Designing and Developing a Computer Program to Assess Building Code Requirements

Nicholas Ogilvie-Thompson and James P. Skon, Ph.D.
Kenyon College Summer Science 2016

Abstract

When designing and constructing a building, careful attention must be paid so that the building meets all requisite code requirements. Building codes differ country-to-country, state-to-state, and sometimes even city-to-city, but no matter where you are, the penalties for failing to be code-compliant are stiff, ranging from hefty fines to necessitating reconstruction before the building can be used. Because of this, architects and architectural engineers take great pains ahead of construction to make sure their designs are up to code. However, there currently is no simple method for an architectural engineer to determine a building's code requirements other than the tedious process of going over the code manually. In order to help ease these issues, I worked to create an architectural planning system that can represent architectural knowledge in such a way that it can ask the questions necessary to determine the code requirements a building would have to meet.

Background

Why do this?

- Code assessment is expensive
  - Involves architectural engineer and lawyer
- Mistakes are extremely costly
- On average buildings are "overbuilt" by ~5-10%

Project Goals

- Systematize the logic of the building code
- Use that to create a proof-of-concept of a program to determine a building's code requirements
- Focus on usability
  - Ask users as few questions as necessary

The Building Code

[F] TABLE 307.7(1)
MAXIMUM ALLOWABLE QUANTITY PER CONTROL AREA OF HAZARDOUS MATERIALS POSING A PHYSICAL HAZARD[a, j, m, n]

MATERIAL	CLASS	GROUP WHEN THE MAXIMUM ALLOWABLE QUANTITY IS EXCEEDED	STORAGE[b]			USE-CLOSED SYSTEMS[b]			USE-OPEN SYSTEMS[b]	
			Solid pounds (cubic feet)	Liquid gallons (pounds)	Gas (cubic feet at NTP)	Solid pounds (cubic feet)	Liquid gallons (pounds)	Gas (cubic feet at NTP)	Solid pounds (cubic feet)	Liquid gallons (pounds)
Combustible liquid[c, i]	II IIIA IIIB	H-2 or H-3 H-2 or H-3 N/A	N/A	120[d, e] 330[d, e] 13,200[e, f]	N/A	N/A	120[d] 330[d] 13,200[f]	N/A	N/A	30[d] 80[d] 3,300[f]
Combustible fiber	Loose Baled	H-3	(100) (1,000)	N/A	N/A	(100) (1,000)	N/A	N/A	(20) (200)	N/A
Consumer fireworks (Class C, Common)	1.4G	H-3	0	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Cryogenics flammable	N/A	H-2	N/A	45[d]	N/A	N/A	45[d]	N/A	N/A	10[d]
Cryogenics, oxidizing	N/A	H-3	N/A	45[d]	N/A	N/A	45[d]	N/A	N/A	10[d]
Explosives	Division 1.1 Division 1.2 Division 1.3 Division 1.4 Division 1.4G Division 1.5 Division 1.6	H-1 H-1 H-1 or 2 H-3 H-3 H-1 H-1	1[e, g] 1[e, g] 5[e, g] 50[e, g] 0 1[e, g] 1[d, e, g]	(1)[e, g] (1)[e, g] (5)[e, g] (50)[e, g] N/A (1)[e, g] N/A	N/A N/A N/A N/A N/A N/A N/A	0.25[g] 0.25[g] 1[g] 50[g] N/A 0.25[g] N/A	(0.25)[g] (0.25)[g] (1)[g] (50)[g] N/A (0.25)[g] N/A	N/A N/A N/A N/A N/A N/A N/A	0.25[g] 0.25[g] 1[g] N/A N/A 0.25[g] N/A	(0.25)[g] (0.25)[g] (1)[g] N/A N/A (0.25)[g] N/A
Flammable gas	Gaseous liquefied	H-2	N/A	N/A 30[d, e]	1,000[d, e] N/A	N/A	N/A 30[d, e]	1,000[d, e] N/A	N/A	N/A
Flammable liquid[c]	1A 1B and 1C	H-2 or H-3	N/A	30[d, e] 120[e, e]	N/A	N/A	30[d] 120[d]	N/A	N/A	10[d] 30[d]
Combination flammable liquid (1A, 1B, 1C)	N/A	H-2 or H-3	N/A	120[d, e, h]	N/A	N/A	120[d, h]	N/A	N/A	30[d, h]
Flammable solid	N/A	H-3	125[d, e]	N/A	N/A	125[d]	N/A	N/A	25[d]	N/A
Organic peroxide	UD I II III IV V	H-1 H-2 H-3 H-3 N/A N/A	1[e, g] 5[e, g] 50[d, e] 125[d, e] NL NL	(1)[e, g] (5)[d, e] (50)[d, e] (125)[d, e] NL NL	N/A N/A N/A N/A N/A N/A	0.25[g] 1[d] 50[d] 125[d] N/L N/L	(0.25)[g] (1) (50)[d] (125)[d] N/L N/L	N/A N/A N/A N/A N/A N/A	0.25[g] 1[d] 10[d] 25[d] NL NL	(0.25)[g] (1)[d] (10)[d] (25)[d] NL NL
Oxidizer	4 3[k] 2 1	H-1 H-2 or H-3 H-3 H-3	1[e, g] 10[d, e] 250[d, e] 4,000[e, f]	(1)[e, g] (10)[d, e] (250)[d, e] (4,000)[e, f]	N/A N/A N/A N/A	0.25[g] 2[d] 250[d] 4,000[f]	(0.25)[g] (2)[d] (250)[d] (4,000)[f]	N/A N/A N/A N/A	0.25[g] 2[d] 50[d] 1,000[f]	(0.25)[g] (2)[d] (50)[d] (1,000)[f]
Oxidizing gas	Gaseous liquefied	H-3	N/A N/A	N/A 15[d, e]	1,500[d, e] N/A	N/A N/A	N/A 15[d, e]	1,500[d, e] N/A	N/A N/A	N/A N/A

SECTION 305
EDUCATIONAL GROUP E

305.1 Educational Group E. Educational Group E occupancy includes, among others, the use of a building or structure, or a portion thereof, by six or more persons at any one time for educational purposes through the 12th grade. Religious educational rooms and religious auditoriums, which are accessory to churches in accordance with Section 302.2 and have occupant loads of less than 100, shall be classified as A-3 occupancies.

305.2 Day care. The use of a building or structure, or portion thereof, for educational, supervision or personal care services for more than five children older than $2\frac{1}{2}$ years of age, shall be classified as a Group E occupancy.

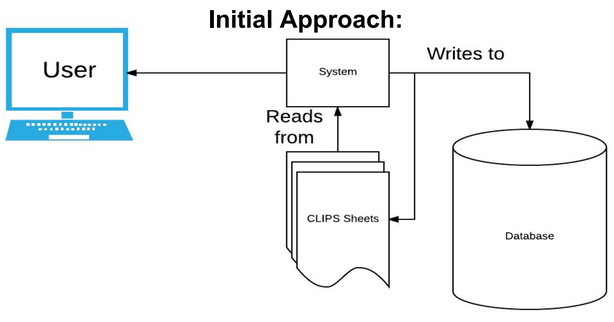
The use of a building or structure, or portion thereof, for educational, supervision or personal care services for more than five but no more than 100 children two and one-half years or less of age, when the rooms where such children are cared for are located on the level of exit discharge and each of these child care rooms has an exit door directly to the exterior, shall be classified as a Group E occupancy.

- The building code can be broken down into what is essentially a series of if/then statements
- Certain classifications relied on one "if", others upon many
- Thus, our dependency system, which relied on Boolean operators, could be used to systematize the code

Usability

- Used the Turbotax model
  - Took a complicated and highly technical topic and made it accessible and easy-to-parse
- Dependencies not only determined the user's code requirements, but also guided them through the program
  - Only asked questions that had not been made irrelevant by previous answers
- Simplified the language of the building code

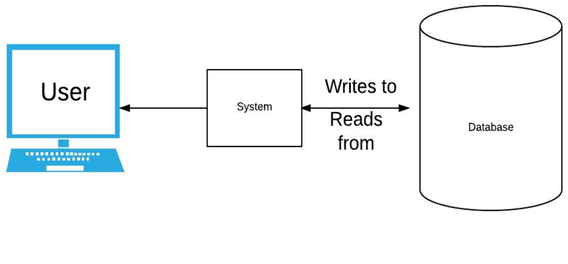
Initial Approach



dependency="qEB_IBC06_0302-1-Assembly,2"

- On startup, database was loaded into a CLIPS file
- Dependencies read from CLIPS, answers were written to CLIPS and database
- Answers, patterns, and question sets all stored in separate CLIPS files
- Overly complex with little scalability

Revised Approach



dependency="sAns(qEB_IBC06_0302-1-Assembly)==2"

- Eliminated CLIPS
- Answers were still written to database
- Dependencies formulated as Java functions which launched SQL queries

Conclusion

- At the end of the summer, development had been completed for Chapter 3 of the Ohio building code, "Classification of Facilities"
- The system will guide the user through a series of questions on their building's occupancy and usage, and then report what classifications it matches
- Additionally, the project is in an excellent position going forward
- The system is scalable to the rest of the code as well as to other building codes
- Development tools are in place to assist those who might continue this project

Acknowledgments

This work was funded by the Kenyon Summer Science Scholar Program, and could not have occurred without the support and guidance of Professor Jim Skon, as well as Mr. Lance DeBenedictis, who provided architectural insight during this project's development.. Additionally, I would like to thank Joseph Schutz and Benjamin Weinberg, who worked on concurrent projects this summer, and whose work was critical to my own.