Integrating IndoorGML and CityGML for Indoor Space

Joon-Seok Kim, Sung-Jae Yoo, and Ki-Joune Li

Department of Computer Science and Engineering
Pusan National University, Pusan 609-735, South Korea
{joonseok,sjyoo,lik}@pnu.edu

Abstract. Recent progress on indoor positioning and mobile devices allows to provide indoor spatial information services such as indoor LBS or indoor disaster management. In order to realize these services, indoor maps are a crucial and expensive component of the system. For this reason, the interoperability among services and sharing indoor map and spatial information is a fundamental requirement of the indoor spatial information system. Several geospatial standards have been and being developed to meet this requirements, among which CityGML LoD 4 (Level of Detail 4) and IndoorGML are the most relevant ones for indoor spatial information. However the objectives and scope of these standards are different although their integration may give a synergy effect. In this paper, we discuss the issues on the integration of IndoorGML and CityGML LoD 4 and propose two methods: automatic derivation of IndoorGML data from CityGML LoD 4 data set and external references from IndoorGML instance to an object in CityGML data. The derivation and reference of external objects are based on the mapping relationship between feature types in CityGML and IndoorGML to be investigated in this paper. A simple prototype will be also presented, which has been developed to validate our methods.

Keywords: IndoorGML, CityGML, indoor spatial information, mapping relationships between IndoorGML and CityGML, derivation of IndoorGML from CityGML, external reference of IndoorGML to CityGML.

1 Introduction

With the progress of indoor positioning technologies and mobile devices such as smart phones, a number of indoor map and navigation services have been provided within large and complex buildings such as shopping malls [4]. For these services, the demand for indoor spatial information has been increasing and accordingly geospatial standards become important to share data and enhance the interoperability. There are three geospatial standards which may cover indoor space: CityGML [6][10] and KML 2.0 of OGC (Open Geospatial Consortium), and IFC (Industrial Foundation Classes)[1] of BuildingSmart, which provide standard data model and XML schema for visualization, geometric representation, and semantic properties of building components. In particular, the

D. Pfoser and K.-J. Li (Eds.): W2GIS 2014, LNCS 8470, pp. 184–196, 2014.

level of detail 4 (LoD 4) of CityGML is intended to describe the interior space of buildings. However they lack of features related with indoor space model, navigation network, and semantics for indoor space, which are critical requirements of most applications of indoor spatial information.

In order to meet the requirements, a working group for candidate standard of OGC, called IndoorGML IndoorGML[8] has been launched since 2012. The basic goals of IndoorGML are to provide a standard framework of semantic, topological, and geometric models for indoor spatial information. However IndoorGML is a complement of the existing standards rather than an independent one. Integration of IndoorGML with these existing standards raises as an important issue for two reasons. First, a part of IndoorGML data can be derived from data of existing standard specifications such as CityGML LoD 4 or IFC. Second, data set in IndoorGML may contain external references to indoor spatial objects defined in other data set such as CityGML dataset. Among the existing standards dealing with indoor space, we particularly focus on CityGML in this paper, since there are common feature types both in IndoorGML and CityGML LoD 4 and it is required to handle them in an integrated way.

In this paper, several issues will be discussed to integrate IndoorGML data and CityGML data. First we study how to derive IndoorGML data from CityGML LoD 4. Second, we investigate the correspondence between the feature types in CityGML LoD 4 and IndoorGML, and propose solutions to integrate IndoorGML and CityGML LoD 4 dataset via external references. The rest of this paper is organized as follows; in section 2, we explain the basic concepts of IndoorGML and CityGML. And in section 3, we investigate the correspondence between IndoorGML and CityGML LoD 4. We propose a method to derive IndoorGML data from CityGML and create proper network model for indoor navigation, which is a key part of IndoorGML data creation, in section 4. In section, we also propose a method to link an IndoorGML feature to a feature in CityGML via external reference. And we conclude the paper in the next section.

2 Related Work and Motivation

2.1 IndoorGML

IndoorGML is a standard data model to represent, store and exchange indoor spatial information and a XML application schema based on GML 3.2.1 [11]. Note that we refer to the version 0.8.1 of IndoorGML in this paper. While CityGML and IFC focus on feature types of building components such as roof, ceiling, floor, and wall, the main focus of IndoorGML is the representation of spaces in indoor, called *Cell*, which is the basic space unit in IndoorGML data model. Therefore IndoorGML provides a standard framework for representing geometry, network, and semantics of cells in indoor space. We briefly explain how to represent these aspects in IndoorGML.

– **Geometry of cell:** there are three options to represent geometry of cell as shown in Figure 1. The first option is to reference an object defined in other data set such as CityGML, which contains its geometric property. The second option is to include geometric property of cell within IndoorGML data, which is either a solid in 3D or a surface in 2D. The third option is not to include any geometry property of cell.

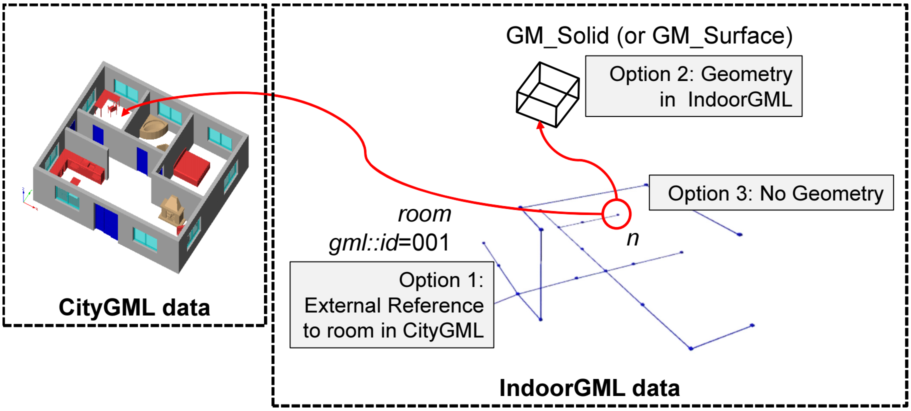

Fig. 1. Geometry of Cell of IndoorGML

– **Network of cell:** IndoorGML is composed of the core module and extension modules, the basic space model of the core module is called *structure space model*, as depicted in Figure 2. While the upper part of Figure 2 shows the primal space with 2D or 3D geometry and induced topology, the dual space is illustrated by the lower part, which represents the network structure called NRG (Node-Relationship Graph) [7]. The transformation from the primary space to dual space is explained by Poincaré duality [7], where a 3D volumetric object and a 2D boundary surface object between two 3D volumetric objects are transformed to a node and link respectively. Note that nodes in geometric NRG have (x, y, z) position data while no position data is included in logical NRG in Figure 2. The data model of NRG is depicted in Figure 3. Nodes and edges are represented as instances of `indoorCore::State` and `indoorCore::Transition` respectively in the model, which form a `indoorCore::SpaceLayer`. While `indoorCore::State` and `indoorCore::Transition` in Figure 3 form a network of dual space, `indoorCore::CellSpace` and `indoorCore::CellBoundary` represent objects in the primal space. It means that they may have inline geometric properties or reference to external objects in other data set.
– **Semantics of cell:** there are different semantical interpretations of an indoor space and each interpretation gives a different semantic model. While the core module of IndoorGML is a neutral model from any semantic interpretation, any extension may be defined on the core module to provide a

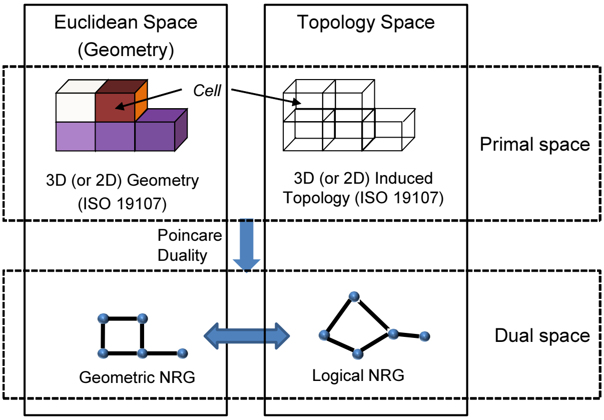

Fig. 2. Structured space model

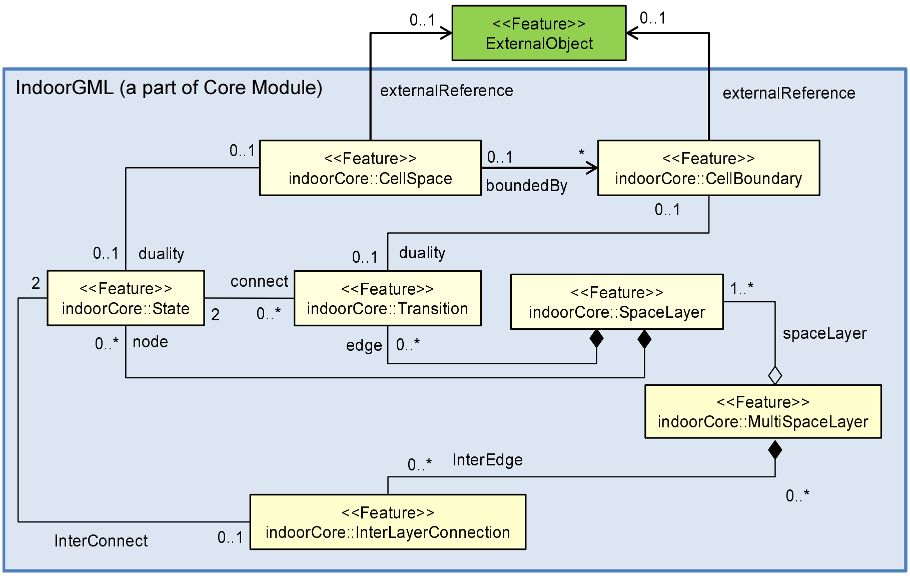

Fig. 3. Core module of IndoorGML

semantic context of model. In the current version of IndoorGML, an extension for a context of indoor navigation is defined as shown in Figure 4. In this data model, several feature types of indoor cells and boundaries are defined in terms of indoor navigation. For example, cells for movement space such as corridor, stairs, or elevator shaft are represented as `indoorNavi::TransitionSpace` in the indoor navigation module, while cells for staying such as rooms are represented as `indoorNavi::TransitionSpace`.

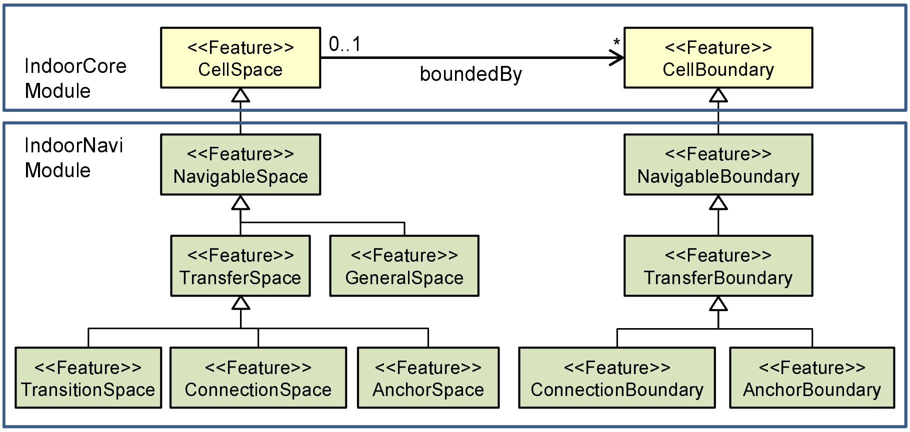

Fig. 4. Navigation module of IndoorGML

An instance from `indoorCore::MultiSpaceLayer` consists of multiple instances of `indoorCore::SpaceLayer`, each of which defines a different decomposition of indoor space as shown in Figure 3. For example, while a big hall is considered as a cell in a space layer, it can be partitioned into multiple small cells in another space layer. The inter-layer relationship between cells of different layer is defined as `indoorCore::InterLayerConnection` in IndoorGML.

2.2 CityGML

CityGML is an OGC standard for XML application schema to represent, store, and exchange 3D virtual city models. The most recent version of CityGML is version 2.0 and based on GML 3.2.1. It includes not only the core module and appearance module but also several thematic modules such as digital terrain, tunnels, bridges, and buildings. It also provides a notion of level of details (LoD) from LoD 0 to LoD 4, where LoD 4 aims to represent building interior space. In this paper, we focus on the integration of IndoorGML and CityGML for indoor space and consequently deal with LoD 4 of CityGML building module.

Figure 5 shows a simplified data model of LoD 4 of the building model. A building mainly consists of three basic components; first `BoundarySurface` such as walls, roofs, ceiling, and floors, second `Room` such as rooms and corridors,

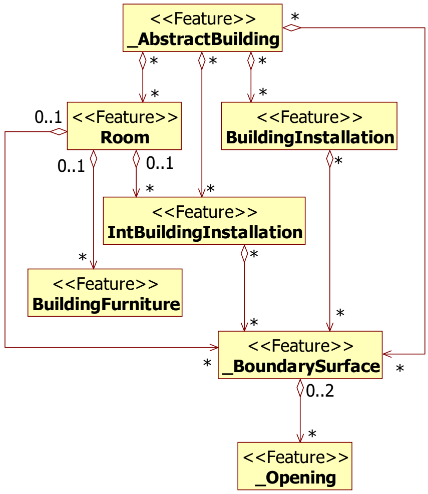

Fig. 5. Building model of CityGML

third `Opening` for windows and door. While `BoundarySurface` and `Opening` are geometrically defined as surface or multi-surfaces (`gml:MultiSurface` in GML), the geometry of `Room` is defined as either inline solid (`gml:Solid` in GML) or a set of `BoundarySurface`. If the geometry of a room is defined as a set of `BoundarySurface`, it must be a closed space. For example, if there is no physical boundary between kitchen and living room, we need to make a virtual boundary by using `ClosureSurface` to make each room a closed space. It implies that the connectivity between rooms is found either via `Opening` or `ClosureSurface` in CityGML.

2.3 Motivation

Linking IndoorGML and CityGML is useful for several reasons. First, a large part of IndoorGML data can be derived from CityGML data. Second, the integration of IndoorGML and CityGML via external references compensates the weakness of each standard. However, few works have been done on rules or guidelines for linking and integrating IndoorGML and CityGML. The goals of this paper are to propose a derivation method of IndoorGML data from CityGML data and to discuss the issues and possible solution for integrating two data sets in both standards via external reference of IndoorGML.

3 Corresponding between IndoorGML and CityGML

In order to explore the integration issues between IndoorGML and CityGML, we need to investigate the relationships between feature types in both data models.

While `Room` feature type of CityGML corresponds `indoorCore::CellSpace` of IndoorGML data model, we have no further classification of room types in CityGML. However it is required to tell which subtypes of `indoorCore::CellSpace` it belongs to in IndoorGML, for example whether `indoorNavi::GeneralSpace` or `indoorNavi::TransitionSpace`.

In this paper, we propose a mapping between feature types of IndoorGML and CityGML based on the ontology in [3] and the code list defined in Annex C of CityGML[10]. According to the ontology in [3], the `Room` in CityGML corresponds to either *room* or *passage* as shown in Figure 6. Then *room* and *passage* are mapped to `indoorNavi::GeneralSpace` and `indoorNavi::TransitionSPace` respectively, where *corridor*, *stairway*, *escalator*, *elevator*, *moving walk*, *ramp*, *lobby* belong to *passage*.

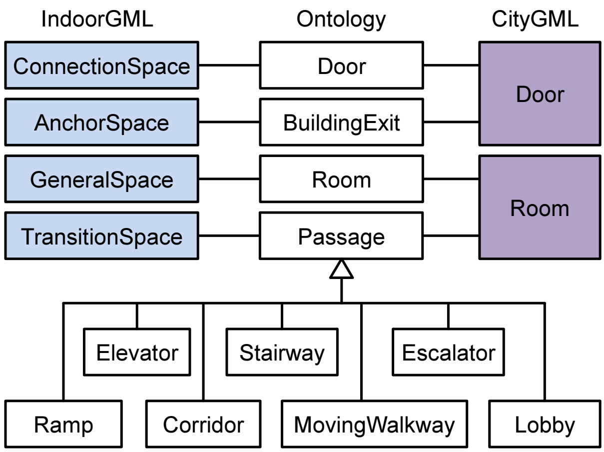

Fig. 6. Mapping relationships between IndoorGML and space types

A more delicate mapping between IndoorGML and CityGML is found between `Door` in CityGML and feature types of IndoorGML.

- **anchor:** if the the instance of `Door` is a gate connecting indoor and outdoor spaces, then it is considered as an anchor rather than a door in IndoorGML and otherwise, it is considered a door.
- **thin door vs. thick door:** while `Door` is geometrically defined as a multi-surface in CityGML, it is either a surface or a solid in IndoorGML depending on the door model, that is, whether thin door model or thick door model as shown in Figure 7. For example, ‘D1’ and ‘Cell D1’ in Figure 7 are represented as thin door and thick door model respectively. If thin door model is employed in IndoorGML, `Door` of CityGML is mapped to `indoorNavi::ConnectionBoundary` of IndoorGML, otherwise, it is mapped to `indoorNavi::ConnectionSpace` in IndoorGML.

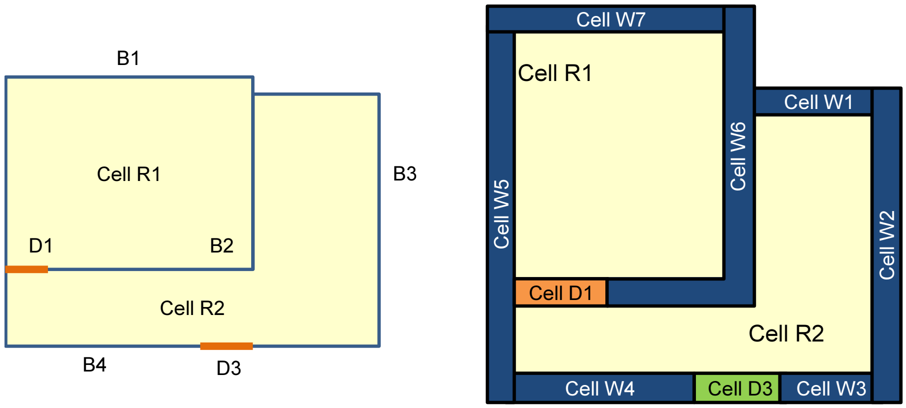

Fig. 7. Thin door model vs. thick door morel

The mapping relationships between feature types of IndoorGML and CityGML are summarized in table 1.

Table 1. Mapping relationships between feature types of IndoorGML and CityGML

CityGML Feature Type(CodeList)	IndoorGML Feature Type
Room(stairs)	TransitionSpace
Room(escalator)	TransitionSpace
Room(elevator)	TransitionSpace
Room(lobby)	TransitionSpace
Room	GeneralSpace
Door	ConnectionSpace
ClosureSurface	ConnectionBoundary

4 Derivation from CityGML to IndoorGML

4.1 Generating Reference of IndoorGML to CityGML

Based on the mapping relationships between IndoorGML and CityGML, we can generate features of IndoorGML data from CityGML data, which belong to either `NavigableSpace` or `NavigableSpaceBoundary`. Then the external references to features in CityGML data set are to be included in IndoorGML data. For this external reference to CityGML data, a unique feature identifier must be included in each object in CityGML.

In order to clarify this mapping relationship, we propose guidelines to identify the correspondence as follows.

- **CodeList:** the definition of `Room` in CityGML is broad and it may be not only room such as bed room, living room, but also corridor or stairway. Therefore we need more information to specify the correspondence. Fortunately a code list for function and usage of `Room` is given in CityGML. For example five code lists for stairway, escalator, elevator, lobby defined in CityGML correspond to `indoorNavi::TransitionSpace` of IndoorGML. Therefore the code list enable to precisely specify the correspondence between feature type in IndoorGML and instances of CityGML. In the case where the code list of `Room` in CityGML is missing or belongs to unknown code lists, it implies that it should be mapped to merely `indoorNavi::NavigableSpace`.
- **Control value:** A level of control value is defined by space syntax theory in terms of connectivity of a cell. in [5], a method to find a space with a high probability for being rooms except living rooms is proposed by using the degree of connectivity. In [3], the concept of space syntax was introduced, which allows to investigate properties of space cell. If the control value of a cell in space syntax is less than 1, then it is considered as an instance of `indoorNavi::GeneralSpace` such as office or bed room. Otherwise it may belong to `indoorNavi::TransitionSpace`, since it is connected with other cells and therefore used as a passage between cells.

4.2 Automatic Generation of Geometry of *State* and *Transition*

In this subsection, we discuss the automatic derivation of `indoorCore::State` and `indoorCore::Transition` from instances in CityGML. Based on the mapping relationships and generation of external references, we see how each instance in CityGML is mapped to an IndoorGML instance, which eventually belongs to either `indoorCore::CellSpace` or `indoorCore::CellBoundary` of the primal space.

In this section, we focus on geometric graph where the positions of `indoorCore::State` and `indoorCore::Transition` are specified. The position of `indoorCore::State` is easily computed as a centroid of `indoorCore::CellSpace`. However in order to generate `indoorCore::Transition`, we need to check the connectivity between cells from CityGML data set. This process is carried out with two steps.

- First, given an instance of `Room` in CityGML data, we find a `Door` instance belonging to `BoundarySurface` or `ClosureSurface` of the room.
- Second, we find `Room` instances which share the `Door` or `ClosureSurface` instance found at the first step. Then we have an instance of `indoorCore::Transition` connecting two `indoorCore::State` instances which correspond with the `Room` instances of CityGML at the first step.

The geometry of `indoorCore::Transition` is determined as a straight line connecting two `indoorCore::State` as shown in Figure 8. This may explain the topological structure of indoor space but is insufficient to figure out the

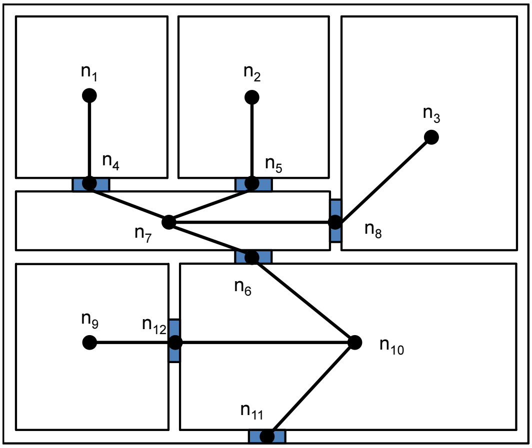

Fig. 8. Topographic layer derived from CityGML

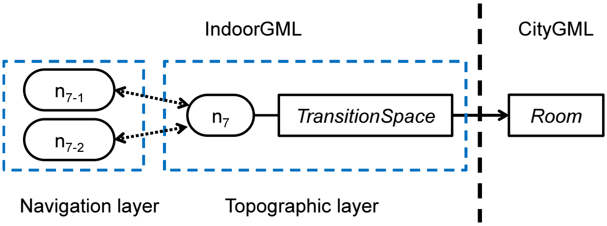

Fig. 9. Connection between *State* and *CellSpace*

navigation routes and therefore to compute optimal route between two points in indoor space. For example, the route between n_2 and n_9 quites differs from the navigation route of ordinary pedestrians. This strange route is explained by two reasons. The first reason comes from a big hall or long corridor with several doors as n_7 in Figure 8. The second reason is due to case that the shape of cell is concave and the straight line between two `indoorCore::State` instances passes through walls.

In this paper, we propose a solution for this problem by introducing an extra layer for navigation as shown in Figure 9. The `indoorCore::State` instances of the original space layer is split into multiple instance to improve the route. For example, a `indoorCore::State` instance n_7 is split to two instances n_{7-1} and n_{7-2} as shown in Figure 10 then the route from n_2 to n_9 becomes more ordinary than the original one in Figure 8. More detail algorithm of node splitting is found in [12].

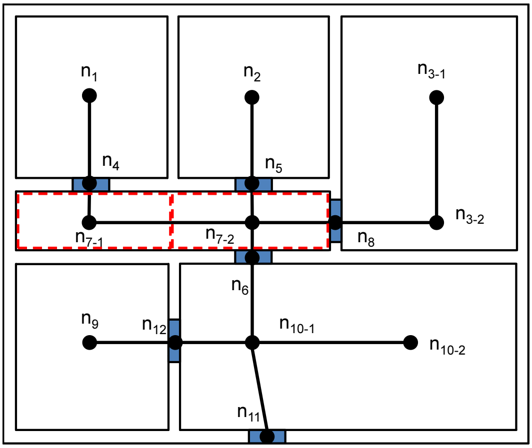

Fig. 10. Navigation layer after subspacing

5 Mapping Cardinality for External Reference

In this section, we discuss how to generate external references from IndoorGML instance to objects in CityGML. The mapping relationships between IndoorGML and CityGML are discussed in section 3. There are cases where the mapping cardinality from IndoorGML instance to CityGML object is not one-to-one. Figure 11 shows an example where two rooms in CityGML correspond with three cells of IndoorGML. However one-to-many mapping and many-to-many mappings are not allowed for the external reference from IndoorGML since only one external reference can be specified for a cell in IndoorGML, while one-to-one or many-to-one are allowed.

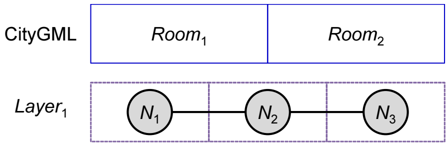

Fig. 11. M:N relationship in spaces

In this paper, we propose a method to resolve this mapping cardinality by introducing an *overlapping layer* (Figure 12) or a *subspacing layer* (Figure 13). Figure 12 shows an example for the first option that an additional layer $Layer_2$ is introduced to guarantee one-to-one mapping between CityGML and IndoorGML (n_4 to $Room_1$ and n_5 to $Room_2$). Then we define the inter-layer connection between $Layer_1$ and $Layer_2$, where the topological property of the inter-layer connection is `equal`, `overlapping` or `contain`.

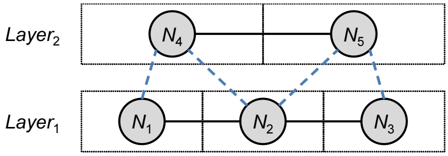

Fig. 12. Overlapping layers

An alternative option for extra layer is shown in Figure 13, where any intersection of `Room` objects of CityGML and cell of IndoorGML becomes an separate cell of IndoorGML. In this case, the topological property of inter-layer connection becomes either `contain` or `equal`. Both of these options can be applied to remove the one-to-many or many-to-many possibility.

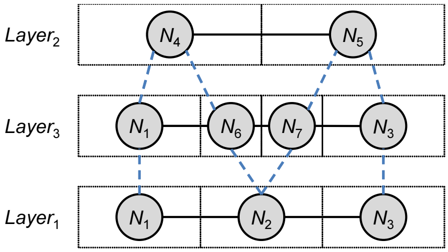

Fig. 13. Sharing the subspaced layer

6 Conclusions

Two geospatial standards - CityGML LoD 4 and IndoorGML - have been developed by OGC (Open Geospatial Consortium) to provide the interoperability among indoor spatial information systems. While CityGML aims to provide a framework of 3D city model including the interior space, IndoorGML mainly focuses on indoor navigation in terms of cellular space model. In order to overcome the mismatches between two geospatial standards, they may be served as a complement for each other and their integration is useful for many indoor spatial information service areas.

In this paper, we discussed the issues on the integration and proposed methods and guidelines for the integration. The main contributions of the paper are

- First the mapping relationships between feature types of CityGML LoD 4 and IndoorGML, which is served as a fundamental understanding for the integration.
- we presented methods and guidelines of automatic derivation of IndoorGML instance from CityGML LoD dataset. Even though some restrictions should be respected by CityGML LoD 4 dataset for successful derivation, we can accurately build IndoorGML dataset with ease.

– we discussed the issue on mapping cardinality between CityGML and IndoorGML for external references and proposed a solution by using the multi-layered space model of IndoorGML.

A prototype has been developed to validate our approaches and proved that the approaches are feasible but require more extensions. In particular, a part of derivation of IndoorGML data is automatic but a manual work is still required in several cases. Therefore more future works are required to strengthen the automatic derivation and reduce the part for manual work. And only integration between CityGML and IndoorGML was handled in the paper. However the integration between IndoorGML and IFC is to be done for the future work since IFC is widely used not only architectural engineering but also geospatial information area and IFC can be used as a source of raw data in many cases.

Acknowledgment. This research was supported by a grant(11 High-tech G11) from Architecture & Urban Development Research Program funded by Ministry of Land, Infrastructure and Transport of Korean government.

References

1. buildingSMART, IFC4 (Industrial Foundation Classes XML 4) RC4, http://www.buildingsmart-tech.org/specifications/ifcxml-releases/ifcxml4-release
2. Computational Geometry Algorithms Library, http://www.cgal.org
3. Hiller, B., Hanson, J.: The Social Logic of Space. Cambridge University Press (1984)
4. Indoor Google Maps, http://maps.google.com/help/maps/indoormaps
5. Kim, J.S., Han, Y.S., Li, K.J.: K-anonymity in Indoor Spaces Through Hierarchical Graphs. In: Fourth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness, pp. 21–28 (2012)
6. Kolbe, T.H., Groger, G., Plumer, L.: Citygml interoperable access to 3d city models. In: 1st International Symposium on Geo-Information for Disaster Management, pp. 21–23 (2005)
7. Lee, J.: 3D GIS for Geo-coding Human Activity in Micro-scale Urban Environments. In: Egenhofer, M., Freksa, C., Miller, H.J. (eds.) GIScience 2004. LNCS, vol. 3234, pp. 162–178. Springer, Heidelberg (2004)
8. Lee, J., Li, K.J., Zlatanova, S., Kolbe, T.H., Nagel, C., Becker, T.: Requirements and Space-Event Modeling for Indoor Navigation. OGC 10-191r1 (2010)
9. Li, K.J., Yoo, S.J., Han, Y.S.: Geocoding Scheme for Multimedia in Indoor Space. In: 20th International Conference on Advances in Geographic Information Systems, pp. 434–437 (2013)
10. Open Geospatial Consortium, OGC City Geography Markup Language (CityGML) Encoding Standard, Version 2.0, OGC 12-019 (2012)
11. Open Geospatial Consortium, OpenGIS Geography Markup Language (GML) Encoding Standard Version 3.2.1, OGC 07-036 (2007)
12. Tsetsos, V., Anagnostopoulos, C., Kikiras, P., Hasiotis, P., Hadjiefthymiades, S.: A Human-centered Semantic Navigation System for Indoor Environments. In: International Conference on Pervasive Services, pp. 146–155 (2005)