Capturing and Integrating the Design Brief in Building Information Models

David Marchant, d.marchant@unsw.edu.au
UNSW, Australia

Abstract
The activities of briefing (stating problems to be solved) and designing (instantiating solutions) are intimately interlinked – both parts of the same process to develop and procure a product which satisfies a need. There is an ongoing "conversation" between problem definition and design proposal in which acceptable proposals progressively add to, and refine, the definition of the whole solution while at the same time potentially generating further problems to be resolved. At any point in time through this process, the developing whole solution is composed of partial solutions which form the context against which further problem statements are made. These problem statements are indicators for a future desired state of the whole solution. The design process also can be seen as starting before professional designers are involved, and continues after they complete their project contribution, so retaining design intentions alongside solutions is proposed here as valuable for the ongoing use of a designed product.

With these ideas as background, this paper investigates the hypothesis that data for briefing and design can be usefully correlated within integrated building information data models. To test this hypothesis, the content of forty six actual project briefs was coded to reveal generic briefing concepts relative to the data schema defined by the international IFC standard for building information models. The analysis showed that the use of type and relationship entities is a feasible way in which briefing and design can be integrated while still retaining an inherent "separateness". Several extensions (six new relationships and one new type entity) are proposed to the current IFC data schema (IFC4 2013).

Keywords: Briefing, Building Information Modelling (BIM), IFC

1 Introduction

In current design practice, information is stored and disseminated digitally, and there is the emerging ability to locate and interact with that information from a shared data repository for the use of all of the participants in a design project (BIM Industry Working Group 2011). However, people have different perspectives on what data they need, and use different software to access that data. For example, the detailed geometric resolution of a building design may be done using building information modelling (BIM) software, but the collection and organization of briefing information is often done in separate software with its own file store or database. This separation is currently being bridged by the ability to selectively synchronize the two distinct data environments, but the essential separation, and duplication, remains. Since building design, construction, and management can include many specialists having full or partial involvement within a relatively lengthy process, the separation of the total data about a building into subsets is a common way to allocate responsibility for that data, and for individual specialists to maintain control over their portion of the whole. The starting point for this research is to ask whether it is possible for these separate subsets to co-exist within a shared "super-set" data schema with a subsequent reduction in data redundancy and an ability to inter-relate the data elements in a rich and useful way. In this regard, the apparently distinct disciplines of briefing and designing are investigated as a case study.

In architectural design, briefing (or programming, as it is known in the US) is concerned with defining the context, vision and client requirements for a proposed building project. Pena (1977) refers to the goal of briefing as to 'state the problem', and Wade (1977) describes the inherent inter-

connection of problem definition (briefing) with solution generation (design). Blyth and Worthington (2001 p.3) distinguish the act, or process, of briefing, from the outputs of that process: 'briefing is the process by which options are reviewed and requirements articulated, whereas a brief is a product of that process'.

While traditional design practice makes distinctions between stages of design (briefing, sketch design, detail design, and documentation) in practice these distinctions are much more fluid. As design progresses, and new knowledge is developing, so the concerns of the briefing adjust and vice versa. Briefing requirements and design decisions relate to the level of detail being addressed as well as to who the intended resolver of those requirements might be. For example, my analysis of the concepts used in current commercially available briefing software shows a predominant concern with briefing at a detailed room level. Briefing at this level of detail is at a stage where implicit decisions have already been made – the departments and rooms are identified, only missing final positioning and finessing of their design detail. In effect, a brief at this stage can be understood as a partial solution to the ongoing design problem. In the client's consideration, prior to this are factors such as organizational configuration, financial viability, and business processes, many of which are relevant to intentions for the final design. Where the room is the organizing entity for detailed room briefing, the organization, business unit and people are relevant organizing entities prior to considerations of space. Additionally, requirements can often be contingent (placeholders) just to get the design moving in a desired direction, but once the design is underway those requirements can be subject to more rigorous consideration and challenge, and therefore also subject to change.

Information modelling in computer science aims to formally specify, in a computer-sensible way, the concepts which make up the universe of discourse (set of entities forming a "data schema") for a particular domain. My research has focused on a reconsideration of information modelling for the design briefing process via an investigation with regard to the following questions:

- What are the concepts that need to be captured in the briefing process?
- What is the nature of the relationship between briefing and designing?
- And, related to the two previous questions, are all the relevant semantics adequately modelled in current standard building information schemata?

The relevant data schema chosen for this research in the architecture / engineering / construction (AEC) domain is the Industry Foundation Classes (2013) because that specification has been accepted as an ISO international standard. Previous research that has addressed IFC in relation to briefing includes Kiviniemi's (2005) research on requirements management. Two IFC extension projects have also dealt with briefing issues: the FM-9 Portfolio and Asset Management – Performance Requirements project (PAMPeR 2004); and, the AR-5 Early Design project (AR-5 2006). More recently the Building Programming Information Exchange project (BPie 2012) has been defining common requirements for space and functional briefing. All of these projects have informed my own research in this area, but Kiviniemi is particularly acknowledged here as my initial inspiration.

A data model for a built facility that integrates briefing (problem statements) and designing (the results of solution decisions) allows for better communication and consequently, tighter integration between those who require design products and those who deliver them. In a building construction environment where risk can be high, profit margins are generally low, and communication fragmented, integration around a shared understanding of the building project should help to reduce waste caused by misunderstandings and lost information. Additionally, because the delivery times for a building project can be relatively long (years) and the occupancy and use of that building can be even longer (many decades), an integrated record of design intent along with the description of the design can serve to aid the understanding of the many interested parties who come and go over the full course of a building's lifecycle. The data schema proposed as a result of my research (IfcPlus – which is IFC extended for briefing) does not necessarily rely on all data being held in one integrated data repository however it provides the range of entities to logically do so.

2 An investigation of briefing artifacts

Written briefs are a form of discourse between the brief writer and the designer/reader in the sense that design context and design intentions for the proposed project need to be conveyed. In order to dissect this material into its constituent generalized concepts, a relevant and applicable set of

research tools to use are those under the broad umbrella of content analysis. Content analysis, in simple terms, is 'a general procedure for objectively identifying the characteristics of textual material' (Jones 1996 p.127).

Schreier (2012) describes content analysis as an iterative 'dialogue with cases'. For the research reported here, these source cases are forty six real project briefs. However, the chosen documents are only a sample of the potential population of interest. Sampling is necessary for obvious reasons since it would be too onerous (and realistically impossible) to code all existing briefing material. The briefs were obtained by contacting briefing consultants, architects, and client representatives from Australia only, but in order to offset an overly Australian bias, briefs were also sourced from the websites of architectural professional organizations such as the Royal Institute of British Architects (RIBA) who publish competition briefs. Consequently, the selection of the briefs for analysis seeks to address: work taking place internationally; work related to different building types (health, education, workplace, and lifestyle); work at different scales (master planning, urban design, building design, interior design, and, in one case, product design); competitions and other types of project procurement; and, briefs written at different levels of detail (broad scope expressions of interest through to very detailed requirement specifications). All briefs were available in English, so it should be noted that there is some inherent bias in the sample relative to non-English speaking designers.

Content analysis is a reductionist methodology in the sense that it condenses the subject material by coding according to categories which may be derived inductively out of the text being analyzed, or applied deductively from prior theoretical considerations outside of the text itself. In the analysis undertaken for this research both of these techniques are employed, the first automated via software, and the second done manually.

For the initial automated analysis the online version of the Leximancer text analytics software was used (Lexi-Portal 2014). Each briefing file was assigned to one of four document folders according to its briefing type:

- Competition (seeking design submissions)
- Strategic ("visionary", broad objectives)
- Functional (more detailed, particularly concerning organizational and functional considerations)
- Comprehensive (very detailed, including room data sheet level briefing as well as functional and strategic requirements)

By dividing the briefs in this way the intention was to allow for cross-comparisons to be made between the subject matter of briefs created for various stages of the design process.

Leximancer discovers the concepts occurring in source documents inductively. The steps a user of the software needs to follow are shown in Figure 1.

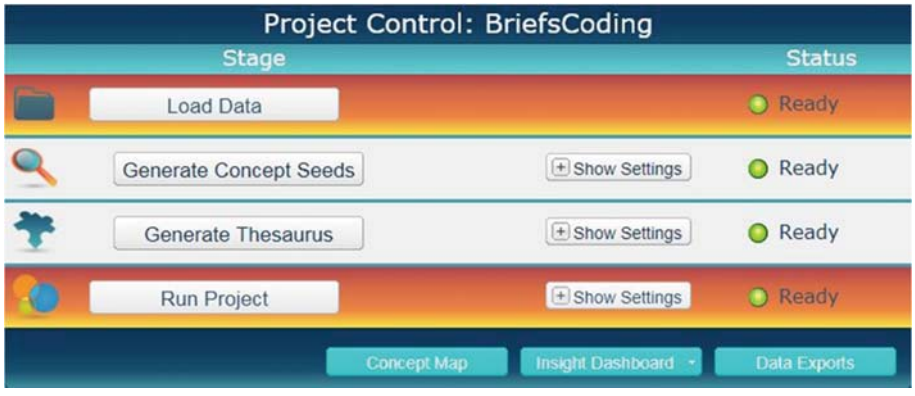

Figure 1 Leximancer process

The software performs both conceptual analysis (identifying each distinct concept and the frequency at which it occurs), and relational analysis (identifying the main relationships between concepts). It produces interactive graphical maps to show the results of a "run", clustering related

concepts as it has ranked them. Apart from the initial division of the briefs into four folders the analysis done using Leximancer is otherwise unmediated.

The use of Leximancer is an example of an inductive category development approach. However, the concepts derived in this way are not directly equivalent to the entities that are defined in the IFC international standard that is increasingly used for AEC projects. For the manual coding of the briefs a deductive category development approach was first undertaken in which theoretical based categories are adopted. In this case, the theoretical basis is the IFC schema. Basing the coding on those entities allows for any apparently missing constructs in IFC to be discovered inductively as part of the ensuing coding process.

Briefs on the whole are written in a formal manner, divided into clear sections that contain clauses which usually address one subject item per clause. For the manual coding, these subdivisions of the source material correspond well to what Schreier (2012 p.131) describes as 'units of coding'. Whole clauses are predominantly used except where it is necessary to split a clause due to more than one concept being addressed. Each clause is further subdivided into "subject", "object", "relation", and "property" component parts to align with the first level of subclasses in IFC (as defined in all released versions) below the fundamental *IfcRoot* entity:

- *IfcObjectDefinition* for subjects and objects ("things" / "nouns")
- *IfcPropertyDefinition* (characteristics of entities)
- *IfcRelationship* (relationships between entities, or between entities and associated property definitions).

Holsti (1969) describes five guidelines for constructing a set of categories:

- Categories should reflect the purposes of the research
- Categories must be exhaustive (all of the material can be assigned to at least one category)
- Categories should be mutually exclusive (that is each item in a text can only be coded against a single category, and, in particular, there should be no need for a catch-all "other" category)
- Categories should be independent – the assignment of one item to a given category should not affect the assignment of other items
- Categories should be derived from a single classification principle – different levels of analysis are not mixed, for example there is 'not a category for oranges plus a category for fruit'

For coding of the subjects/objects and relationships, IFC entity categories are used plus further extended categories that are derived through the initial coding and category revision process. For coding of the properties, the concepts are allowed to emerge inductively through coding that identifies the subject matter of the property (usually the predominant word used in the text which best summarizes that property). This coding frame in general satisfies Holsti's five criteria except it should be noted that IFC, because of the nature of its sub-classing and inheritance mechanisms, from generic entities (for example, *IfcOpening*) down to more specific ones (for example, *IfcWindow* and *IfcDoor*) does violate the last criterion. The violation also emerges through a consideration of types relative to instances of entities in the progression from generic to specific as briefing and designing develop in more detail across stages of the design process. The contradiction is allowed to remain in order to preserve this insight in the reported analysis. The *IfcProxy* class which is used in IFC to deal with otherwise non-differentiated entities violates the third criterion since it is a miscellaneous catch-all mechanism, therefore it is not used as a coding category. The coding of each clause is overlaid with an additional code to signify the type of requirement, to interpret the strength of the desire expressed (increasing from wish to need), and to discriminate contextual statements (those describing the existing situation) from requirement statements.

The results of the manual analysis were recorded in a relational database from which analytical charts were produced, distinguishing results by design stage, and by subject, object, property, and relationship entity types.

2.1 Coding reliability

Different emphases are proposed in the relevant literature regarding coding reliability. Quantitative content analysts like Krippendorff (2009) instruct that two or more coders are required, and the coding reliability between the coders is rigorously measured. Qualitative content analysts like Schreier (2012 pp.146-151) allow for cases where the quantity of material is too great, or where reliable coders are not available, that it is also permissible to use a single coder who codes the material twice at separated points in time. A mixed methodology is employed in this research. Computer software (Leximancer) is used to extract concepts and their frequency from the text material first. These concepts are informative, but are not directly correlated to IFC entities, so the "code twice" method (for this research the two coding sessions are separated by six months) is used to code the briefs against IFC entities, and to also inductively derive emergent concepts which do not fit what is currently available in IFC. Then, as a further investigation, two other coders were asked to code one of the briefs. Both coders are expert in their respective fields: one with a client perspective, and with some advisory capacity on the chosen project; the other with a designer perspective. Their coding is compared against mine for the same brief, and the results analyzed in terms of correspondences and differences. This one aspect of the coding process differs from formal content analysis in that, rather than seeking high levels of coding reliability between the coders, in this case the difference between the coders is also relevant and informative because it reveals client versus designer interpretations of what a brief is communicating.

Krippendorff (2009 p.3) points out that 'raw word counts can be suggestive, and content analysts often use word counts to get a sense of the vocabulary they are facing'. The use of content analysis methodology here is precisely this - to elucidate just such a vocabulary from the sample briefs, and to use the results of that discovery process to formally define a useable generic data schema for briefing that is a proper fit within an overall building information modelling framework. The validity of the methodology is described by Schreier (2012 p.175) as 'an instrument is considered valid to the extent that it captures what it sets out to capture. A coding frame is valid to the extent that the categories adequately represent the concepts under study'.

3 Analyses

3.1 Automated coding

The written project briefs were first converted to a common file format (PDF) and then grouped into folders indicating the degree of resolution in the design process that each brief addressed. This file repository was then input to Leximancer for analysis. Figure 2 shows a concept map produced from 1000 iterations on the source data. The source folders are shown in black capitals, with concepts clustering at distances away from their sources, and each other, relative to their inter-relatedness. Concepts shown in hot colours (red being the "hottest") are those ranked with most importance. As colours move towards the other end of the spectrum, so their importance diminishes.

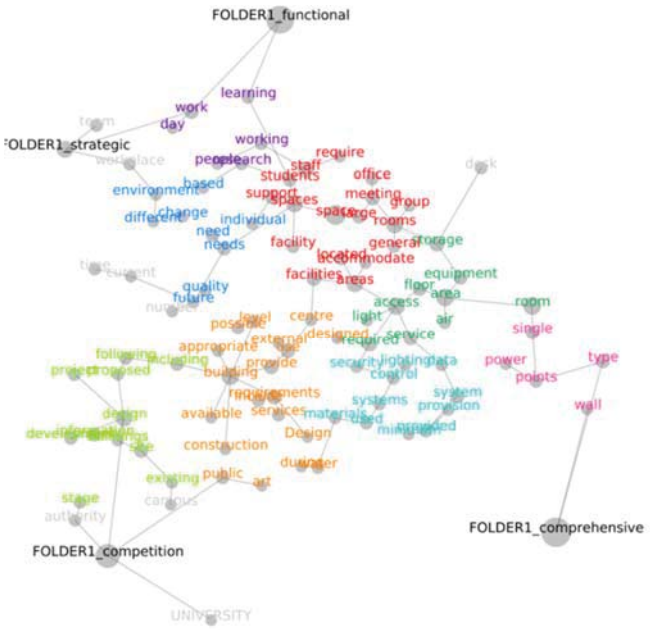

Figure 2 Unmediated analysis of sample briefs – concept map

While not completely clear-cut, there is some discernible logic to be found in the grouping of the concepts on this map. Competition briefs contain considerable amounts of information concerned with the process and conditions of the competition. Organizations, people and their activities tend to cluster in the strategic quadrant. Spaces and space types tend to cluster in the functional quadrant. Physical building elements tend to cluster in the comprehensive quadrant. This would seem to agree with a continuous model of the briefing process over time. Generically, briefs at different stages of a design process seem to progress from people-centric to space-centric to physical element-centric as Wade (1977) describes in his person-purpose-behaviour-function-object model.

3.2 Manual coding

Every brief in the sample set was then coded manually clause-by-clause. At a meta-level, the clauses were each coded according to their intention ("requirement level"). A requirement level is categorized in terms of: level of need expressed ("vision" and "wish" versus a more tangible "need"); information about the current situation at the point in time at which the brief is written ("context"); a pointer to some other information external to the brief ("reference"); an example to inform the design ("exemplar"); a limiting factor ("constraint"); or a possible solution option ("proposal"). Figure 3 shows the results for this component of the overall coding.

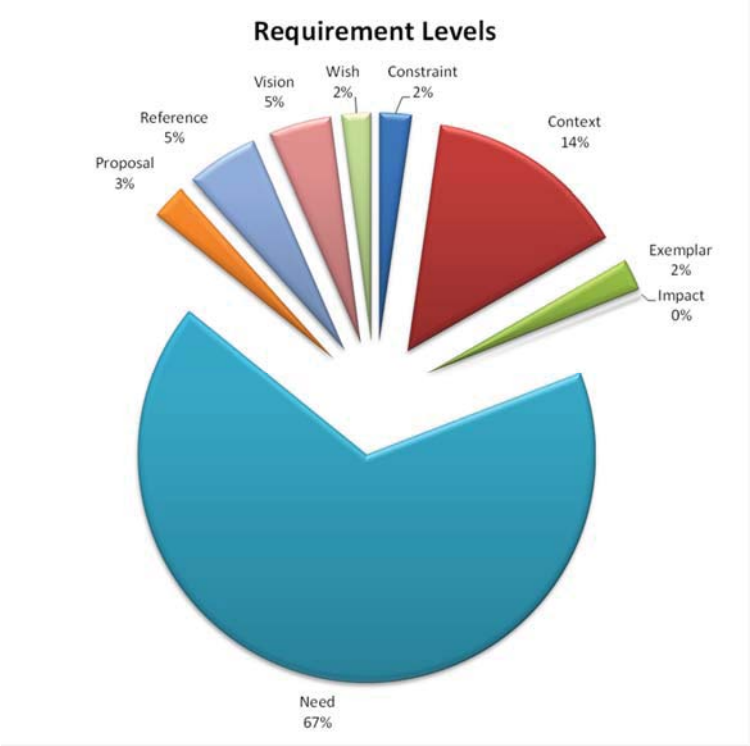

Figure 3 Manual coding – requirement level of a clause

While "need" is by far the most common interpretation for a clause in a brief, it can be seen from the graph that briefs also contain a significant quantity of contextual entities (24% if "exemplar", "reference", and "proposal" are counted alongside "context"). This finding is one indicator towards the idea that briefing (stating the problem) is a continuous process alongside, and intertwined with, designing (defining the solution). From an object-oriented perspective (BIM), decisions that have been made up to a given point in time find expression as object instances (an existing site, building, space, other object, or object property), and decisions that are to be made in the future are indicated either as descriptive statements (requirement properties attached to those existing contextual object instances), or by reference to generic type objects as placeholders for the final decision. The identity of these "instances" relative to currently defined IFC entities was coded in terms of the subjects and objects of a clause ("the nouns"), the properties associated with those subjects and objects, and the relationships between them. Where an item had meaning other than what could be clearly associated with a corresponding IFC entity (other than as a generic *IfcProxy*), an extra coding category was added to identify that and similar items where these were found. The analyses for the manual coding were done according to the same grouping of briefs that was used for the Leximancer analysis, and the progression from people-centric to space-centric to physical element-centric was again observed. The chart shown in Figure 3 is an example of what was also produced for subjects, objects, properties, and relationships to show the results of the coding. The majority of items could be coded successfully as IFC entities, but it is from the identification of the exceptions that insights can be gained regarding potential extensions to IFC for briefing purposes.

The exceptions identified were of two distinct kinds: those related to a requirement relationship between entities; and, those related to the use of entity types to make generic statements of requirement.

As an example of a requirement expressed as a relationship, consider this statement from one of the briefs – "The location of the kitchen to be adjacent to the café/bar area". This is indicating a topological relationship between two proposed spaces. However, the statement is made in words and is pre-geometric because neither of these two spaces yet has form or location. The requirement could be expressed in IFC as an adjacency property in a property set associated with one of the two spaces but that would mean that there would be no corresponding inverse against the other space. Using a relationship entity that is linked to each of the two spaces to capture the adjacency

requirement would seem to be a stronger means to capture the semantics of this statement. Furthermore, if that relationship is qualified by the degree of adjacency required, the "requirement level" of the clause can also be expressed.

Types were found to be used for briefing as generic placeholders for later instantiation of actual instances of the type, or as shorthand means to describe an existing situation. For example, here is a requirement for a quantity of a particular space type – "Provide 100 guestroom suites". It is stated as an imperative, therefore the strength of the requirement can be understood as relatively strong. The target of the requirement (the other side of the relationship) is unstated. Most often in these situations the implied target is the "project" as a catch-all entity. This example therefore shows a relationship for a required quantity between an instance entity (the project) and a type entity (the guestroom suite). If a statement uses the quantity of a type to describe an existing situation, again there is no corresponding IFC relationship entity that can be used. For example – "The Division breakdown: General Manager (1), Senior Project Analysts (5), Project Analysts (6), Contractors (2), Support Staff (3)". Here an organizational entity currently has a relationship with (contains numbers of) people in particular roles. The organization can be instantiated in IFC as an *IfcActor*, but for the person roles there is no corresponding actor type in IFC. For this example, two possible extensions to IFC are indicated: the quantification relationship, and an *IfcActorType* entity.

If "requirement by quantity of type" and "requirement by adjacency" are expressed as requirement relationships, there is also a corresponding implication for two other relationship constructs that were found in the source material. Firstly, one way to state a requirement is by doing so using a qualifying property attached to an entity. For example, "an executive office shall have an area of 18 square metres". This can be done in IFC by including an area property in a property set which then has an *IfcRelDefinesByProperties* relationship to the space or space type entity. The area property could be named as *RequiredArea*, or alternatively, the property set can include "Requirements" in its name. The BPie (2012) initiative has standardized some property sets for briefing using the property set naming methodology. However, an alternative construct could be to use a specialized requirement relationship to link these (or other) property sets to object instances or types. In this case, instances of standardized property sets can be understood as representing actual property values ("definedbyproperties") or required property values ("requiredbyproperties") depending on the kind of relationship used to do the linking. Similarly, a requirement that is expressed as a reference to some source outside of the brief itself could be instantiated using a relationship. In this case, IFC already has an *IfcRelAssociatesDocument* relationship that could be used, but if the same idea of qualifying the relationship as a requirement is applied, then both the explicit notion of "requirement" and the "requirement level" attribute can be expressed. For example, in the following statement the requirement level is a strong "need" relative to the external reference document – "the building shall be designed to comply with the Building Code of Australia".

3.3 Parallelism between briefing and designing

In the situation where briefs and designs exist in separate repositories cross-checking and synchronization related to "briefed" versus "designed" entities relies on maintaining a common identifier for corresponding entity instances. For example, the brief contains a space instance with a user-defined code, and the design contains a corresponding space instance with the same user-defined code. This simple coding mechanism constitutes an implicit relationship which allows functionality such as checking the properties of the two space representations (as-briefed and as-designed) against each other, and where the entities contained by those spaces are also coded, checking the correspondences of the contained entities as well. A consideration of the same example in terms of an integrated brief/design repository reveals an apparent problem. If both instances co-exist in the same repository, then without some form of differentiation, an uninformed query to return all space instances and sum their areas will return the wrong total because it will include all space instances (regardless of whether they are as-required or as-designed) instead of filtering to retrieve one set or the other.

The discussion of properties in section 3.2 indicates one way in which the required properties of an entity instance can be maintained separately from its actual properties through the use of two distinct kinds of relationship. There is now the possibility for the requirement properties and designed properties to be contained in one repository without the need to duplicate the entity

instance. Stating required and actual quantities of an entity using relationships is a further useful mechanism when we understand that the entity on the required side of the relationship must be a type. To illustrate this point, consider the distinction between an instance of a space and the instance of a space type in a data repository. Both are unique, but the space represents a concrete "thing" whereas the type is the definition for a "kind of thing". For example the Executive Office type defines the characteristics of what the generic space may be, but it is only notional. It is not an instance such as the space named Room 101 that is a kind of Executive Office. In IFC, the relationship that defines "is kind of" between the two is an *IfcRelDefinesByType*. Using the proposed quantity relationships, various numbers of the Executive Office can be required by many organizational units without the need to instantiate any more than the one instance of that type. When instances of actual spaces defined by this type are eventually created, there is no misunderstanding between what is briefed (the number as an attribute of the relationship and the type it refers to) and what is designed (the instances that are kinds of that type).

To give some indication of the possible reduction in redundancy using this type quantification methodology for briefing, consider the following partial example. The design brief for a hospital inpatient department requires a quantity of ten 4-bed wards. If this is expressed using an instance of *IfcActor* for the department, types for the ward (*IfcSpaceType*) and bed (*IfcFurnitureType*), collected together using the proposed required quantity relationship, the number of entities in the briefing model will be 5. That is, one for the *IfcActor* instance, plus 2 relationships, and 2 types. More generally, the formula for the number of entities to be instantiated can be expressed as:

$q1 + 2n$
where $q1$ equals 1, and n is the number of nested levels below the root instance

On the other hand, if this example is expressed entirely using instances of the entities department (*IfcActor*), ward (*IfcSpace*), and bed (*IfcFurniture*) collected together using an *IfcRelAggregates* relationship, the number of entities needed to do the briefing will be 62. That is, one for the *IfcActor* instance, plus one *IfcRelAggregates* instance to collect the ten wards to the department, plus ten *IfcSpace* instances for the wards, plus ten *IfcRelAggregates* instances to collect each set of beds to their associated ward, plus forty *IfcFurniture* instances for the beds. In this case, the generalized formula for calculating the number of entities to be instantiated in a briefing model is:

$q1 + (q1 + (q1 \times q2)) + \ldots ((qn\text{-}2 \times qn\text{-}1) + (qn\text{-}1 \times qn))$
where $q1$ equals 1,
n is the number of nested levels below the root instance,
and qn is the number of required instances at level n

In other words, at each level of an aggregated hierarchy of instances, the number of relationships will be equal to the number of instances in the level above, and the number of instances will be the required number of instances at this level multiplied by the number of instances at the parent level above.

For this simple example, there are roughly twelve times more entities instantiated using the instance methodology versus the type methodology. In a large project such as a hospital, where repetition of types recurs frequently, this proliferation of instantiated entities can be significant. On the other hand, where a project exhibits very little type repetition, the difference between the two methodologies will be negligible because the qn factors will mostly have a value of 1.

4 Summary

Several briefing-specific concepts have been identified from the analyses of the source briefing material that suggest possible extensions to the IFC standard. Six new relations and one new type are proposed:

Table 1 Proposed extensions to IFC for briefing

Relationships

IfcRelRequires	An abstract subtype of IfcRelationship under which to collect the instantiable requirement relationships. This entity includes two optional attributes that have been adopted from the earlier Pset_ProductRequirements property set: - DemandImportanceValue to indicate the strength of a need - SatisfactionValue to indicate the assessed satisfaction of the requirement.
IfcRelRequiresByProperties	A subtype of IfcRelRequires to aggregate an IfcProperytSet to an IfcObjectDefinition in which all the properties are specified as requirements.
IfcRelRequiresByDocument	A subtype of IfcRelRequires to handle the assignment of a document containing requirement information to object occurrences or object types.
IfcRelRequiresByAdjacency	A subtype of IfcRelRequires to handle the specification of a requirement for adjacency between objects or object types.
IfcRelRequiresByType	A subtype of IfcRelRequires to handle the aggregation of types by required quantity.
IfcQuantifiesByType	A subtype of IfcRelationship to handle the aggregation of types by quantity.

Types	
IfcActorType	The IfcActorType defines a list of commonly shared information for occurrences of actors.

The IFC4 (2013) release of the IFC specification has significantly increased (over previous releases) the number of types that are defined. *IfcActorType* is proposed here as a further addition that has applicability for briefing at a pre-spatial, people-centric phase. Another entity type that may be relevant to briefing (particularly if, and when, BIM is employed at an urban scale) is an *IfcBuildingType*. However, no explicit evidence for the use of this entity type was found in the analyses undertaken.

The proposals made here add complementary functionality relative to related work that has gone before. The use of *IfcRelRequires* and its relationship subclasses clarifies the distinction between briefed and designed, while also allowing for a connection between the two. Additionally, the use of types in the progression from generic to specific is revealed to be an important component of the briefing/designing process. Further analyses of briefing material (for other project types, project scales, and disciplines) may uncover additional briefing concepts to be accommodated in an extended IFC schema. These generic entities should provide a logical foundation on which to build, if required.

References

AR-5 (2006), IFC Extension Project: AR-5 Information Requirements Specification – [AR-5] Early Design – V5 (project leader Francois Grobler), A report by IAI NA and UK

BIM Industry Working Group (2011), BiM Management for value, cost and carbon improvement: A report for the Government Construction Client Group http://www.bimtaskgroup.org/wp-content/uploads/2012/03/BIS-BIM-strategy-Report.pdf

Blyth, A. & Worthington, J. (2001), Managing the Brief for Better Design, Spon Press, London and New York

BPie (2012), Building Programming information exchange,
- General information http://www.nibs.org/?page=bsa_bpie
- Requirements analysis http://projects.buildingsmartalliance.org/files/?artifact_id=4700
- Exchange requirements http://projects.buildingsmartalliance.org/files/?artifact_id=4699

Holsti, O. R. (1969), Content analysis for the social sciences and humanities, Addison-Wesley, Reading Massachussets

IFC2x3 (2006), Industry Foundation Classes IFC2x Edition 3, http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc2x3-release

IFC4 (2013), Industry Foundation Classes IFC4 Official Release,
http://www.buildingsmart-tech.org/ifc/IFC4/final/html/index.htm
Jones, R. (1996), Research Methods in the Social and Behavioral Sciences, Second Edition,
Sinauer Associates Inc, Massachussets
Kiviniemi, A. (2005), Requirements Management Interface to Building Product Models, PhD Thesis,
Stanford University, California
Krippendorff, K. & Bock, M. A. (2009), The Content Analysis Reader, SAGE Publications
Lexi-Portal (2014), Leximancer software (online Version 4).
Available: https://www.leximancer.com/lexiportal/
Marchant, D. (2015), Capturing and integrating the design brief in building information models, PhD thesis, UNSW Australia
PAMPeR (2004), IFC Extension Project: FM-9 Portfolio and Asset Management - Performance Requirements (project leader Gerald Davis),
http://www.buildingsmart-tech.org/future-extensions/ifc_extension_projects/completed/fm-9-portfolio-and-asset-management-performance-requirements-pamper
Pena, W. with Caudill, W. & Focke, J. (1977), Problem Seeking: An Architectural Programming Primer, Cahners Books International, Boston
Schreier, M. (2012), Qualitative content analysis in practice. Sage Publications
Wade, J. (1977), Architecture, Problems and Purposes, Wiley-Interscience, New York