Details of a Classification System to Extract Project-Specific Building Codes

Dana J. Vanier[1]

Abstract

This paper describes the details of the classes of components required to construct a Classification System to Extract Project-Specific Building Codes. It deals primarily with the National Building Code of Canada, but the Classification System could be used for other Canadian Model Codes.

Introduction

Although there has been considerable work in the area of representation of building components for product modelling, CAD representation and Computer Integrated Construction; many research projects attempt to model the entire construction industry, a difficult task to say the least.

In this paper, I describe the details of a classification system to extract project-specific building codes. The classification system attempts to model buildings, but only how they relate to building codes, standards and regulations. The functioning and advantages of the classification system have been described in another paper [Vanier 91]; this paper details the Classes or components of the Classification System.. These Classes have been established *a posteriori* to meet the needs of extracting code requirements for specific building projects. The development of the procedure was an evolutionary process, it started with a NIAM representation of the relations between objects [Turner] and evolved to a system that explicitly defines relationships, properties and interconnectivity of objects as they relate to the Canadian Building Codes.

The Canadian Building Codes consist of four discrete documents; the National Building Code of Canada, the National Fire Code of Canada, the Canadian Plumbing Code and the Canadian Farm Building Code. In the case of the *Classification System*[2], the domain and the delimitations of the project are the National Building Code of Canada.

Future work will describe how the *Classification System* was developed and how future software development will allow the user to extract code requirements for a specific building project.

The *Environment* of the NBCC is only a small component of the process called the construction industry. I make no attempt to model the construction industry, however the Classification System is a thorough, concise representation of the requirements of the

[1]National Research Council Canada, Institute for Research in Construction

[2]All *Classes* within the *Classification System* will be *italicized. Concepts* will be identified by double quotation marks and *Relations* by angle brackets. All these italicized terms will be described in this paper.

National Building Code of Canada. It is hoped that the development of the *Classification System* for the NBCC can be expanded to meet the need of other domains including CAD databases and facilities management.

Meta-Classification

Since this is a description of a *Classification System* for the National Building Code of Canada, it would be best to first describe the classification system of the *Classification System*, -- the meta-Classification System.

Definitions

Classification System refers to the Classification System to Extract Project-Specific Building Codes.

The *Source Document* is the document evaluated for the *Classification System.* In this case, it is the National Building Code of Canada, Edition 1990 with errata and revisions to Jan 1992. It will be referred in this text as the *Source Document* or the NBCC.

Functional Requirement describes the requirements the *Source Document.* A general summary of the *Functional Requirement* of the NBCC is 'a code of minimum regulations for public health, fire safety and structural sufficiency with respect to the public interest' [NBCC 90]. A more specific *Functional Requirement* is 'a standard of fire safety for the construction of new buildings, the reconstruction of building, including extensions, alterations, or changes in occupancy and upgrading of buildings to remove an unacceptable fire hazard'

Eye of the Observer is the term used to describe the perspective of the *Functional Requirement.* In the *Classification System,* the *Observer* is a fictional character enforcing the *Functional Requirements* of the NBCC, as they pertain to the *Concepts, Relations* and *Properties* in question.

Examples of *Concepts* used in this paper are between double quotation mark to assist the readability of the example. (eg. "Buildings"). *Relations* among *Concepts* are placed within angle brackets as in the example: "Mezzanines" are <Part_of> "Buildings". The *Relations* will contain an underscore in place of a space between words.

Delimitations of the *Classification System* extend to the *Source Document* and the terminology used in the NBCC. In most cases the prevalent terms used the NBCC will be selected for the *Classification System,* however in a limited number of cases the more common, colloquial term will be employed. In these cases, both *Concepts* will exist in the *Classification System* and a <Used_for> and <Use_term> *Relation* will be identified. That is, "Gypsum Plasterboard" is <Used_for> "Dry Wall Sheathing". The *Limitations* of the *Classification System* are defined by the correctness and accuracy of the existing NBCC.

In some cases, additional *Concepts, Relations* or *Properties* are required to properly interrelate the *Functional Requirements* of the *Source Document*, even though these terms are not employed in the *Source Document.* In these cases additional *Concepts, Relations* or *Properties* will be created in order to fill these voids. These are Dummy entities and are suffixed with an asterisk "*".

The *Classification System* describes generic components of a building and how they relate to the NBCC. The *Classification System* is not a database of instances of components for any particular building or type of building. Until the *Classification System* is finished it is not known if it can be used to fully describe a building in a database sense. Because of the *Delimitation* of the *Source Document,* many components of buildings will not be represented in the *Classification System* because they have no relation to the *Functional Requirements* of the NBCC (eg. Works of Art).

Classes

There are a number of *Classes* in the *Classification System* each playing an essential role in the development of the hierarchies that represent the *Functional Requirements* of the NBCC. These are *Environment, Concepts, Hierarchies, Relations, Properties, Property Values* and *Instances* (These will be described in detail later on in the text). In addition, there are a number of formal terms to assist in the definition of the *Classification System,* such as *Roles* and *Domains* (See below).

Definition:	*Classes* are those meta-components of the *Classification System* needed to fully describe and contain the *Functional Requirements* of the NBCC.
Description:	A Class is a definition to assist the description of the *Classification System* and not an essential component of the *Classification System.*

Environment

Definition:	An *Environment* is the area of influence of the document under investigation. In the case of the *Classification System,* the *Environment* is the NBCC.
Description:	This not only describes the boundaries of the area but also the *Functional Requirements* of the model. A *Functional Requirement* Model to describe building codes is different from a model to describe the components of the building.
Requirement:	The Environment defines the *Limitations* and *Delimitations* of the *Classification System.*
Application:	The *Environment* is a definition required to describe the bounds of the *Classification System.*
Examples:	The *Environment* encompasses all of the *Functional Requirements* of the NBCC. However, the *Concepts* and *Relations* developed may not be appropriate for the National Fire Code of Canada. The overall structure of the Classification System may not be appropriate for German building codes and standards.

Functional Requirements

The *Functional Requirements* of the NBCC are a series of requirements established by the code authorities. Additional research is required to itemize all the *Functional Requirements.* In general, the *Functional Requirements* deal with 'a code of minimum regulations for public health, fire safety and structural sufficiency with respect to the public interest' [NBCC 90]

Definition:	The *Functional Requirements* of the NBCC include all rules of conformance of the *Environment.*

Description:	The Functional Requirement, called Technical Solutions or TS in other publications [de Waard 92], are the set of rules of conformance that embody the *Environment*. During the course of future research in this project it is hoped that a limited number of *Functional Requirements* will be researched and classified. At this time only the major requirements have been identified.
Requirement:	This *Class* of component in the *Classification System* is the link between the technical intent of the NBCC and the function of individual *Concepts*. It is too early in the research phase to clearly identify how and where *Functional Requirements* will be used and how they will relate to other *Classes*.

Cross-referred standards are viewed as additional *Functional Requirements*. Over 244 related standards are referenced in the NBCC, these in turn can reference over 1000 other municipal, provincial, national or international standards.

Application:	Individual Functional Requirements will be treated as Concepts at this phase of research. They will have a <Part_of> Relation to other Functional Requirements and will have compliance Relations to other Concepts.
Examples:	"Buildings" <Must_comply_with> "Life Safety" "Beams" <Must_comply_with> "Structural Strength Limits" "Air Supported Structures" <Must_comply_with> "Standard CAN3-S367

Hierarchies

Hierarchical structures are used to inter-relate all *Concepts*. Although most of the relationships are a standard tree-hierarchy as in the case of <Part_of>, <Type_of>, and <May_contain> Relations; a number of Concepts can be related by one and only one Relation as in the case of <Opposite_of>

This Hierarchy will not represent the actual composition of a specific building. It is used to describe how a building is interconnected but not the representation of the instances of building components for a specific project. That is in the domain of an instance model for the *Classification System* and beyond the scope of this project.

Concepts

Definition:	*Concepts* are entities, things, or parts [Salthe 85], that are real or abstract, permanent or temporal, static or dynamic, and of definite or indefinite extents. The terms used to describe *Concepts* are not formal definitions, they are used to describe the range of *Concepts*.
Description:	*Concepts* are the components of the *Classification System*. They are inter-related by *Relations* (See below) and they can possess any number of *Properties* (See below). They can inherit any number of *Properties* from Parent *Relations*. These have been described as objects, entities, parts and concepts in other models [de Waard 92] [Salthe 85] [Sneath 73].
Requirement:	*Concepts* are the major *Class* of the *Classification System*. *Concepts* all perform a function that is related to the *Functional Requirement* of the NBCC.

Application: *Concepts* are used in the *Classification System* to denote entities, things, or parts of a tangible nature. A *Concept* is a well-delimited, well-behaved, well-described "thing" that performs a function within the *Functional Requirements* of the NBCC. *Concepts* differ from other *Classes* in the *Classification System* in that *Concepts* can stand totally alone; *Concepts* have *Relations* to other *Concepts*; and *Concepts* can possess any number of *Properties* with a defined range of *Property Values.*

Examples: *Concepts* can be real or abstract such as "Building Plans" and "Building Designs", respectively. *Concepts* can be permanent or temporal such as "Beams" or "Scaffolding", respectively. *Concepts* can be static or dynamic such as "Foundation Walls" or "Daylighting", respectively. *Concepts* can be of definite or indefinite extents such as "Hallways" or "Circulation Spaces", respectively.

Plural form: *Concepts* are normally identified in the plural form, however if the *Concept* is associated with the notion 'How much _______?' it will be recorded in the singular form (eg. 'How much Scaffolding?'). If the *Concept* is is associated with the notion 'How many_______?' it will be recorded in the plural form (eg. 'How many Bricks?') [CTCST 74]. The reason for the standardization on the plural form is to control the vocabulary for the persons generating the *Classification System* and the people using it.

Many-to-one: *Concepts* can have Many-to-one associations with any other *Concept.* That is, a *Concept* can have any number of other *Concepts* as Child *Relations* and the *Concept* can be a Child *Relation* to any number of other *Concepts.*

Properties: *Concepts* can possess any number of *Properties.* (See below)

Relations: *Concepts* can possess any number of *Relations.* (See below)

Events: *Concepts* can be *Events*, a special class of *Concept.* (See below)

Instances: *Concepts* are generic *Instances*, they do not represent any specific component or entity in the building process but they represent this class of component. (See below)

Events (Type of *Concept*)

Definition: An *Event* is the addition of energy to a situation to produce a temporary or altered *Concept.*

Description: *Events* are an essential but little-used *Class* of *Concept* in the *Classification System*; let us say in many situations it is more of an abstract *Concept* than a Thing. The need for this type of *Class* is best seen in examples: "Fires" are an *Event* that alters the characteristics of material; "Creep" is a deformation over time that could affect the structural safety of concrete; and "Rust" will reduce the strength of brick anchors and ties. "Condensation" is the <Result_of> the temperature of a surfaces reaching their "Dew Points".

Requirement: The class of *Events* is needed to identify situations that do not exist but are a possibility. In actual fact, with respect to structural safety and protection of occupants, building codes are primarily interested in preventing specific

Events from occurring. However, if *Events* such as "Fires" are possible and it is essential to understand the consequences, then that portion must also be modeled.

Application: *Events* exist to describe phenomena that would otherwise be difficult to identify or Concepts that are of a dissipative nature, meaning ones that change over time.

Examples:

"Fires"<Cause> "Reduced structural strength of Steel Members"
"Fires"<Cause> "Radiative Heating"
"Fires"<Cause> "Smoke"
"Condensation" is <Result of> "Reaching dew point".

Events require a different set of *Relations* to describe the interconnectivity. Unfortunately, this is a complex *Class* with little application in *Functional Requirement* model and has not been investigated in detail.

Instances

Definition: *Instances* define a specific occurrence of a *Concept* in a real life situation.

Description: *Instances* are beyond the limits of the *Classification System* at this time.

Properties

Definition: *Properties* are variations to a *Concept* according to the *Functional Requirement* of the *Classification System.*

Description: The *Properties Class* allows common attributes to be passed to the *Domain* of related *Concepts* without creating a exhaustively detailed selection of types. An example is "Bowling Lanes"; although the number of lanes in a "Bowling Alleys" can vary it is not necessary in the *Functional Requirement* of the *Classification System* to classify 16 types of "Bowling Alleys" according to the need to distinguish between "Bowling Alleys" with 1 "Bowling Lane" and 16 "Bowling Lanes". The *Functional Requirement* of the *Classification System* must know only when "Bowling Alleys" have more than 4 "Bowling Lanes", for example.

The *Classification System* must be viewed as a dynamic entity subject to modification and updating. If the *Source Document* is altered in such as way as to affect the *Functional Requirement* of the *Classification System,* it must be redefined. Normally in a *Source Document* such as the National Building Code of Canada, this will be a refinement of the *Classification System.*

What is the difference between a *Relation* and a *Property*? A clear distinction in the *Eye of the Observer* must be made between a *Relation* and a *Property.* It is not obviously clear as to when *Relation* or *Property* is to be used (even to the author). If there are three and four lane "Bowling Alleys" are these not different <Type_of> *Relations* or are they Properties of "Bowling Alleys"? By definition of the *Classification System* if there is only one attribute that distinguishes a potential series of Child *Relations* then this should be identified as a *Property.*

Requirement:

Properties are defined by an *Property* Identifier and a set of allowable *Property Values*. The *Property* Identifier is one of a parsimonious selection of identifiers available in the Classification System. Each *Property* has a discrete number of *Property Values* that reflect the needs of the *Functional Requirement* of the *Classification System.*

Child Level *Properties* override Parent Level *Properties*

Properties must be parsimonious. There must be a minimum number of *Properties* and a sparse number of *Property Values* based on the *Functional Requirement* of the *Classification System.*

Must answer the following question: -- Are the variants of the Parent Level identical in all functional aspects with the exception of this *Property*?

Application: *Properties* are used to identify differences in the *Concepts*. Normally, *Properties* are extracted from the text of the *Source Document* by observing *Relations* between terms in the NBCC.

Examples: Standard Number CSA B78.5 is a *Property* of the CSA Standard for Computer-Aided Design Drafting (Buildings)

Property Values

Definition: *Property Values* are a discrete, well-defined, parsimonious set of values that are applicable to the *Property* to meet the *Functional Requirement* of the NBCC.

Description: *Property Values* are a key *Class* in the operation of the *Classification System.* Along with the Child *Relations*, they permit the full description of the attributes of *Concepts. Property Values* are established according to the *Functional Requirements*; only those sparse values which are identified in the *Source Document* are identified as *Property Values.*

Property Values of Child *Relations* also be more specific than Parent *Relations*. That is, if the Parent can possess the Property X with Property Values Ranging from A to X, then the Child of that Concept cannot contain additional Property Values of Y and Z.

Relations

Definition: A *Relation* is a mutual involvement of *Concepts.*

Another definition proposed [Salthe 85] for *Relations* is 'The state of being mutually involved; mutual influence'.

Description: *Relations* are the kernel of any *Classification System*, they describe how *Concepts* are mutually involved and they identify the polarity of the relationship. The *Relation*, along with the involved *Concepts* is described as a *Role* (See definition below).

Typically, one thinks of the two prominent involvements: <Part_of> and <Type_of>, when one thinks of *Relations*. However, there is a robust but limited selection of *Relations* for the *Functional Requirement* of the *Classification System* and the specific *Relations* are a function of the requirements of the *Source Document.*

Relations can range from involvement such as hierarchical associations such as "Part_of" to relational involvement such as <Standard_for>. In the latter, one *Concept* such as CSA Standard "Access Scaffolding for Construction Purposes" could be involved with the *Concept* "Scaffolding" in such a Relation. In turn, both *Concepts* could be related to others in the standard fashion; "Scaffolding" is a <Part_of> the *Concept* "Construction Phases" and "Access Scaffolding for Construction Purposes" is a <Type_of> the *Concept* "CSA Standard".

Relations exist in a Triad between two *Concepts.* This is typical in the hierarchical *Relations* such as <Type_of>. In this example there are normally Parent Level - Focus Level - Child Level *Relations*, so any Focus Level is related to the Parent Level in a specific *Relation* and the Focus Level is related to the Child Level in the reverse *Relation.* That is, in the case of "Designer" - "Engineer" - "Foundation Engineer". The Focus Level "Engineer" is a <Type_of> the "Designer" and the Focus Level possesses a number of <Types_of> "Engineers" including "Foundation Engineer".

Typically in a hierarchical *Relation* the Parent Level is providing general rules and constraints and the Child Level is detailing the possibilities. [Salthe 85] Take for example "Industrial Occupancies"; the general rules here apply to all of the underlying hierarchy, whereas the Child Levels such as "High", "Medium" or "Low Industrial Occupancies" are dictating the possible states of the general rule. It is also true that *Concepts* at the Parent Level appear to be more static than those of the Child Level [Salthe 85].

What is the difference between a *Relation* and a *Property*? A clear distinction in the *Eye of the Observer* must be made between a *Relation* and a *Property.* It is not obviously clear as to when one or the other is to be used. If there are three and four lane "Bowling Alleys" are these not different <Type_of> Relations or are they properties of "Bowling Alleys"? If there is only one attribute that distinguishes the children of a *Concept* then it can be represented by a *Property.*

Requirements:

Parsimony is a required quality of *Relations*. Within any *Source Document* there are a limited selection of logical *Relations* that can occur. Any *Classification System* can contain an infinite number of types of *Relations*; the *Classification System* would prove to be inaccurate and misleading.

Two *Relations* that are functionally identical in the *Eye of the Observer* should be represented as one *Relation*. For example, objects are glued together; objects are fastened or objects are connected; if one relation does subsume all the others and they in turn are seen as identical by the observer, then one *Relation* will suffice.

No new *Relations* can be created until the existing similar ones are evaluated for their suitability and changed if necessary. If a new *Relation* is required, the domain of similar *Relations* will be re-evaluated and changed if necessary for all affected *Concepts.*

For every *Relation* there is a *Reverse*. For each obverse *Relation* of A to B there in an reverse *Relation* of B to A. *Relation* A-B does not necessarily equal *Relation* B-A. If *Concept* A is the <Standard_for> the *Concept* B, then *Concept* B <Complies_with> *Concept* A. Notice that in most cases the reverse of the *Relation* is a different *Relation*; however sometimes the reverse is the same *Relation*, as with <Replacement_for>.

Some *Relations* associate two different *Concepts* or two instances of the same *Concept*. The *Concept* "Adfreezing" is the <Adhesion_of> the "Soil" to the "Foundation Units". In the latter case an example would be that "Tunnels" <Connects> different instances of "Buildings".

Application: *Relations* are a key element to any *Classification System* and are the binder between *Concepts*. For every two *Concepts* that are involved or have a *Role*; there exists at least one *Relation*.

Examples: Examples of types of *Relations* include the following:

Classes of *Relations* - This is an expansion of the 4 general types of *Relations* identified by other researchers [Danner 88], namely Active, Dative, Locative and Partitive. These could be viewed as a series of subtypes of Relations. These are somewhat arbitrary typing and not a solid portion of the *Classification System.*

Classification: This is the standard <Type_of> classification hierarchy. *Concepts* are placed in a string hierarchical tree structure according to their *Functional Requirement* in the *Eye of the Observer.*

"Engineers" are <Type_of> "Designers"
"Designers" include <Type> "Engineers"

Aggregation: This is the standard <Part_of> classification hierarchy. *Concepts* are placed in a string hierarchical tree structure according to *Functional Requirement* in the *Eye of the Observer.*

Concept "Chimney Liners" are <Part_of> the *Concept* "Chimneys"
Concept "Chimneys" are <Parent_Assembly> to the *Concept* "Chimney Liners"

Association: This type of *Relation* is a collection of the *Relations* that do not correspond to the other headings.

Concept "Architects" is <Responsible_for> the "Building Designs"
Concept "Building Designs" is <Responsible_of> the "Architects"

Possibility: This type of Relation deals with the possibility of a *Relation.* This is necessary to describe *Relations* that are temporal and change over time; are not always necessary; or are a possibility.

"Stairs" <May_Contain> "Riser Lights"
"Riser Lights" <May_be_Part_of> the "Stairs"

Substitution: This is the representation of the poor vocabulary control for most texts and users. In this case a specific phrase is the preferred one in the *Eye of the*

Observer, all other identical terms <Use_Substitute>. This is a feature that is predominant in thesauri [CTCST 74].

"Bleachers" <Use_for> "Grandstands"
"Grandstands" <Use_Substitute> "Bleachers"

Analogy: One *Concept* can performs the function of another but can be morphologically different [Salthe 85]. This is the expansion of the <Related_to> *Relations* and serves a number of useful features. One being the possibility of cleanly linking two *Concepts* that are completely different but serve the same purpose, eg "Fluorescent Stair Strips" are a <Replacement_for> Stair tread lights.

"Riser Marking Stripes" are a <Replacement_for> "Riser Lights"

Calculation:

"Low Fire Load Occupancies" are <Derived_from> the "Fire Loads"
"Fire Loads" <Determines> the "Low Fire Load Occupancies"

Negation: Sometimes the negation of a *Concept* is equally important to the obverse *Concept*. This structure allows both. Many examples exist in building codes such as: If the building is not greater than 8 storeys, if the patients are Nonambulatory; or if the building was built before 1975. The Negation *Relation* permits easy manipulation of *Concepts*.

"Ambulatory Patients" are the <Opposite_of> "Nonambulatory Patients"

Location: Some *Concepts* are related to others owing to physical location. Although location is a *Property* that is inherited in most Aggregation *Relations*, there are a number of Location *Relations* that are referenced in building code documents.

"Rock" is <Below> the "Buildings"

Complex *Relations* *Relations* to three or more *Concepts* will be treated as a multiple Relation

"Tunnels" <Connect> "Buildings" to other "Buildings"

Roles

Definition: [Danner 88] A *Role* is a link the can be used between a *Relation* with which it is associated and a *Concept* in forming an idea.

This is a semantic definition to aid explanations and not a *Class* in the *Classification System*.

Description: A *Role* is the obverse and reverse of the *Relations* and the involved *Concepts*.

Concept "Tunnels" <Connect> the "Buildings" to other "Buildings". The role in this relationship is the involvement of "Buildings" and "Tunnels". In the *Eye of the Observer*, this is a special unique relationship.

Requirement: Classification definition

Application: Classification definition

Examples: "Tunnels" <Connect> the "Buildings" to other "Buildings"

Domains

Definition: A *Domain* describes an inter-related set of *Concepts* dealing with a specific *Functional Requirement*; a self-contained cluster of *Concepts.*

This is a semantic definition to aid explanations and not a *Class* in the *Classification System.*

Description: A *Domain* is all the *Concepts* and their *Relations* within the entire functional area that deal with a specific functional area in the *Eye of the Observer.* These can be viewed as stand alone functional areas with little relationships to the remaining set of *Concepts*. It is best described with an example: The *Domain* of all *Concepts* dealing with "Demolition Phases".

Requirement: Classification definition

Application: Classification definition

Examples:

Domain of all Concepts dealing with "Illumination of Signs"

Domain of all Concepts dealing with "Daylighting"

References

[CTCST 74] *Canadian Thesaurus of Construction Science and Technology*, Industry Science and Technology Canada (formerly Department of Industry Trade and Commerce), Ottawa, Canada, 1974

[Danner 88] Danner, William F., *A Global Model for Building-Project Information: Analysis of Conceptual Structures*, NBSIR 88-3754, National Bureau of Standards, April 1988.

[NBCC 90] *National Building Code of Canada*, Tenth Edition, Institute for Research in Construction, Ottawa, Canada, 1990

[Salthe 85] Salthe, Stanley N., *Evolving Hierarchical Systems - Their Structure and Representation*, Columbia University Press, New York 1985.

[Sneath 73] Sneath, Peter H.A., Sokal, Robert R., *Numerical Taxonomy*, W.H. Freeman and Company, San Francisco, 1973

[Turner] Turner, James A., PDES/STEP AEC NIAM Model, Architecture and Planning Research Laboratory, The University of Michigan, Personal Correspondence.

[Vanier 91]	Vanier, Dana J., "A Parsimonious Classification System to Extract Project-Specific Building Codes", *Proceedings of the Workshop on Computers and Building Regulations*, VTT Symposium Series No. 128, Espoo Finland, May 27-29, 1991.
[de Waard 92]	Waard, Marcel de, *Computer Aided Conformance Checking*, Koninklujke Bibliotheek, The Hague, 1992