more than eight billion dollars. In 2003, PBS’s Office of Chief Architect (OCA) started National 3D-4D-BIM Program. Within this program, they wanted to “explore the use of BIM technology throughout a project’s lifecycle in the following areas: spatial program validation, 4D phasing, laser scanning, energy and sustainability, circulation and security validation, and building elements” (Url-4). From these six topics, tools that check compliance against the following three of them, are developed; spatial program validation, energy and sustainability, and circulation and security validation.

PBS is the largest owner of commercial space in the United States. Therefore, they needed a method to measure all of this space in an efficient and consistent way. In the past, GSA required from architects to validate their projects' spatial programs according to GSA approved program, which included requirements like area measurements (net or usable area) and efficiency measurements (fenestration ratio) in the concept stage (Url-5). To check the spatial program architects needed to understand how GSA measures its space from "PBS Business Assignment Guide" and they had to draw 2D polygons on the spaces they need to measure. This technique was reported not to be consistent or efficient. For this reason, GSA started a demonstration project for spatial program validation and starting from 2007, it made compulsory for projects that are done for GSA to submit BIM data for spatial program review.

The process works like this; architects define spaces in their projects with appropriate space tags in the BIM program. The spaces must be separated to represent its functional space even if they are part of a whole. The example in the manual is, if there is a security area within a lobby, lobby and security areas must be modeled as separate non-overlapping spaces. The security area will be labeled with Office tag and the remainder area will be labeled with Building Common tag. After the modeling of the project, it is saved as an IFC file in the BIM program and transferred to the PBS for them to check the spatial program. The calculation method used in this process is ANSI/BOMA area guide with some modifications defined in PBS Business Assignment Guide. After the checking, the result is reported as in Figure 3.23. If this does not comply with PBS values, OCA helps architects to find a remedy. Currently, the BIM analysis does not substitute the required documents submitted in the final stage, it is just used to complement the analysis done in the concept stage.

PBS published a detailed manual for spatial program validation, which has four sections on it; Spatial Program BIM, Spaces and Zones, Building Elements and BIM-analysis rules. In the Spatial Program BIM section the history, objective and process of spatial program validation is discussed along with requirements in the BIM and how the IFC will be submitted. In the Spaces and Zones part, is about how to lay out space boundaries and include the necessary spatial information in the space object (Figure 3.24). Building Elements section, describes the needed building elements for spatial program validation. Finally BIM-analysis Rules introduces the rules, which the PBS will use in spatial program validation. This part also includes special cases like modeling cavity walls etc… This document is complemented with an appendix document (Url-6), which includes how to use certain BIM programs for spatial program validation, and the programs listed are Autodesk Revit, Autodesk ADT, Graphisoft ArchiCAD, Onuma and Bentley Architecture. It introduces the best practices to use the specific software to the architect and is a valuable documentation to maintain consistency in the submitted projects. These documents show that to implement successful rule based tool aimed for the use of the general designer circle, documentation is critical to maintain consistency and to make the process less costly and more efficient.



	Gross	Net sf	Parking	Vert Penet	Rentable	Usable	AnyCourt	Efficiency
Level 1	60,765	59,224	26,583	363	**33,819**	22,631	2,218	**69.3%**
Level 2	74,177	72,483	0	1,049	73,128	62,662	24,739	**86.5%**
Level 3	71,585	66,883	0	6,694	64,891	55,369	43,878	**82.8%**
Roof	14,967	14,966	0	0	14,967	0	0	**0.0%**
GA Tech	**221,494**	**213,556**	**26,583**	**8,106**	**186,805**	**140,662**	**70,835**	**72.2%**
Architect	**221,194**		**26,970**			**131,018**	**74,524**	**67.5%**
Difference	**300**		**-387**			**9,644**	**-3,689**	

Figure 3.23 Results of Spatial Program (Eastman et al, 2009).

GSA is also the largest consumer of energy in USA, consuming nearly 14.9 trillion BTUs per year (Url-7). GSA started a national initiative to reduce the annual energy consumption in its buildings. The aim is to lower energy consumption by 55% in the year 2010 and by the year 2030 all new projects that will be constructed need to be net zero energy buildings. GSA explored techniques for energy modeling using BIM to help to succeed in this ambitious effort as the usage of BIM based energy tools in a project may help an architect by giving more reliable and consistent energy estimates in the early design stage. GSA sums its aim in this study as “to highlight the opportunities and provide best-practice guidance to project teams for achieving improved energy and thermal comfort performance of GSA’s current and future building stock through the use of emerging BIM based energy modeling techniques” (Url-7).

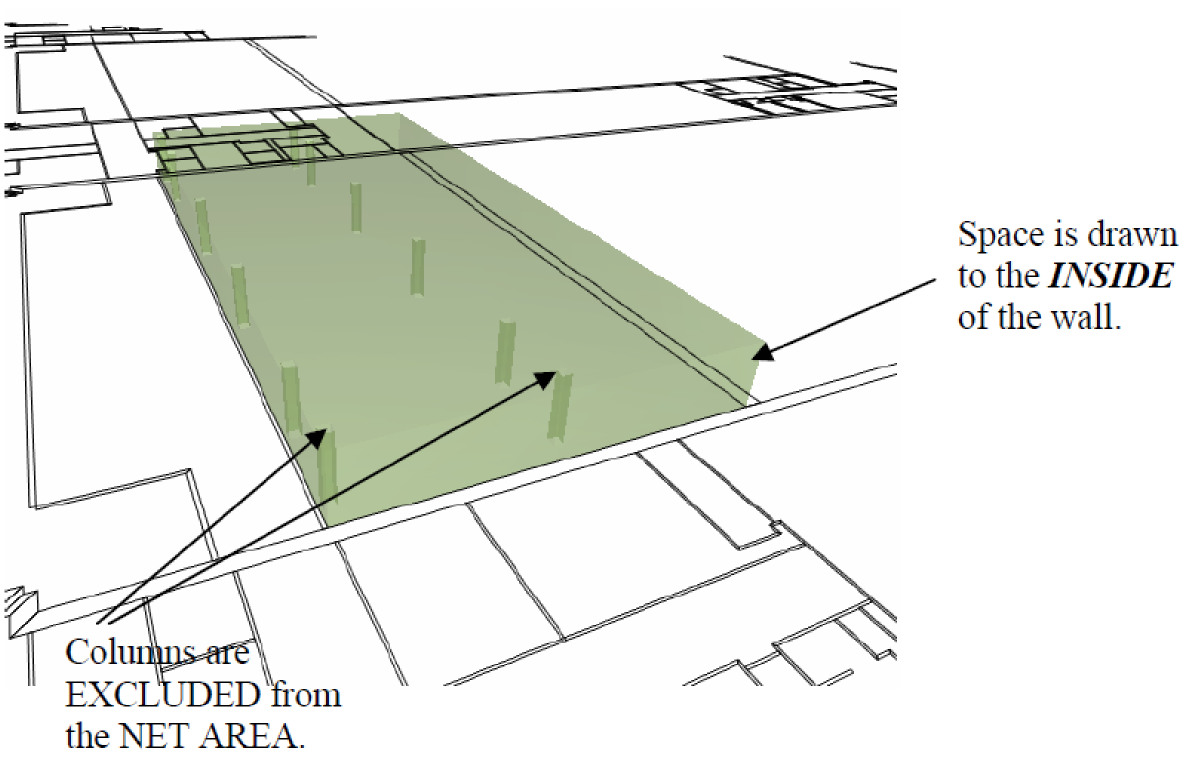

Space is drawn to the *INSIDE* of the wall.
Columns are EXCLUDED from the NET AREA.

Figure 3.24 : Space area highlighted. Space is defined as the area inside the wall excluding columns (Url-5).

Energy consumption in buildings is a result of complex interaction between parameters like outside environment, geometry of a building, envelope of the building, air circulation and equipment loads in a building such as lighting, heat-dissipating machinery etc…

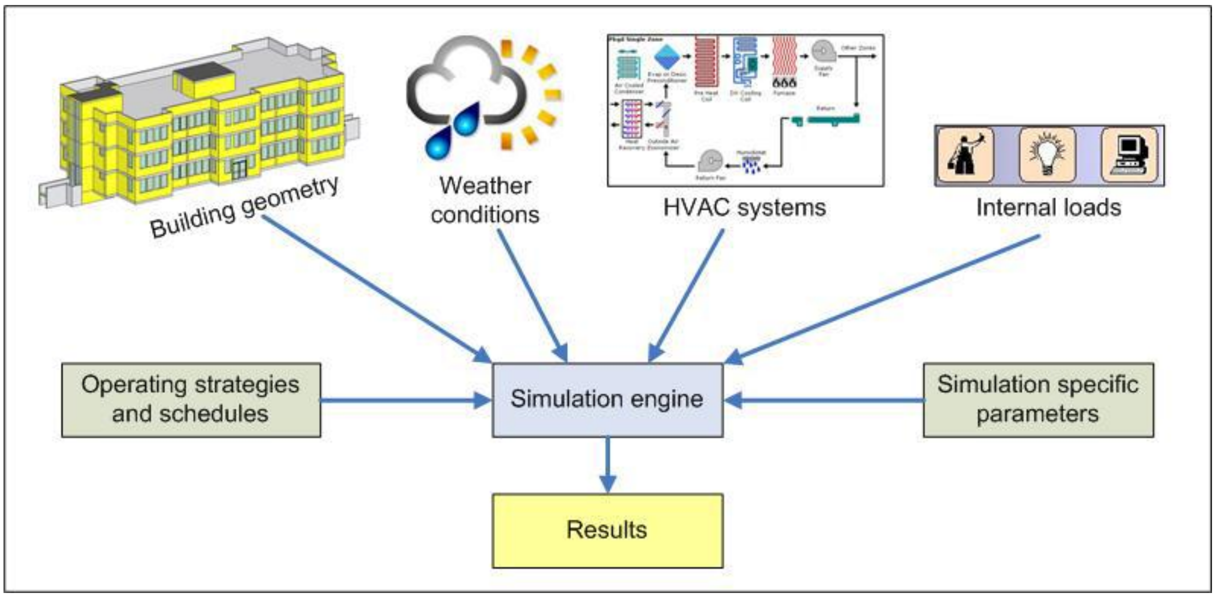


Figure 3.25 : Overview of calculation process of energy consumption (Url-7).

Without the use of ACCCS, it is impossible to manage an environment, which has so many parameters. GSA reported improvement in the performance of its buildings and considerable energy saving is succeeded. GSA team regards this project, successful.

3.5 Statsbygg – Norway

After the success of CORENET, European countries started to search for ways to take advantage of BIM. From those, Nordic countries like Norway, Denmark, and Finland were the leading countries in this effort. In Norway, this effort turned into a project named Byggsok. Byggsok is an e-government system, which has three modules: information, zoning, building (Rooth, 2004).

The information module stores and distributes documents about zoning proposals and building application processes. It has achieved to gather all the information that is published by 433 different local government entities and made these documents accessible online.

The zoning module is for applying for zoning approval process. It also makes communication between authorities and developers possible. It sends zoning approval request to the local government for checking.

The building module is for submitting building plans to the system. The submission is done by using IFC building model.

Byggsok became successful after its release to public in 2003. In 2006 more than 50% zoning applications were done via Byggsok.The work for improving the system is ongoing and the current issue is to add a fourth module which makes automated checks. Another issue they are developing is adding geographic information systems (GIS) with “IFC for GIS” (IFG) data format which is an extension of IFC. It will allow “placing buildings and their services within the context of the local environment” (Rooth, 2004)

Other than Byggsok, some projects are selected to be used in the testing of automated design checking. Firstly, CORENET e-PlanCheck system was tested in “Munkerud housing project for checking building distance/height/utilization and Akershus University Hospital (AHUS) project for checking evacuation related rules” (Eastman et al., 2009). In AHUS project, there were more than 1000 unique rooms, and the automated checking proved useful as one major flaw was discovered: 57 rooms that require water connection had not received water connection (buildingSMART, 2005).

After these two experimentations of automated design checking, one major BIM project named HITOS (Tromso University College) was used to experiment interoperability of different platforms along with rule checking capabilities by Statsbygg.

Two different rule types were examined in HITOS. The first one was about spatial program validation. For this purpose, the project team used dRofus as the rule based system. In the project different teams designed different parts of the building simultaneously, thus it was required that the system allows simultaneous operation. dRofus acts as a database that allows managing of the architectural programs, technical functional requirements and equipment from early stage planning.

dRofus does not require rule interpretation as it is a dedicated application which has the rules preloaded. dRofus reports the required spatial program along with the actual space area. Designers can see the difference and correct any problematic spaces.

The other rule checking system handles the codes for accessibility. To succeed in this, the Statsbygg team used SMC rule based system.

3.6 Conclusion

At this chapter, five different examples of automated code compliance checking systems are reviewed. By looking on these examples we have decided some of the features of our system before starting to the implementation process.

First decision we have done, was to use IFC as the data source for our system. All of the examples mentioned in this chapter uses IFC for this purpose and it became a standard in automated code compliance checking systems.

Then we decided on the audience of the system. In the CORENET example, CORENET was designed for the code officials. It is possible for the designers to check their designs but only if they can connect to the CORENET site. It is not possible to use the system locally. On the other hand, DesignCheck is designed for the architects to check their buildings locally. Our decision was to make a system that can be used for both purposes.

Moreover, we have decided to have two modules information and rule checking like in the case of CORENET and Statsbygg, but we lack the submission module as our system works locally on the computer instead of being located in the web.

Lastly, we have used the pre-implementation specification structure that is used in the DesignCheck in the rule translation stage of the implementation of the system.

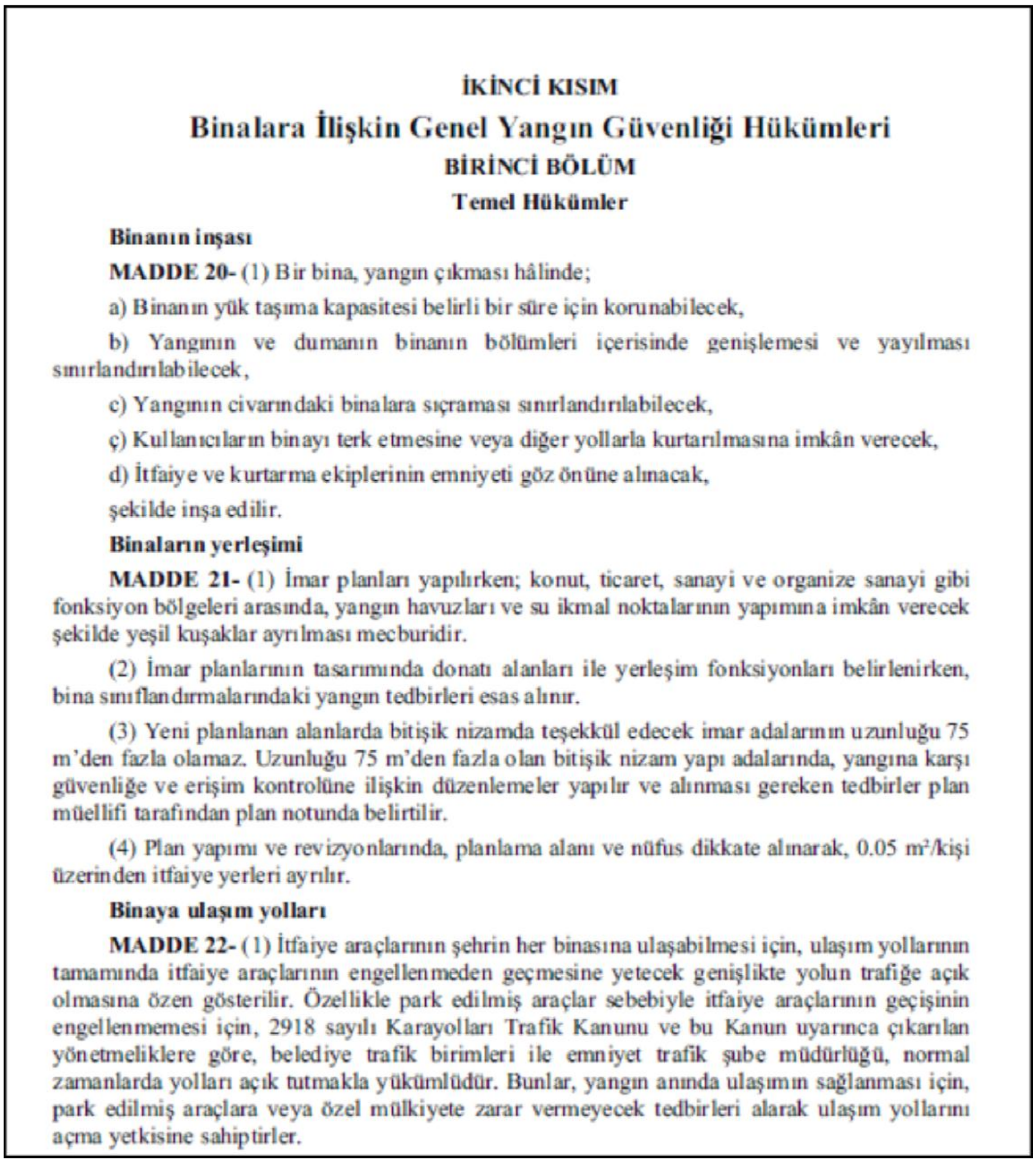
4. **FIRE CODE CHECKER**

Developing a system that can be used in the automated code compliance checking will have certain positive impacts in the AEC industry that is discussed in the introduction chapter. Because of this reason, there have been studies to develop an ACCCS for more than two decades. In this study, a system that can check a building model according to some clauses that is in the Turkish Fire Codes is developed. It is intended that the system will be a foundation to further development that will be continued after this study.

The system has two modules. In the first module, the clauses of the Turkish Fire Codes are presented to the users. Its aim is to ease locating the required information for the user. The clauses are presented in tree structure and users can find out the content of any clause by clicking on the needed tree branch. In the second module, the checking is carried out and the result is presented to the users.

We have selected Turkish Fire Codes as the topic of investigation, thus we named the system Fire Code Checker (FCC). Turkish Fire Codes (TFC) has twelve parts and in those parts there are 171 sections overall. The structure of the TFC is demonstrated in Figure 4.1. Numbering and naming conventions and hierarchy used in it is shown in Table 4.1.

The TFC includes following topics in the parts; 1) General provisions, Building Use and Hazard Classes, 2) General Fire Safety Provisions for Buildings 3) Egress Routes, Egress stairs, Special occasions, 4) Regulations on Parts and Facilities of Buildings, 5) Electrical Installations and Systems, 6) Smoke Control Systems, 7) Fire Retarding Systems, 8) Storage and Handling of Hazardous Substances, 9) Responsibility about the Fire Safety, 10) Articles about the existing buildings, 11)Historic Buildings, 12) Final Remarks. From these parts, we focused on part three, which is about egress routes. We have discussed the reasons for this selection in the first chapter of the thesis.



İKİNCİ KISIM

Binalara İlişkin Genel Yangın Güvenliği Hükümleri

BİRİNCİ BÖLÜM

Temel Hükümler

Binanın inşası

MADDE 20- (1) Bir bina, yangın çıkması hâlinde;

a) Binanın yük taşıma kapasitesi belirli bir süre için korunabilecek,

b) Yangının ve dumanın binanın bölümleri içerisinde genişlemesi ve yayılması sınırlandırılabilecek,

c) Yangının civarındaki binalara sıçraması sınırlandırılabilecek,

ç) Kullanıcıların binayı terk etmesine veya diğer yollarla kurtarılmasına imkân verecek,

d) İtfaiye ve kurtarma ekiplerinin emniyeti göz önüne alınacak,

şekilde inşa edilir.

Binaların yerleşimi

MADDE 21- (1) İmar planları yapılırken; konut, ticaret, sanayi ve organize sanayi gibi fonksiyon bölgeleri arasında, yangın havuzları ve su ikmal noktalarının yapımına imkân verecek şekilde yeşil kuşaklar ayrılması mecburidir.

(2) İmar planlarının tasarımında donatı alanları ile yerleşim fonksiyonları belirlenirken, bina sınıflandırmalarındaki yangın tedbirleri esas alınır.

(3) Yeni planlanan alanlarda bitişik nizamda teşekkül edecek imar adalarının uzunluğu 75 m'den fazla olamaz. Uzunluğu 75 m'den fazla olan bitişik nizam yapı adalarında, yangına karşı güvenliğe ve erişim kontrolüne ilişkin düzenlemeler yapılır ve alınması gereken tedbirler plan müellifi tarafından plan notunda belirtilir.

(4) Plan yapımı ve revizyonlarında, planlama alanı ve nüfus dikkate alınarak, 0.05 m²/kişi üzerinden itfaiye yerleri ayrılır.

Binaya ulaşım yolları

MADDE 22- (1) İtfaiye araçlarının şehrin her binasına ulaşabilmesi için, ulaşım yollarının tamamında itfaiye araçlarının engellenmeden geçmesine yetecek genişlikte yolun trafiğe açık olmasına özen gösterilir. Özellikle park edilmiş araçlar sebebiyle itfaiye araçlarının geçişinin engellenmemesi için, 2918 sayılı Karayolları Trafik Kanunu ve bu Kanun uyarınca çıkarılan yönetmeliklere göre, belediye trafik birimleri ile emniyet trafik şube müdürlüğü, normal zamanlarda yolları açık tutmakla yükümlüdür. Bunlar, yangın anında ulaşımın sağlanması için, park edilmiş araçlara veya özel mülkiyete zarar vermeyecek tedbirleri alarak ulaşım yollarını açma yetkisine sahiptirler.

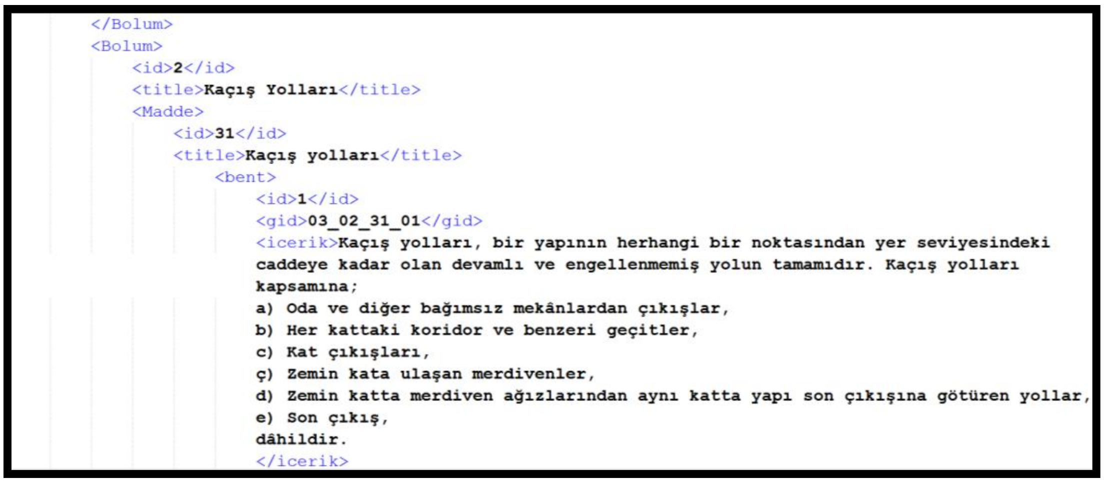
Figure 4.1 : The structure of Turkish Fire Codes.

Information module of FCC requires that written codes of TFC to be converted into electronic format that can be processed within the system. For this reason, we first digitalized the TFC. In this process, we used XML because it is markup language that is both human readable and machine-readable. This feature of XML allows the codes in XML to be processed by computer while it can be still read by humans. In addition, XML files can be easily transferred around computers, thus any changes in codes that are in XML format can be instantaneously sent to the interested parties. Today most of the countries have their regulations in XML format because of the reasons we mentioned. However, Turkey lacks codes in XML format, but with this study, there is now a digitalized copy of the TFC in XML (Figure 4.2).

Table 4.1 Hierarchical Structure of Turkish Fire Codes

Number	Subdivision	Heading
2	Part	Binalara İlişkin Genel Yangın Güvenliği Hükümleri
1	Section	Temel Hükümler
20	Article	Binanın İnşaası
1	Sentence	N/A
a	Clause	N/A

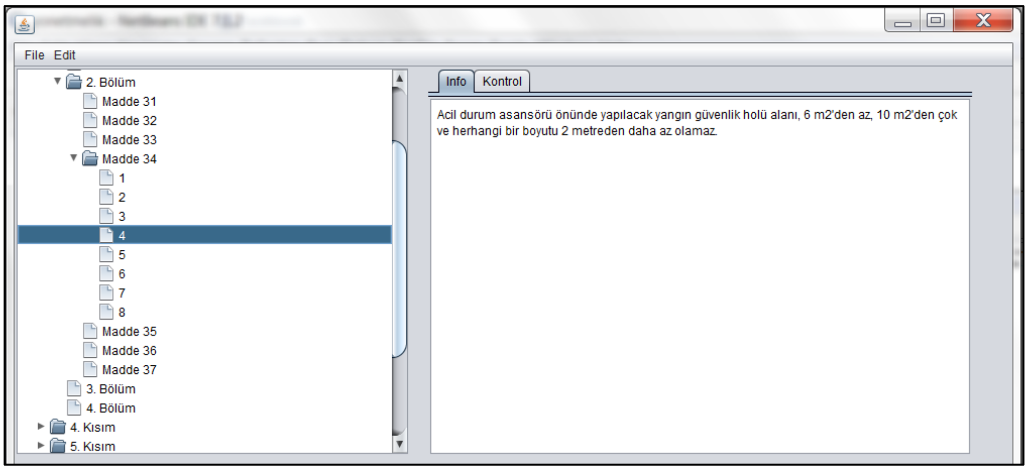
When converting the written codes into XML, the structure of TFC was preserved and parts were stored under <kisim> tag, sections were stored under <bolum> tag, articles were stored under <madde> tag, sentences and clauses were under <icerik> tag. A global ID, which is unique to every sentence, was constructed with the numbers in following structure under the tag <gid>: PartNumber_SectionNumber_ArticleNumber_SentenceNumber. The rules have attributes showing in which design stage they are applicable like <stage=”early”>, <stage=”detailed”> or <stage=”document”>. These are used for selecting the rules according to the design stage.



```
</Bolum>
<Bolum>
    <id>2</id>
    <title>Kaçış Yolları</title>
    <Madde>
        <id>31</id>
        <title>Kaçış yolları</title>
            <bent>
                <id>1</id>
                <gid>03_02_31_01</gid>
                <icerik>Kaçış yolları, bir yapının herhangi bir noktasından yer seviyesindeki
                caddeye kadar olan devamlı ve engellenmemiş yolun tamamıdır. Kaçış yolları
                kapsamına;
                a) Oda ve diğer bağımsız mekânlardan çıkışlar,
                b) Her kattaki koridor ve benzeri geçitler,
                c) Kat çıkışları,
                ç) Zemin kata ulaşan merdivenler,
                d) Zemin katta merdiven ağızlarından aynı katta yapı son çıkışına götüren yollar,
                e) Son çıkış,
                dâhildir.
                </icerik>
```

Figure 4.2 : TFC in XML format.

Designers can select part, section, article or sentence number to access the needed information. In addition, users can search for information within the system, which speeds up information retrieval. The system also has a glossary feature, such that when a user encounters an entity that is unknown to him, he can click and learn its meaning or definition (Figure 4.3).



Acil durum asansörü önünde yapılacak yangın güvenlik holü alanı, 6 m2'den az, 10 m2'den çok ve herhangi bir boyutu 2 metreden daha az olamaz.

Figure 4.3 : Information module of FCC.

For accessing the information stored in the XML file, we used an XML parser. There are different types of XML parsers and each one of it has different strengths and weaknesses. From those we selected Document Object Model (DOM) parser, because it is easier to implement and quicker when working with relatively small files. DOM parser for JAVA is readily included in JAVA Application Programming Interface (API) so it is ready to use.

In the interface of the information module, at the left area, the parts, sections, articles, and sentences of TFC are listed in a tree structure. Users can select one of the sentences to retrieve the information presented at the sentence. When user clicks, to the tree object, DOM parser is called with the number of the sentence in which the user clicked. DOM parser retrieves the content of the sentence from the XML file using that sentence number. The content is displayed at the right text area (Figure 4.4).

Algorithms that were used to check clauses were generated. Each clause is partitioned into entities and requirements. Every entity defined in the codes whether it is a spatial entity such as egress route, circulation or an architectural element such as doors, ramps; it were regarded as an object. Objects have parameters like width of a door and methods for extracting the needed information such as distance between two doors. For example in a clause about properties of escape stairs, it is written "In anyone of

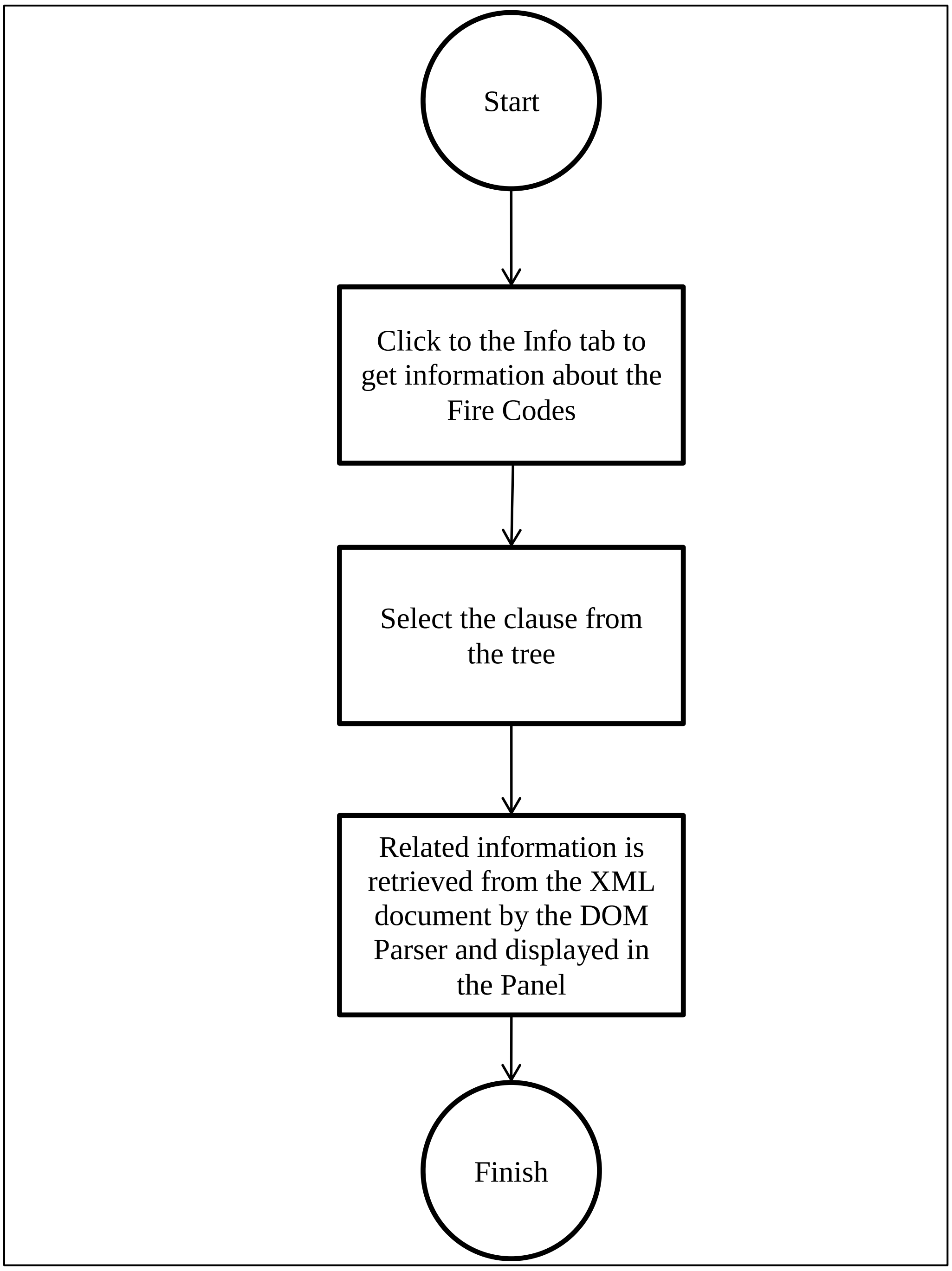
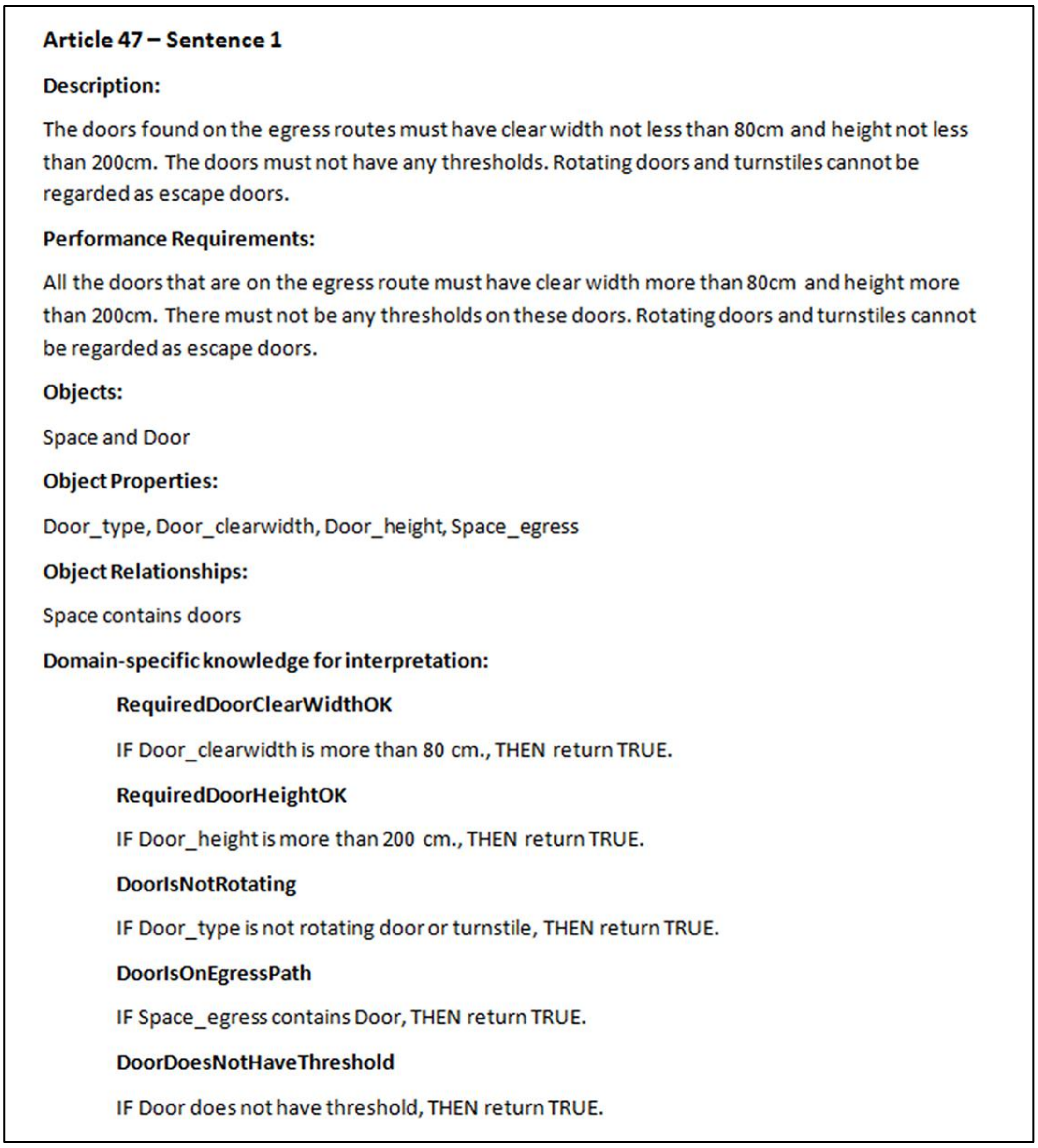

Start
Click to the Info tab to get information about the Fire Codes
Select the clause from the tree
Related information is retrieved from the XML document by the DOM Parser and displayed in the Panel
Finish

Figure 4.4 : Flow diagram of the information module.

the escape stairs it is not possible to have riser height more than 175 mm and tread width less than 250mm". We pick this up by the keyword "escape stairs" and it becomes the entity. Then the two requirements about this entity regarding riser height and tread width are stored in the model. Finally, the conditions required by the clauses are translated into pseudecodes that will be used in the checking of the building model.

To make the translation of the Turkish Fire Codes into pseudecodes easier, we have adapted the pre-implementation specification structure methodology used in the DesignCheck system. In this methodology, each clause have Description, Performance requirements, Object, Properties, Relationships, and Domain specific knowledge for interpretation and by preparing these the translation process becomes more systematic and quicker to implement.



Article 47 – Sentence 1

Description:

The doors found on the egress routes must have clear width not less than 80cm and height not less than 200cm. The doors must not have any thresholds. Rotating doors and turnstiles cannot be regarded as escape doors.

Performance Requirements:

All the doors that are on the egress route must have clear width more than 80cm and height more than 200cm. There must not be any thresholds on these doors. Rotating doors and turnstiles cannot be regarded as escape doors.

Objects:

Space and Door

Object Properties:

Door_type, Door_clearwidth, Door_height, Space_egress

Object Relationships:

Space contains doors

Domain-specific knowledge for interpretation:

RequiredDoorClearWidthOK

IF Door_clearwidth is more than 80 cm., THEN return TRUE.

RequiredDoorHeightOK

IF Door_height is more than 200 cm., THEN return TRUE.

DoorIsNotRotating

IF Door_type is not rotating door or turnstile, THEN return TRUE.

DoorIsOnEgressPath

IF Space_egress contains Door, THEN return TRUE.

DoorDoesNotHaveThreshold

IF Door does not have threshold, THEN return TRUE.

Figure 4.5 : Pre-implementation specification structure applied into Turkish Fire Codes.

As a design feature, we have separated rules and clauses in terms of which stage of the design it is used. For example, the rules about circulation are about early design stage and this is coded internally. For detailed stage of the design, rules like stair width etc… are used. Finally, for specification stage of the design rules like material of the door etc… is used. This separation of clauses of the rules makes code checking on different stages possible.

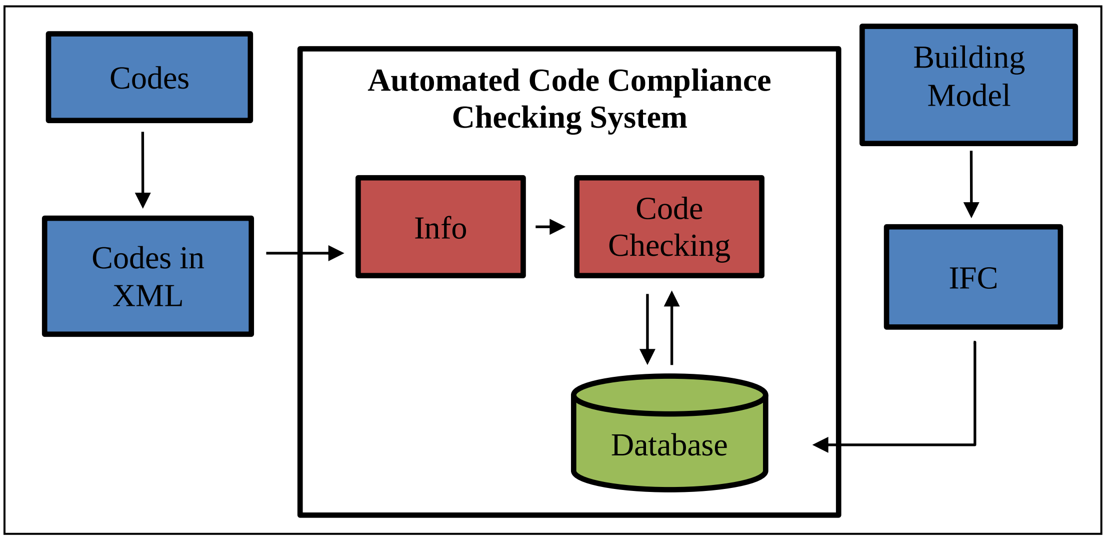


Figure 4.6 : Overall structure of FCC.

In second module, rule checking is performed. This module uses the loaded IFC file to retrieve data needed for rule checking. The IFC file is again parsed with DOM parser and all the related objects are stored into a database. In this step, all the IFC objects are stored in responding database entities (Figure 4.7). In addition, all the properties of the related object are stored in the same structure to the database. For instance, all IFCWall objects are stored in one table, and its properties like NominalLength are stored in another but it keeps its relationship.

We used MySQL to create the database that is used in FCC. It is an open source relational database management system (RDBMS) and it is being widely used. It can be used in a local machine or via internet. For this system, we had setup a local database server. All the data is stored within this database and this data is queried when the checking is done.

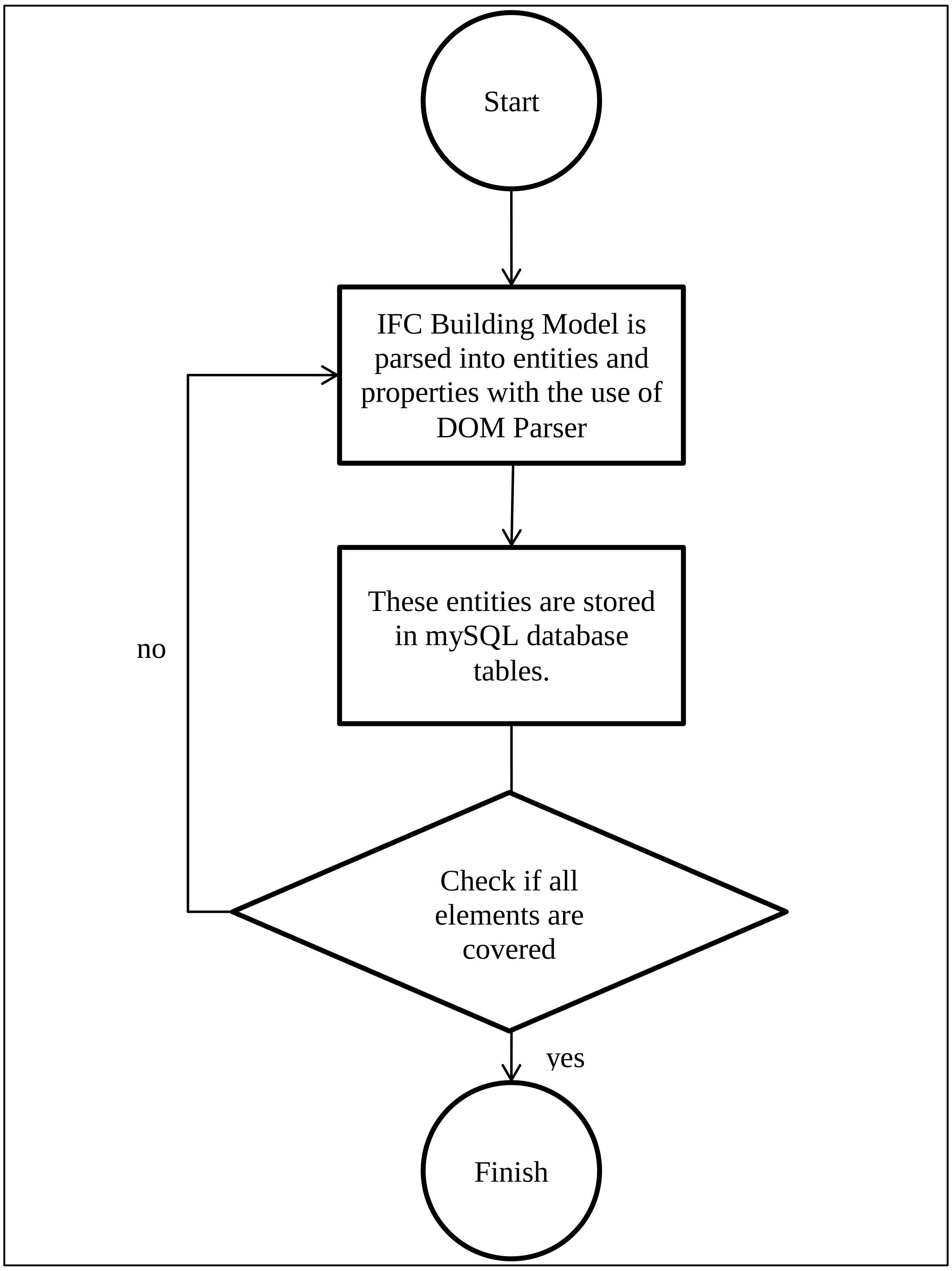

Start
IFC Building Model is parsed into entities and properties with the use of DOM Parser
These entities are stored in mySQL database tables.
no
Check if all elements are covered
yes
Finish

Figure 4.7 : The entities from the IFC building model is stored in the mySQL database.

After the database is populated, the system becomes ready for rule checking. If the user selects a sentence number, the system locates the function to apply the checking. If there is no special selection the system follows full check, which means checking of all sentences. In this case, the functions are called one another to check for the sentence. If an entity complies with the sentence, the result is stored as true in the database, and if it does not comply with the clause, the result becomes false.

For example, article 47, sentence one of the Turkish Fire Codes requires “The doors found on the egress route must have a clear width not less than 80 cm. and height more than 200 cm.”. For this clause, to pass the checking the width and height value of the required doors are retrieved from the database. They are compared with the required value and if they pass the requirements, they pass the clause. If they fail, they fail the clause. And if there is a missing information this is informed to the user with a warning (Figure 4.8).

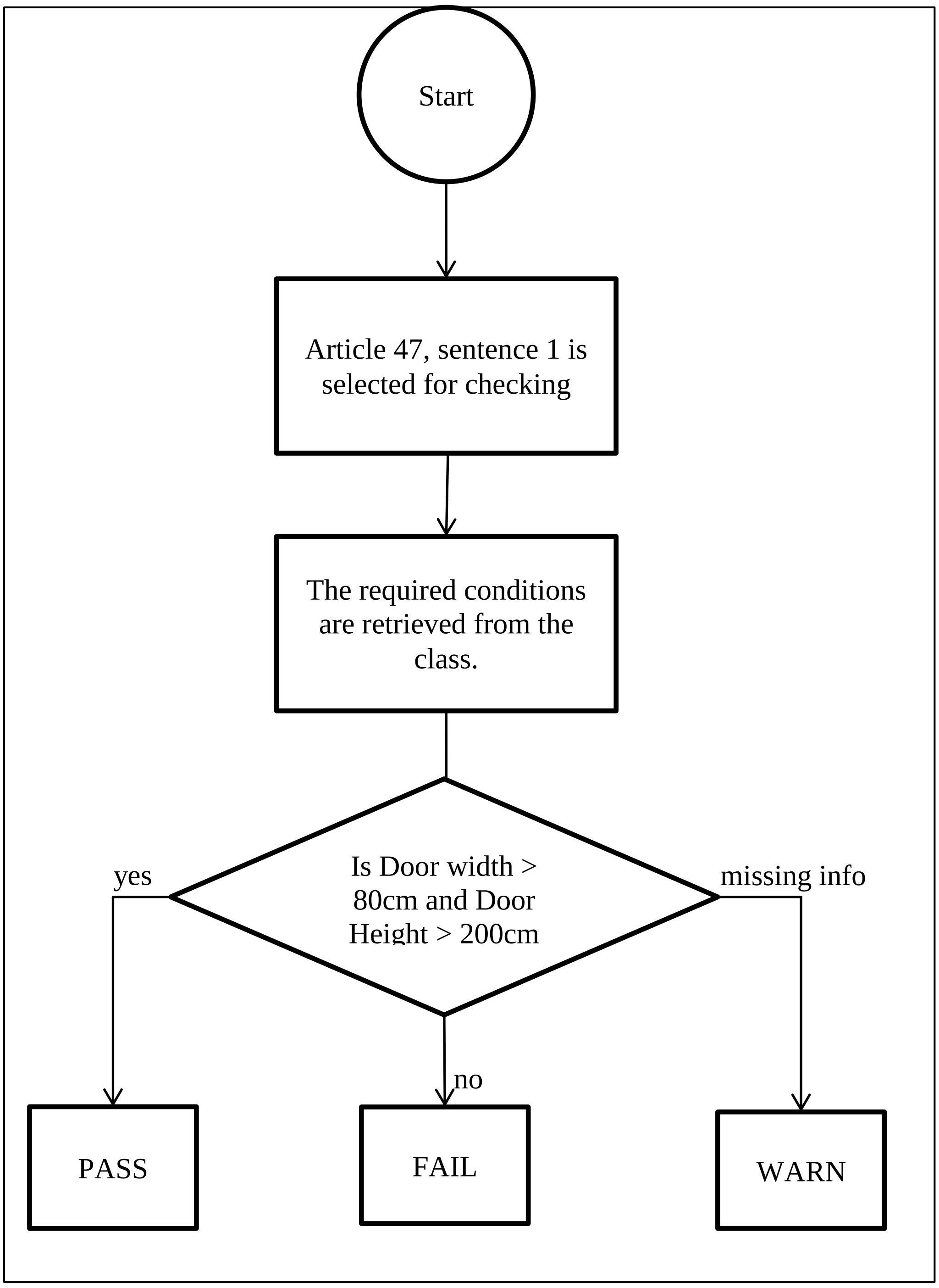
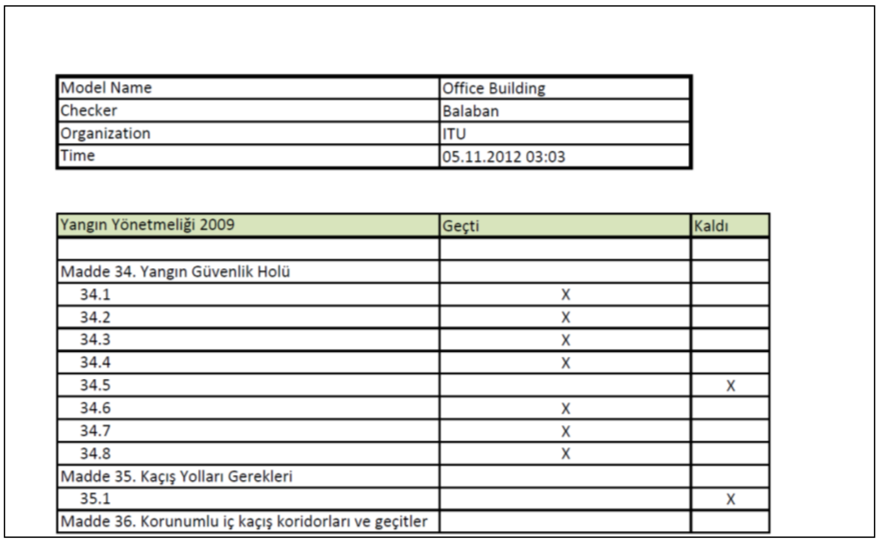

Start
Article 47, sentence 1 is selected for checking
The required conditions are retrieved from the class.
Is Door width > 80cm and Door Height > 200cm
yes
missing info
no
PASS
FAIL
WARN

Figure 4.8 : Flowchart for an example of a checking of one clause.



Model Name	Office Building
Checker	Balaban
Organization	ITU
Time	05.11.2012 03:03

Yangın Yönetmeliği 2009	Geçti	Kaldı
Madde 34. Yangın Güvenlik Holü		
34.1	X	
34.2	X	
34.3	X	
34.4	X	
34.5		X
34.6	X	
34.7	X	
34.8	X	
Madde 35. Kaçış Yolları Gerekleri		
35.1		X
Madde 36. Korunumlu iç kaçış koridorları ve geçitler		

Figure 4.9 : Text based reporting of FCC.

After the rule checking module finishes checking the building model, the results are retrieved from the database and presented to the user. The results can be PASS if all of the codes are satisfied, FAIL if one or more clauses fail and N/A if there is some missing information, which prevents the program to run. If the result is, FAIL then the program reports, which clauses of the code fail, and shows the cause of the problem. The report page will be interactive so that users can check each clause individually. It is possible to export this information in pdf, doc or xml format or to take print outs of the report.

For the implementation of this system, we used Java programming language. Java is an object oriented programming language and it is one of the most used programming languages. There are lots of integrated development environment to work with which is a tools suite that cover a language interpreter, a debugger, database tools etc… We chose Netbeans integrated development environment, which is published by Oracle itself.

For using the system, first step is to import an IFC building model that will be checked against Turkish Fire Codes. In the demonstration of the FCC, one office building example that is supplied by BuildingSmart organization is used (Figure 4.10). The building model of this office building is in IFC format. However, any building model that is modeled in a BIM capable drafting tool with IFC exporting property can be used in the system. When the program starts, the entities in the IFC file is automatically stored in the database. After this step, user can select the clauses from the check boxes and the building model will be checked against selected clauses. The results of the checking can be seen in the panel that is in the right side of the checkboxes (Figure 4.11). Finally, these results can be printed for offline inspection purposes.

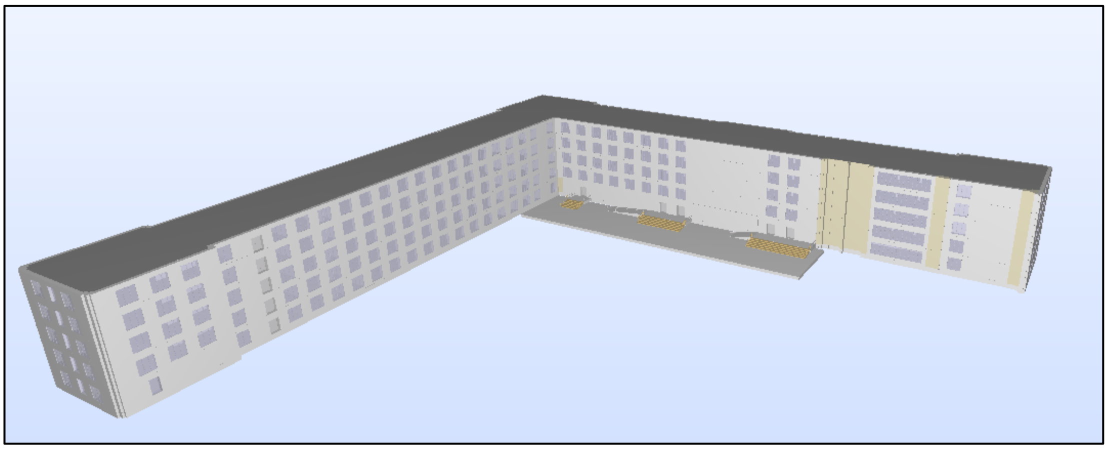
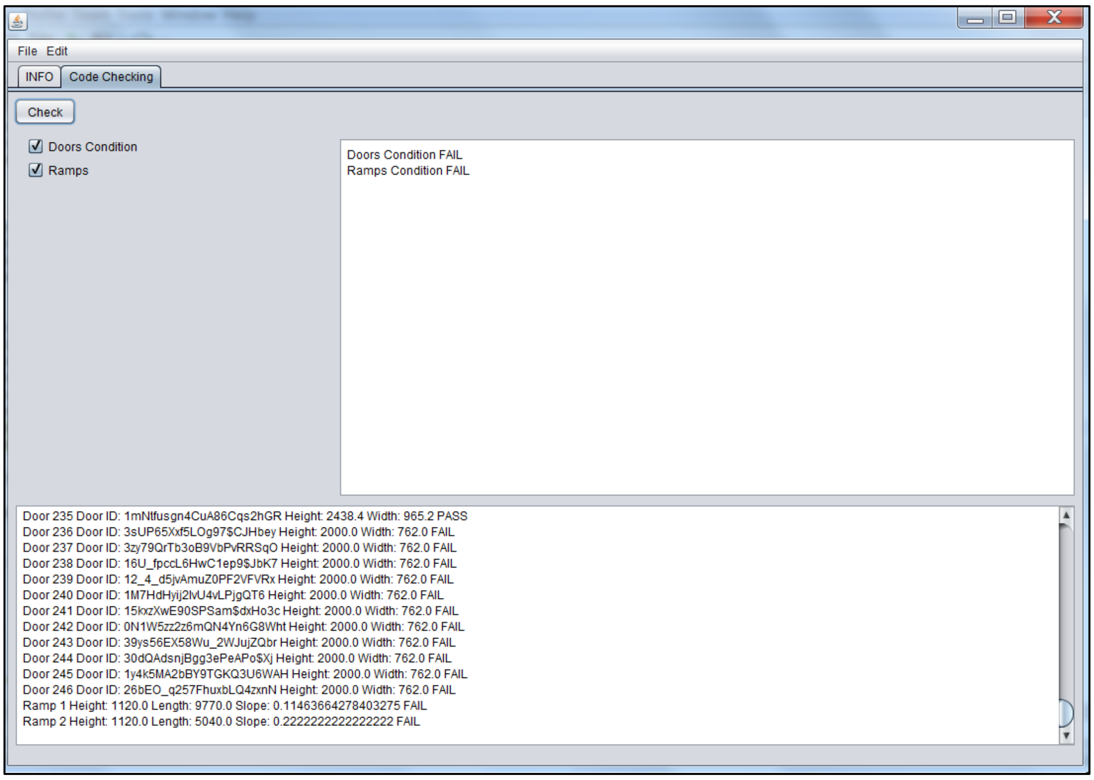


Figure 4.10 : View from the building model used for testing the system.



```
Doors Condition FAIL
Ramps Condition FAIL
```

```
Door 235 Door ID: 1mNtfusgn4CuA86Cqs2hGR Height: 2438.4 Width: 965.2 PASS
Door 236 Door ID: 3sUP65Xxf5LOg97$CJHbey Height: 2000.0 Width: 762.0 FAIL
Door 237 Door ID: 3zy79QrTb3oB9VbPvRRSqO Height: 2000.0 Width: 762.0 FAIL
Door 238 Door ID: 16U_fpccL6HwC1ep9$JbK7 Height: 2000.0 Width: 762.0 FAIL
Door 239 Door ID: 12_4_d5jvAmuZ0PF2VFVRx Height: 2000.0 Width: 762.0 FAIL
Door 240 Door ID: 1M7HdHyij2lvU4vLPjgQT6 Height: 2000.0 Width: 762.0 FAIL
Door 241 Door ID: 15kxzXwE90SPSam$dxHo3c Height: 2000.0 Width: 762.0 FAIL
Door 242 Door ID: 0N1W5zz2z6mQN4Yn6G8Wht Height: 2000.0 Width: 762.0 FAIL
Door 243 Door ID: 39ys56EX58Wu_2WJujZQbr Height: 2000.0 Width: 762.0 FAIL
Door 244 Door ID: 30dQAdsnjBgg3ePeAPo$Xj Height: 2000.0 Width: 762.0 FAIL
Door 245 Door ID: 1y4k5MA2bBY9TGKQ3U6WAH Height: 2000.0 Width: 762.0 FAIL
Door 246 Door ID: 26bEO_q257FhuxbLQ4zxnN Height: 2000.0 Width: 762.0 FAIL
Ramp 1 Height: 1120.0 Length: 9770.0 Slope: 0.11463664278403275 FAIL
Ramp 2 Height: 1120.0 Length: 5040.0 Slope: 0.2222222222222222 FAIL
```

Figure 4.11 : Results of the checking.

The current implementation of the system is just a demonstration to show the main working principles of the systems. There is still work to do even to fully cover egress routes part of the TFC.

5. CONCLUSION

This thesis underlined the need for a successful ACCCS. ACCCSs make code checking faster, with less error, and is more cost efficient than the traditional way. In addition, ACCCSs make the code checking process more transparent, which in turn lessens the corruption that is prevalent in the process of getting approvals especially in developing countries.

5.1 Potential users

ACCCS are developed to serve to two kinds of users. The obvious one is the code compliance checking authorities. They need ACCCS to check projects before giving authorization to construct. With the help of ACCCS, this process will be faster, cheaper and with fewer errors.

Another advantage of using ACCCS for the code checkers is that because the code officials will be part of the development of the ACCCS, the written codes will be more structured and less ambiguous as the code officials will plan and write the codes to suit the rule translation stage.

The second user type will be architects and designers. In this case, the automated code compliance systems will be used for checking the project according to the codes that it must satisfy. Therefore, the design stage of the projects will considerably get shorter and there will be fewer errors, which may require costly after construction modifications.

One more usage for these systems is controlling the design according to the project specifications. Today this function is used with limitations in some systems, especially for checking spatial layout requirements.

This side of the ACCCSs serving diverse users, needs different capabilities from the system. For code officials, the ability to easily modify the stored clauses or enter new clauses is needed to keep up with the frequently changing codes. Whereas for the designers graphical reporting and selecting the required clause group for the design stage functions are needed to locate the problematic building element.

The only working ACCCS, CORENET only works for the code checking authority. Architects working with CORENET must check its projects by uploading the files into the server and get the results on the server. This requires frequent file transfers between the server and the designer. It is not the ideal usage of ACCCSs, if we think from designers' side. Thus, a successful automated code compliance checking system will be the only tool for architects to automatically check his designs.

5.2 Future work

As mentioned previously, the successful ACCCS are many years away. Our work too, is in its development stage. There are many steps to succeed to develop a fully automatic code compliance checking system.

In our work the rule base does not cover all of the fire codes, just some parts about the egress routes are covered. First, we must cover the whole of the clauses in the fire codes. This has its difficulties, as some clauses require some intrinsic information, which is difficult if not impossible to acquire from IFC.

Our system lacks graphical reporting features which is required for easily locating the problematic building element. For now the system only supports textual reporting in which the element is referred by its number. This makes finding the element harder.

One another thing to do is to follow IFD initiative and build Turkish dictionary in parallel with the IFD effort. This is important as the tools that are used in design firms are global and it is important to map Turkish terms with the global ones.

REFERENCES

AEC (n.d.). AEC - International Code Council. Date Retrieved: 15.04.2012, Address: http://www.aec3.com/en/5_013_ICC.htm.

buildingSMART. (2002). buildingSMART Case Studies - The CORENET project in Singapore. Date Retrieved: 15.04.2012, Address http://www.civ.utoronto.ca/sect/coneng/i2c/D-Base/On the web-short/Case studies/Singapore-CORENET.pdf.

Ding, L., Drogemuller, R., Jupp, J., Rosenman, M. A., and Gero, J. S. (2004). Automated code checking, *CRC CI International Conference 2004*, Gold Coast.

Ding, L., Drogemuller, R., Rosenman, M., Marchant, D. and Gero, J. (2006). Automating code checking for building designs – Designcheck, *Clients Driving Innovation: Moving Ideas into Practice*, March 12-14.

Eastman, C., Lee, J-m., Jeong, Y-s., and Lee, J-k. (2009). Automatic rule-based checking of building designs, *Automation in Construction*, **18**(8) 1011-1033.

Fatt, C. T. (2005). CORENET e-Plan Check System: The Pillar of IFC Implementation in Singapore. Retrieved from http://www.iai.no/2005_buildingSMART_oslo/Session01/eSubmissioe_eplancheck_Singapore_Case.pdf

Fenves, S.J. (1966). Tabular decision logic for structural design, *Journal of Structural Engineering 92* (ST6) 473–490

Fenves, S.J., Wright, R.N. (1977).The representation and use of design specifications, Technical Note 940, NBS, Washington, DC.

Fenves, S.J. (1982). A methodology for the evaluation of designs for Standards conformance, *PhD Thesis*, Stanford University.

Han, C., Kunz, J. and Law, K. (1997). Making automated building code checking a reality. *Facility Management Journal*, 22-28.

Han, C., Kunz, J., and Law, K, (1999) Building design services in a distributed architecture, *Journal of Computing in Civil Engineering*, ASCE 13 (1) 12–22.

ICC. (2007). SMARTcodes. Date Retrieved: 15.04.2012, Address: http://www.stanford.edu/group/narratives/classes/0809/CEE215/ReferenceLibrary/International%20Code%20Council%20(ICC)/SMARTcodes%20fact%20sheet%2011-01-07.pdf.

ICC. (n.d.). About ICC. Date Retrieved: 15.04.2012, Address: http://www. iccsafe.org /AboutICC/Pages/default.aspx.

IFD Library Group. (n.d.). Overview - IFDWeb. Date Retrieved: 15.04.2012, Address: http://www.ifdlibrary.org/index.php?title=Overview# Who_is_IFD_Library.3F.

Khemlani, L. (2005). CORENET e-PlanCheck: Singapore's Automated Code Checking System. Date Retrieved: 15.04.2012, Address: http://www.aecbytes.com/buildingthefuture/2005/CORENETePlanCheck.htmh.

Lau, G. T. (2004). A Comparative Analysis Framework for Semi-structured Documents, with Applications to Government Regulations, *PhD Thesis*, Stanford University.

Liebich, T., Wix, J., Forester, J. and Zhong, Q. (2002). Speeding-up Building Plan Approvals plan checking based on IFC, *European Conferences on Product and Process Modelling (ECPPM) 2002*, Portoroz, Slovenia. 1-6.

Nikkhah, R. (2003). Laing sues over disabled access, *Building Magazine*. date Retrieved: 30.04.2012, Address: http://www.building.co.uk/news/ laing-sues-over-disabled-access/1028407.article.

novaCITYNETS (2012). FORNAX plan check expert. Date Retrieved: 15.04.2012, Address: http://www.novacitynets.com/fornax/brochure/ en_fornax.pdf.

Raskopf, R. L. and Bender, D. (2003) "Cross-Border Data: Information Transfer Restrictions Pose a Global Challenge," *New York Law Journal*, July 29, 2003.

Rooth, O. (2005). Byggsok: plan – informasjon – bygning, *IAI Conference,* Oslo.

Solihin, **W.** (2005). Implementing IFC : Automatic Code Checking Automated checking of building design against conformance to. Date Retrieved: 15.05.2012, Address: http://www.iai.no/2005_buildingSMART_oslo /Session 07/novaCITYNETS Oslo IAI presentation May 2005.pdf.

Sing, T. F. and Zhong, Q. (2001). COnstruction and Real Estate NETwork, *Facilities*, 19(11), 419-427.

Tan, X., Hammad, A. and Fazio, P. (2010). Automated Code Compliance Checking for Building Envelope Design. **24**(2).

Young, N. W., Jones, S. A. and Berstein, H. M. (2007). Interoperability in Construction Industry, McGraw Hill Construction, New York, NY.

Yurchyshyna, A. and Zarli A. (2009). An Ontology-based Approach for Formalisation and Semantic Organisation of Conformance Requirements in Construction**,** *International Research Journal "Automation in Construction"*, **18**(8) 1084-1098.

Zou, P.X.W. (2006). Strategies for Minimizing Corruption in the Construction Industry in China, Journal of Construction in Developing Countries, **11**(2).

Xu, R., Solihin, W. and Huang, Z. (2004)**.** A Code Checking Object Model for Computer Aided Building Design**.** Date retrieved: 15.05.2012 Address: ftp://ftp.mpi-sb.mpg.de/pub/conferences/vmv04/submissions /101/paper.pdf.

Url-1 retrieved from http://www.doingbusiness.org/data/exploreeconomies/turkey/ #dealing-with-construction-permits, date retrieved 15.04.2012.

Url-2 retrieved from http://www.doingbusiness.org/data/exploreeconomies/singa pore/#dealing-with-construction-permits, date retrieved 15.04.2012.

Url-3 retrieved from http://www.gsa.gov/portal/content/104444 Public Buildings Service, date retrieved 15.04.2012.

Url-4 http://www.gsa.gov/portal/content/105075 3D-4D Building Information Mo-deling, date retrieved 15.04.2012.

Url-5 http://www.gsa.gov/graphics/pbs/BIM_Guide_Series_02_v096.pdf GSA Bu-ilding Information Modeling Series 02-Spatial Program Validation DRAFT, date retrieved 15.04.2012.

Url-6 http://www.gsa.gov/graphics/pbs/GSA-BIM-02-Appendix-V09.pdf, date re-trieved 15.04.2012.

Url-7 http://www.gsa.gov/graphics/pbs/GSA-BIM-Guide-Series.pdf, date retrieved 15.04.2012.

Url-8 http://www.corenet-ess.gov.sg/ess/, date retrieved 15.04.2012.

CURRICULUM VITAE

Name Surname:	**Özgün Balaban**
Place and Date of Birth:	**18.06.1981 - İskenderun**
Address:	**Duatepe Mah. Şahadet sok. 40/5 Feriköy Şişli İstanbul**
E-Mail:	**ozgunbalaban@gmail.com**
B.Sc.:	**Electrical and Electronics Engineering**

Professional Experience and Rewards:

1st place in Cyberpark Innovative Project Competition 2004

List of Publications and Patents:

- **Güngör, Ö., Çağdaş, G., Balaban, Ö**. (2011) Genetik Algoritmaya Dayalı Kitlesel Bireyselleştirme Amaçlı Konut Tasarım Modeli, *Mimarlıkta Sayısal Tasarım 2011 Ulusal Sempozyum Bildiri Kitabı*, May 23, 2011.
- **Güngör, Ö., Çağdaş, G., Balaban, Ö.** (2011) A Mass Customization Oriented Housing Design Model Based on Genetic Algorithm, *Respecting fragile places, 29th eCAADe International Conference*, September 21, 2011.

PUBLICATIONS/PRESENTATIONS ON THE THESIS

- **Balaban, Ö., Yağmur Kilimci, E. S., Çağdaş, G.** (2012) Otomatik Yönetmelik Kontrolü Modeli, *Mimarlıkta Sayısal Tasarım 2012 Ulusal Sempozyum Bildiri Kitabı,* May 18, 2012.
- **Balaban, Ö., Yağmur Kilimci, E. S., Çağdaş, G.** (2012) Automated Code Compliance Checking Model for Fire Egress Codes, *30th eCAADe International Conference,* September 13, 2012