A Fine-grained Geospatial Representation and Framework for Large-Scale Indoor Environments

by

Jonathan Battat

S.B. Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 2007

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING AT THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2010

Signature of Author: _____

Electrical Engineering and Computer Science
February 5, 2010

Certified by: _____

Seth Teller
Professor of Computer Science & Engineering
Thesis Supervisor

Accepted by: _____

Christopher J. Terman
Senior Lecturer
Chairman, Department Committee on Graduate Students

A Fine-grained Geospatial Representation and Framework for Large-Scale Indoor Environments

by

Jonathan Battat

Submitted to the Department of Electrical Engineering and Computer Science
on February 5, 2010, in partial fulfillment of the requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science.

ABSTRACT

This thesis describes a system and method for extending the current paradigm of geographic information systems (GIS) to support indoor environments. It introduces features and properties of indoor multi-building environments that do not exist in other geographic environments or are not characterized in existing geospatial models, and proposes a comprehensive representation for describing such spatial environments. Specifically, it presents enhanced notions of spatial containment and graph topology for indoor environments, and extends existing geometric and semantic constructs. Furthermore, it describes a framework to: automatically extract indoor spatial features from a corpus of semi-structured digital floor plans; populate the aforementioned indoor spatial representation with these features; store the spatial data in a descriptive yet extensible data model; and provide mechanisms for dynamically accessing, mutating, augmenting, and distributing the resulting large-scale dataset. Lastly, it showcases an array of applications, and proposes others, which utilize the representation and dataset to provide rich location-based services within indoor environments.

Thesis Supervisor: Seth Teller
Title: Professor of Computer Science & Engineering

Acknowledgements

The journey to the end of the road of this thesis was an arduous one, and could not be possible without acknowledging the support, encouragement, cooperation, and assistance of certain people.

Of course, to the RVSN, BMG, OIL, TBH, and Nokia families, who provided the social and intellectual stimulation that made this experience an enjoyable one.

To my thesis advisor, Professor Seth Teller, who besides providing unparalleled academic, intellectual, and professional guidance and insight, also maintained an unyielding patience that one might expect from a parent.

To my parents, whose love and support, emotional and material, and unyielding patience worthy of sainthood, sustained me through this experience.

To the one and only Laura, my fiancée and soon-to-be wife, who supplied the boundless wellspring of love and motivation, more than she knows, that made this thesis possible, and to whom this work is dedicated.

And lastly, to God, from whom this entire body of work emanated.

Table of Contents

1 Introduction

Mapping technology has matured to the point where much of the developed outdoor world has been captured, modeled, and is readily available for use in location-aware applications. If we are to advance mapping technology to the next logical stage, and facilitate the fine-grain representation of complex indoor environments, then a thorough evaluation of current methods and a rich assessment of next-generational needs must be undertaken. This thesis attempts to address those challenges.

This chapter more thoroughly introduces the motivations behind this body of work, its contributions, some background information, and related work.

1.1 Motivations

As electronic devices shrink, computational capabilities grow, and costs persist in their

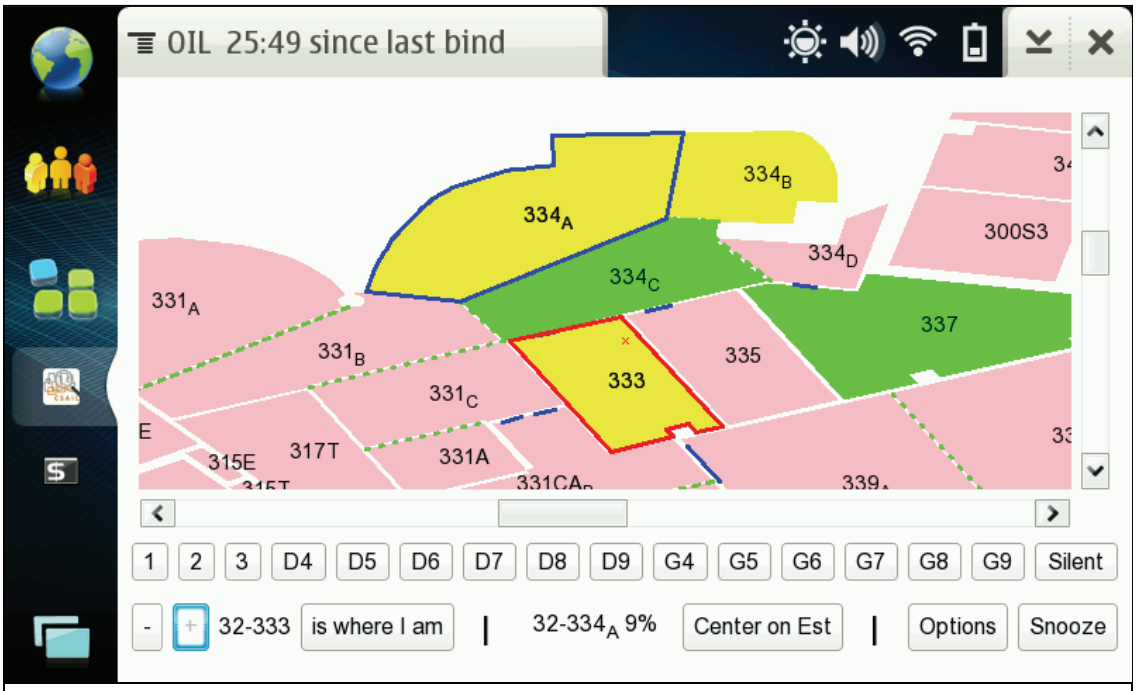

Figure 1. A map of an interior environment, as used in a mobile, location-based service.

rapid decline, personal mobile communications and data services continue to grow exponentially. Personal mobile devices offer many features that only a decade ago were inconceivable or simply not mature enough for broad adoption. These features include the common contenders, such as voice, video and data communications, personal data and life management, gaming, and media consumption, but also less predictable ones, such as social networking, media generation and location-based services (Figure 1), among others. Of these, location-based services had until recently received only limited attention. More recently, with the advent of broadband data networks, cheaper and more pervasive GPS hardware, and online map services that offer global geographic information systems in real-time and on-demand, location-based services have begun to penetrate consumer markets (Figure 2).

As location-based applications and services come online and become as ubiquitous as other

Figure 2. A typical GIS-derived map of an outdoor environment, as seen in an actual GPS device. [TomTom2009]

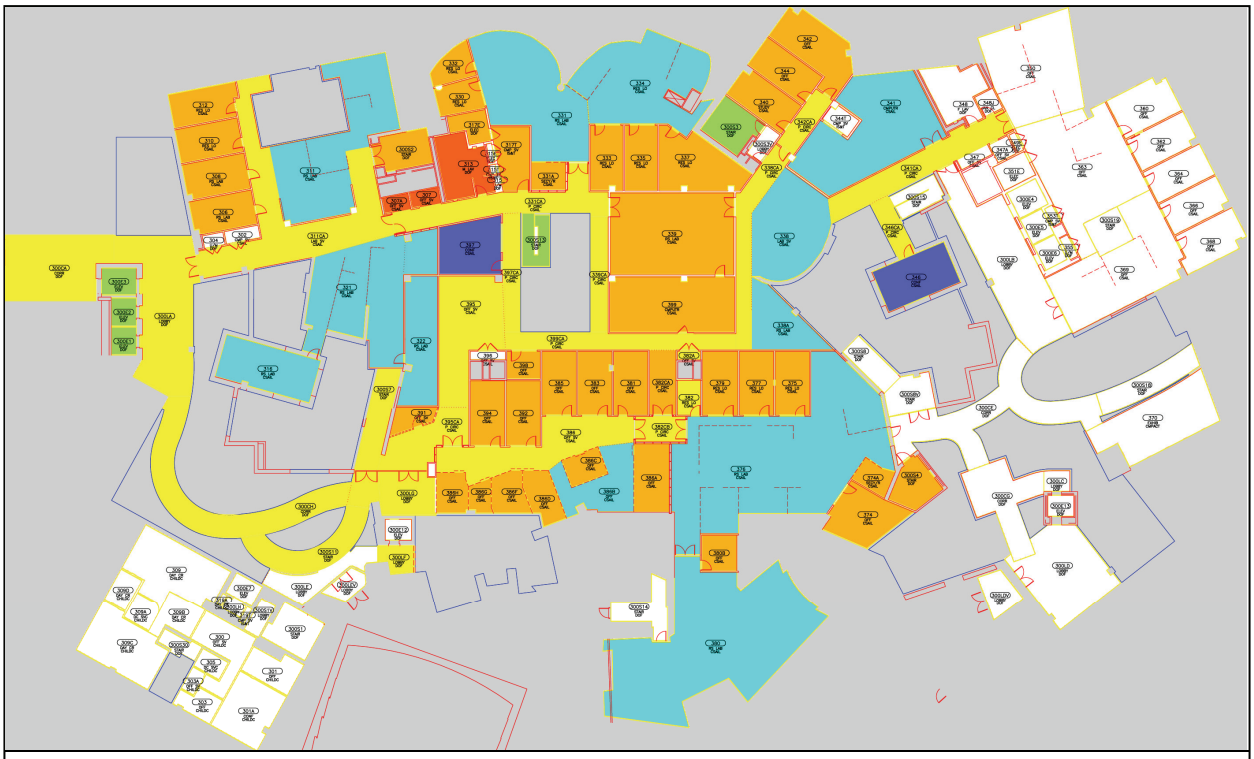

Figure 3. A fine-grained map of an indoor environment that fully characterizes indoor geospatial features such as space connectivity, containment relationships, and extensive semantic attributes.

mainstream technologies, demand for location technology will continue to grow. This assessment notwithstanding, however, it is necessary to point out that existing location-based services have limitations. The majority of location-based services provide low-precision positioning, work only outdoors, and perform poorly in dense urban areas and "urban canyons." Furthermore, the geographic information system (GIS) models that these services rely on are themselves coarse-grain, low resolution data sets. What they offer in breadth, namely ubiquity, excellent coverage of many developed regions, they lack in depth and definition, representational capability, and content richness. More concretely, existing GIS models lack the ability to properly and completely describe fine-grain indoor environments. If location-based services are to continue their growth in utility, usage, and popularity, the GIS models they rely on must provide the representational capability to describe the type of venue where people spend the bulk of their time: indoors. This thesis addresses the representational gap in the ability of current GIS systems to properly and comprehensively modeling fine-grain indoor environments (Figure 3).

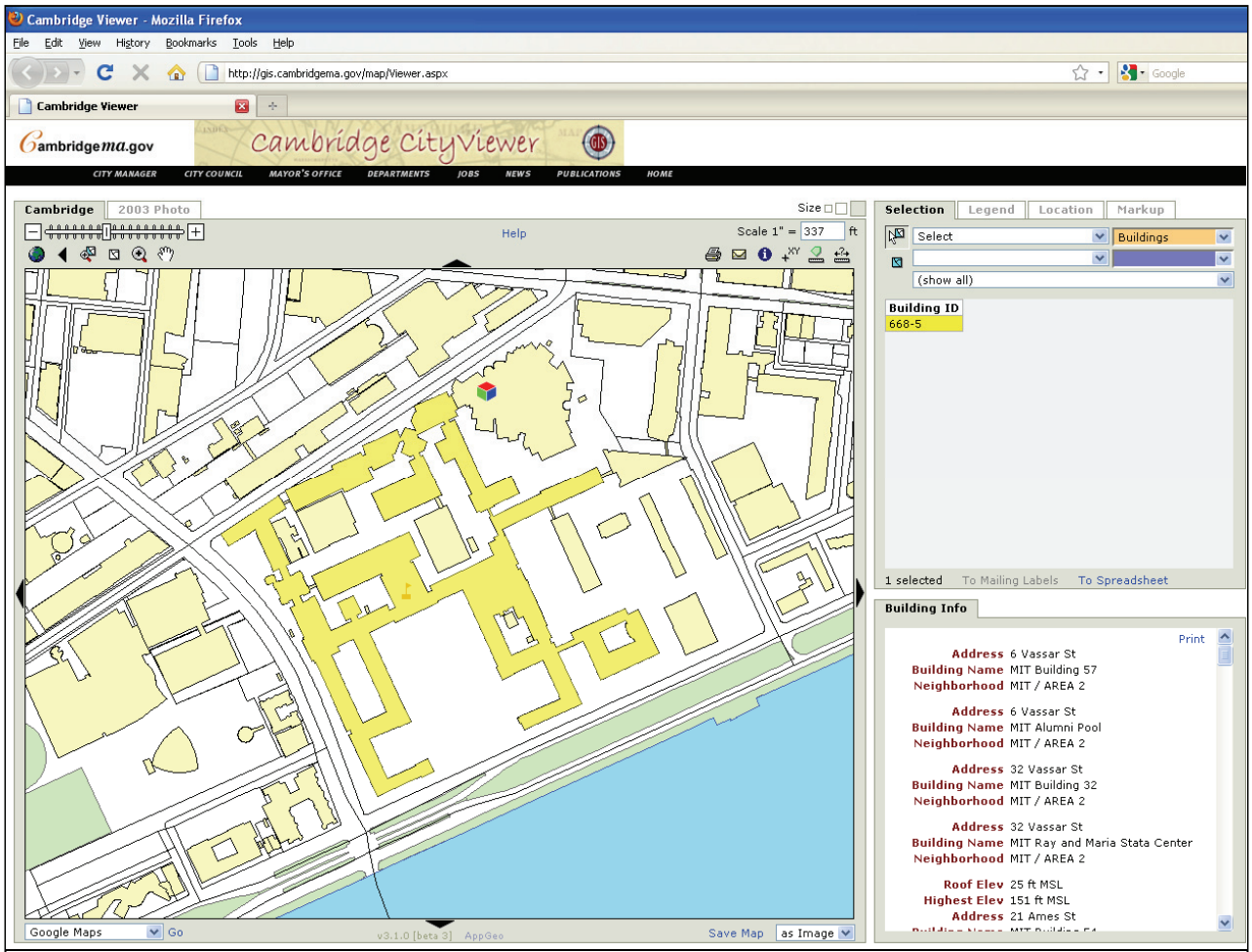

Figure 4. A typical GIS model, designed principally for the representation of outdoor environments. Current GIS limits its support of geospatial features to relatively coarse-grain entities, such as entire buildings.

1.2 Contributions

This thesis describes a system and method for extending the current paradigm of geographic information systems (GIS) to support indoor environments. Existing GIS is designed principally for the representation of outdoor environments (Figure 4). If we are to properly capture and fully characterize indoor geospatial features, existing GIS representations must be further developed and new geospatial notions introduced.

This body of work presents features and properties of indoor multi-building environments that do not exist in other geographic environments or are not characterized in existing geospatial

models, and puts forward a comprehensive representation for fully describing such environments. Namely, the discussion of Chapter 2 focuses on four components of indoor geospatial environments and studies how current geographic models fare when applied to them. Some of these representational components, such as geometric and semantic features, have close parallels in indoor and outdoor environments, and thus existing GIS models need only undergo minor tweaking to support such indoor properties. Other representational components, however, such as connectivity between spatial entities (i.e. graph topology), and notions of spatial containment (hierarchy), are either inadequately represented in current models or exhibit aspects that require the introduction of new geospatial notions in order to confront the additional demands of indoor environments. Chapter 2 highlights and addresses these various issues and presents a comprehensive indoor geospatial representation.

Chapter 3 describes and demonstrates a framework for populating such a model, using a college campus as a prototypical input environment. A method to automatically extract indoor spatial features from a corpus of semi-structured digital floor plans is presented, and a concrete data storage architecture is proposed. Mechanisms for dynamically accessing, mutating, augmenting, and distributing the resulting large-scale dataset are outlined.

Lastly, Chapter 4 showcases an array of applications that utilize the geospatial model to provide rich location-based services to users within indoor environments.

1.3 Background

Development of geographic information systems is currently dominated by coarse-grain **geospatial models**[*] designed to represent large geographic environments. Examples of such

[*] Bold terms are defined in the glossary

environments are nations and their defining regional borders, networks of highways, and municipal districts. Relatively scant attention has been given to the representation of indoor environments, where people in the developed world spend the bulk of their time, and where the bulk of **location-based services** will be used.

The representation of indoor environments poses interesting problems that rise above and beyond the complexity of outdoor GIS representations. Of course, there are clear parallels between outdoor and indoor geospatial environments; each indoor environment can be considered a microcosm of a large geographic region, with landmarks, destinations, spatial containment, and links between spaces within that environment. Indoor environments, however, include additional concrete and abstract elements that do not have parallels in outdoor environments. For example, indoor environments incorporate an additional spatial dimension, both geometrically, in height and altitude, and topologically, in the connectivity of spaces across different floors within a building. It is true that outdoor areas may also contain a z-component, such as in height above or below sea-level, or in stacked street levels (bridges, overpasses, and the like). However, in representing such features, the conventional GIS map typically projects them onto a flat surface,

Figure 5. A side-by-side comparison of a GIS representation for an outdoor geospatial environment, and an actual image of the same environment. As can be seen, the environment is multi-leveled and vertically-spanning, but poorly modeled as such in the GIS representation due to lack of necessity.

effectively modeling the environment as a two-dimensional plane; interactions between points on the map do not typically account for this extra dimension (Figure 5).

For indoor environments, on the other hand, both height and vertical connectivity are critical components of the model. Geometrically, the additional spatial dimension in indoor environments usually finds its way into the representation via "stacking," or layering 2D areas along the vertical axis, thereby creating a "two-and-a-half" dimensional arrangement (Figure 6).

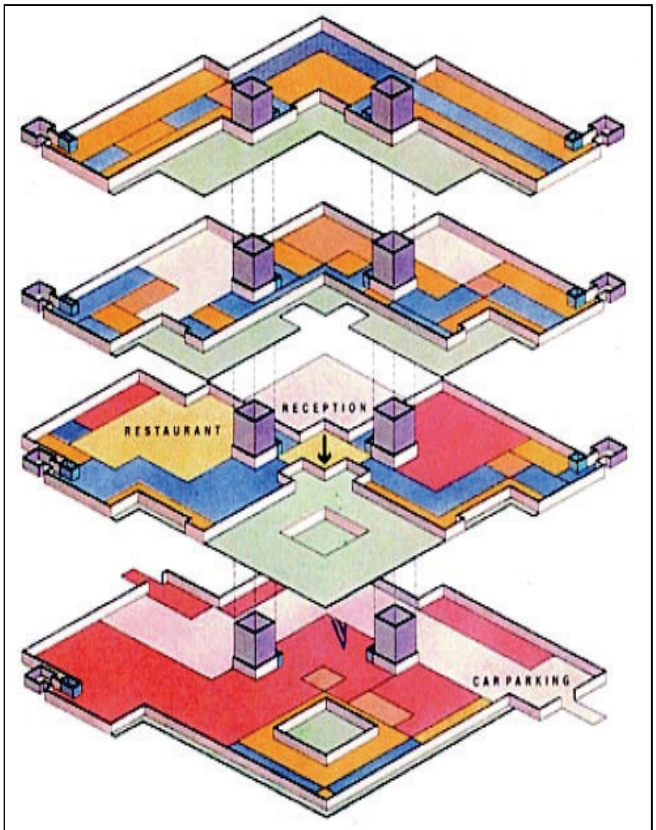

Figure 6. Individual two-dimensional floors can be stacked vertically in order to more authentically represent an actual indoor building environment. [Niederauer2003]

This is accomplished by introducing visualization support for the *z*-component, and by adding a hierarchical spatial containment capability that allows for layering spatial regions. Both of these components are essential for a comprehensive indoor GIS model.

Another difference between outdoor and indoor environments is in the complexity and wealth of topological relationships present. Outdoor environments, such as road networks, can be sufficiently described using one-dimensional line segments. The notion of **adjacency**, or the connectivity between proximal spatial entities, is simple. Landmarks are often described as points, which are linked by directed line segments, and paths are formed through the traversal of these line segments (Figure 7). The result can be thought of as a 1D manifold. Indoor spaces, on the

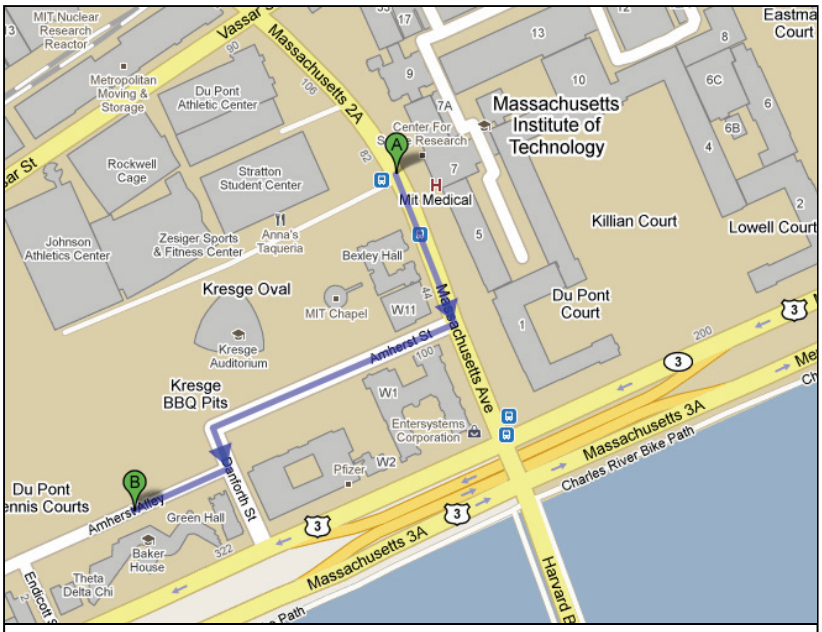

Figure 7. Typical GIS models represent landmarks as points, and the connections between them as directed line segments.

other hand, include regions with two-dimensional structure, and adjacencies that span all three dimensions. Furthermore, the variety of **adjacency types** is greater. For example, at the simplest level, the 3D nature of indoor environments already introduces the notion of vertical and horizontal adjacency. Vertical adjacencies include stairs, elevators and ramps that connect multi-level spatial environments. Horizontal adjacencies include **explicit** connections, such as doorways, and **implicit** connections, such as non-physical administrative boundaries that separate two spaces (such as where two corridor segments abut). Similarly, because paths can traverse many region types, such as intra-floor, intra-building, inter-building, and outdoor regions, each cross-region adjacency type can be noted as well, further adding to the representational demands of indoor environments.

Another core difference between the current representational capabilities of (outdoor) geographic information systems and the requirements of multi-building indoor environments is the notion of **spatial containment** or hierarchy. This concept can be easily understood with a simple example. A representation of Earth might model the planet as being at the root of a containment hierarchy. Earth contains continents; continents contain countries, countries contain states, states contain cities, and so on. These relationships are known as containment relationships,

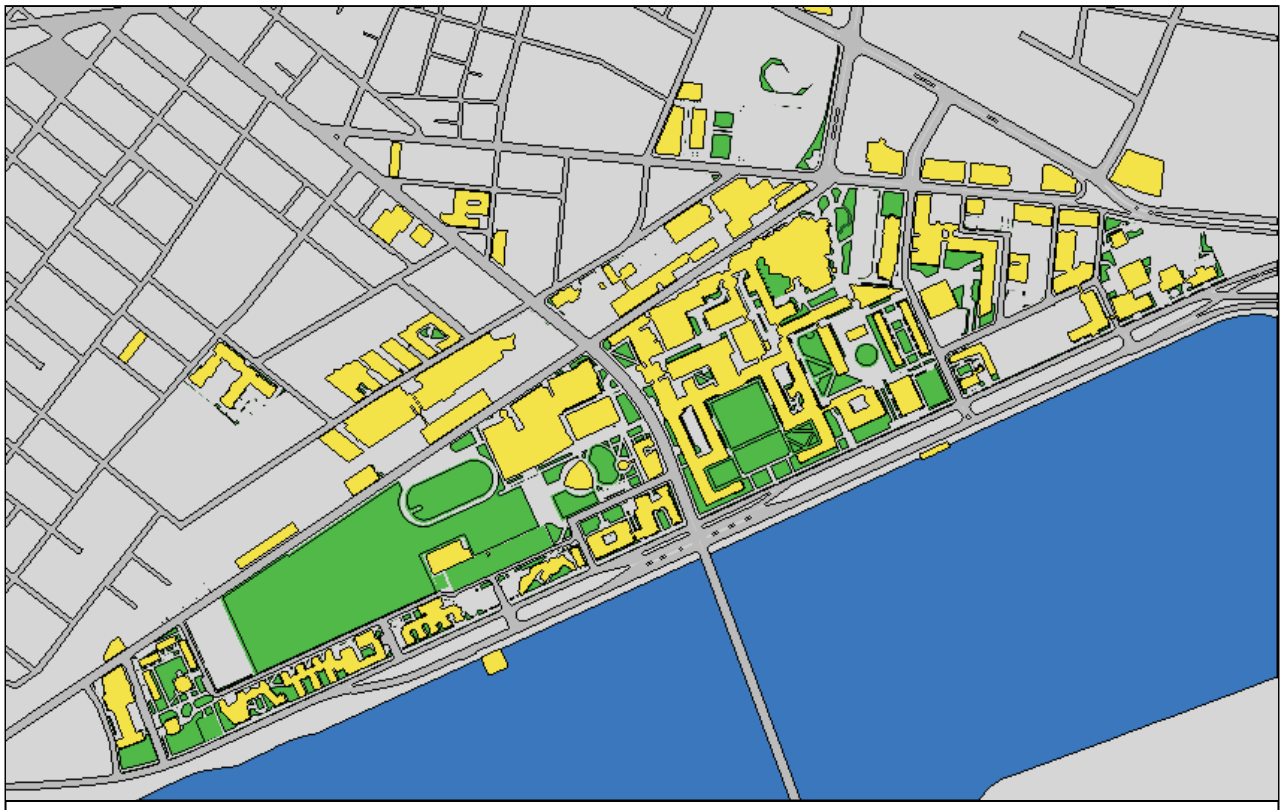

Figure 8. A visualization of GIS-modeled *layers*, with each layer depicted as a different color. Although this model conveys an ostensible richness, layers lack a basic ability to express broader hierarchical depth, such as that found in any multi-level building environment. Other drawbacks are discussed in section 2.4.

with each region being a container, or a *parent* to sub-regions, or *children*. Of course, this concept clearly exists in outdoor spatial environments as well as those indoors. However, support for spatial hierarchy in conventional GIS is currently limited. Rather than model explicit containment relationships, GIS defers these relationships to more primitive abstractions known as "layers." Layers allow for geospatial object segmentation based on semantic properties (e.g. by grouping objects into *Landscapes* or *Buildings*, as in Figure 8). This provides a kind of hierarchy, making it convenient to operate on or store such information. However, these limited abstractions do not generally encapsulate the notion of containment, which becomes increasingly important in indoor environments with buildings, floors, and rooms.

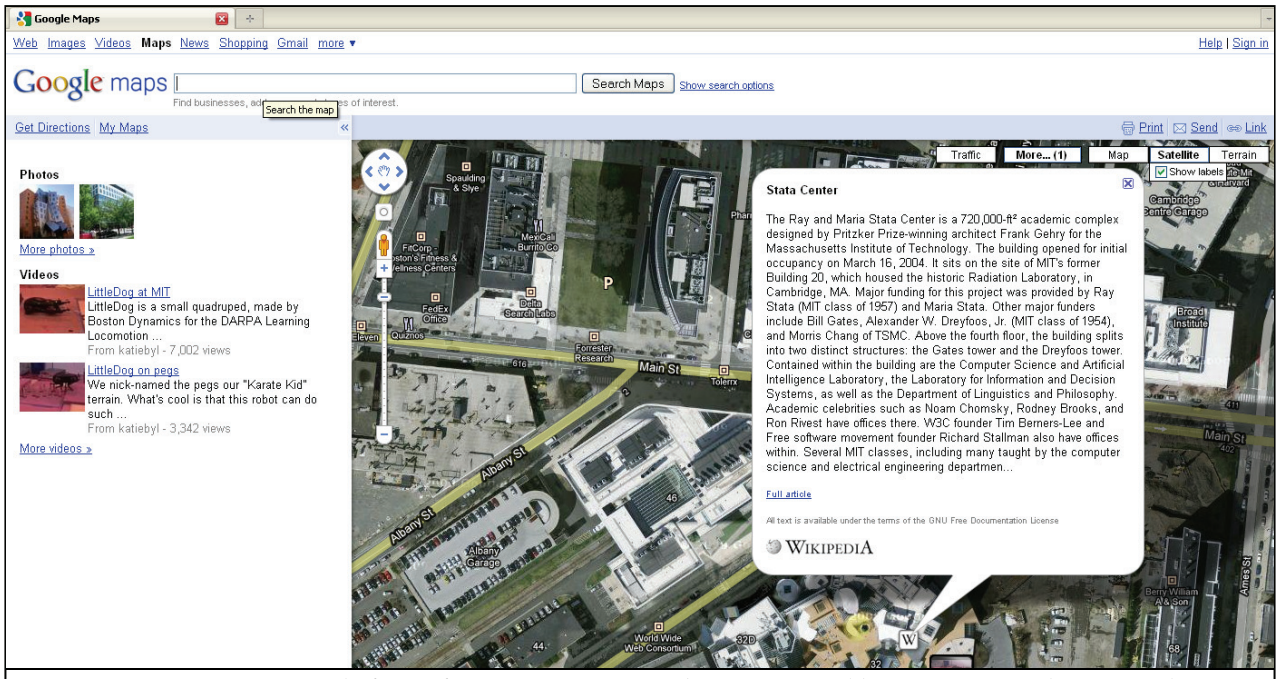

Figure 9. Geo-tagging is a simple form of semantic annotation that is supported by current GIS. Above, Google Maps is integrated with Wikipedia to provide a semantic annotation for the latitude and longitude coordinates of the Ray and Maria Stata Center at MIT.

The geospatial model presented here provides support for rich and arbitrary **semantic** annotations. Semantic properties are not a unique aspect of indoor environments, nor is support for such properties completely lacking in existing GIS. For example, the recent emergence and growth in popularity of crowd-sourced, wiki-style location annotations or geo-tags demonstrates s semantic representational capability (Figure 9). However, these annotations are basic. Existing semantic annotations reference only geometric points, or simple polygons at best; they can be grouped only in elementary ways (e.g. *user-contributed tags*), and topology grammar currently makes very little use of them (e.g. one-way streets). An improved method for representing semantic properties would allow for the annotation of every geospatial object represented in the model, as well as for the hierarchical and topological relationships between them. An indoor geospatial model without such formal semantic support would be incomplete.

1.4 Related Work

Evidence of environmental mapping by humans can be traced as far back as the hunter-gatherer periods of the pre-Neolithic era 17,000 years ago [Shekhar2003]. The need to visualize spatial environments, and subsequently attach meaningful properties to them arose as early humans attained a situational awareness of their environment. Hunter-gatherers, for example, with their nascent intelligence, could record and follow prey migration patterns using drawn maps. It was at this point in history that the most basic form of GIS, the linking of geographic image to semantic attributes, was invented.

With the advent of sophisticated war planning, resource management, urban planning, and other logistical methods that demanded faithful geographic representations coupled with rich semantic knowledge, more advanced forms of geospatial modeling emerged, incorporating both conceptual and physical innovations. The notion of cartographic layering was derived in order to

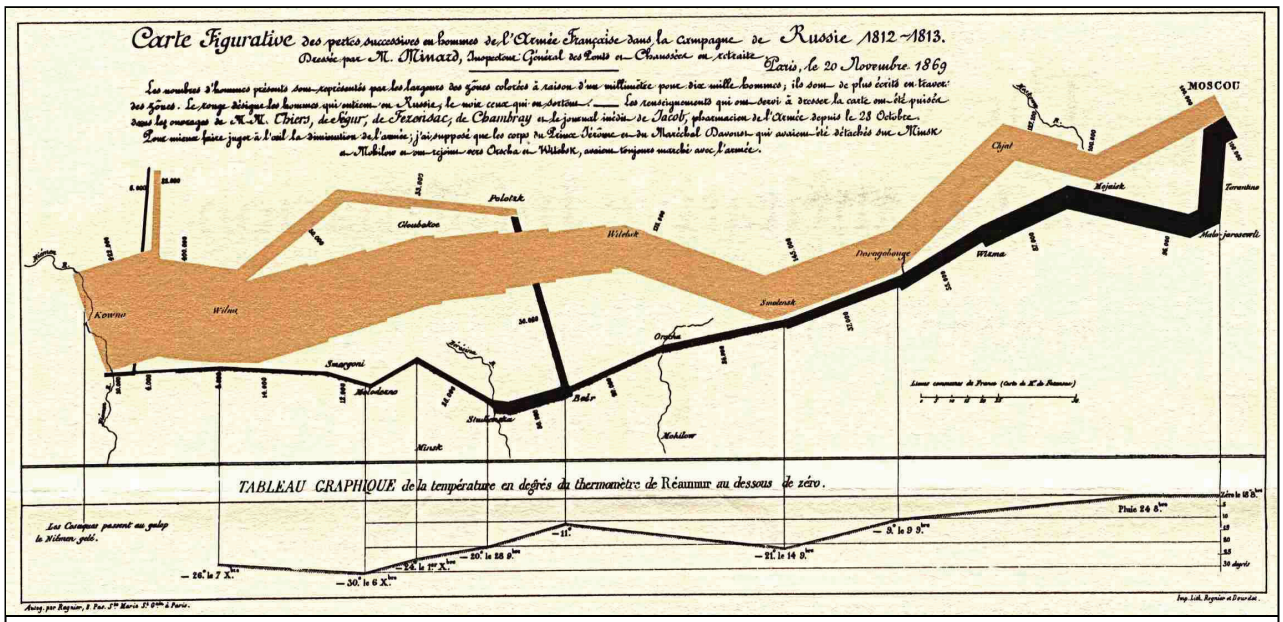

Figure 10. Minard's *Carte figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie1812-1813*, a map of successive losses in men of the French Army in the Russian campaign of 1812-1813. Minard was a pioneer in the field of information graphics, a precursor to GIS. [Tufte1983]

group similar geographic features and to analyze non-geographic phenomena. Before printing methods such as photolithography and blueprinting allowed for the physical separation of map layers, clever coloring and illustration techniques were employed to depict multiple layers of information. Famously, 19th century French civil engineer Charles Joseph Minard produced his *Carte figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie1812-1813*, a map of successive losses of the French Army in the Russian campaign of 1812-1813 [Tufte1983]. In it, he shrewdly manages to integrate a geographic map, the round-trip movement of soldiers through Europe, their successive troop counts along the way, and the corresponding temperature during the trek (Figure 10). Modern GIS would require multiple overlays to accomplish a similar depiction.

More recently, the earliest major developments in modern GIS occurred in the 1960s. First, Dr. Roger Tomlinson, together with the Canadian Department of Forestry and Rural Development and IBM, developed the "Canada Geographic Information System." Using a uniform coordinate system and an advanced representational architecture, CGIS was "the first GIS," used primarily "as a computerized map measuring system" [Longley2005]. Subsequently, the US Census Bureau developed a street-level map model of the United States to aid in the 1970 population census. The efforts by Tomlinson and the US Census Bureau spurred development in more generalized GIS by Dr. Howard Fisher of Harvard's Laboratory for Computer Graphics and Spatial Analysis, and later the release of commercial products from ESRI and others.

Today, GIS is a mature discipline. It is used by all modern governments to catalog myriad forms of land use, and as a strategic resource in defense. Private industry has made huge strides in mass-surveying and modeling, and is currently able to offer accurate, exhaustive street-level maps

Figure 11. A specially-equipped NAVTEQ car navigates through city streets and acquires massive amounts of raw sensor data. This data is later processed offline and refactored into a commercial GIS dataset product [Symbian2009].

for much of the developed world, and soon perhaps, the rest of the world. Companies such as TeleAtlas and NAVTEQ specialize in creating, storing and managing large-scale geospatial datasets. Furthermore, much of this is accomplished using automated feature extraction from raw sensor data (e.g. optical cameras, LIDAR, GPS, etc. See Figure 11). The rapid growth of novel content requires advanced geospatial representations to handle the constant additions of semantic, topological, and other geospatial annotations. For instance, space graphs of road networks are frequently updated with fresher data, which is more accurate, precise and complete.

However, cutting-edge GIS methods still do not address several emerging trends in geospatial modeling. First, although GIS maturely and comprehensively models the outdoors, it is sorely lacking in the indoor domain. Although many organizations manage and administer their interior spaces, primarily through in-house surveying and floor plan drafting, there are currently only a handful of standard indoor modeling specifications (see section 3.2.3.2). Typically, each organization must construct and manage its own drafting standard, and must subsequently derive proprietary analysis tools to match its representation. Furthermore, importing knowledge from external sources becomes non-trivial, because that information must be compatibly formatted and may not even be supported. This makes the process of constructing and augmenting indoor GIS

with rich data all the more onerous. Data federation and proliferation become increasingly difficult without uniform representational standards.

More importantly, conventional GIS architectures are simply ill-equipped to handle the richness of indoor environments. The topological constructs that currently represent road networks are lacking in their ability to describe higher-dimensional interior networks. The simple notions of feature-grouping (i.e. layering) that exist in current GIS are not adequate for describing multi-level buildings. Indoor environments must be studied and current models evaluated to see where geospatial representations can be improved.

Figure 12. A polygonal mesh efficiently generated from a stored geometric representation of an indoor office space.

Previous work on indoor building model representations has been done, primarily to optimize virtual, 3D building walkthroughs. Advanced geometric data models have been developed so that an interior environment can be efficiently stored and displayed at multiple **levels of** geometric **detail** (LOD) [Teller1991, Funkhouser1992]. This is essential for the purposes of visualization (Figure 12), but is lacking in its ability to characterize the aforementioned meta-geometric data, such as administrative space segmentation and adjacency, space use, and so on. More recently, work on capturing and deriving indoor geospatial models from legacy floor plans [Lewis1996, Wehowsky2001, Nichols2004], and for deriving indoor spatial connectivity, albeit manually [Look2005], has come closer to more faithfully representing indoor environments, and is the inspiration for the current thesis.

Another emerging trend in GIS is the incremental, organic contribution of information, also known as crowd-sourcing. The breadth and depth of geospatial information, its frequency of change, and the massive geographic scale in which it exists, are simply intractable features for any one centralized entity to originate. GIS, especially as it pertains to indoor environments, must be amenable to incorporating proliferated or organically-supplied data. This requires an extensible, open architecture for representing and referencing distributed geospatial data. Current examples of such architecture design, primarily used for outdoor environments, are the Wikimapia project and the integration of Wikipedia with Google Maps. The current thesis of work strives to extend the recent support for incremental, organic contribution to indoor geospatial models as well.

2 An Indoor Geospatial Model Defined

2.1 Introduction

The indoor geospatial model is a representation of indoor building environments that is designed to be descriptive and exhaustive, yet extensible to new forms of geospatial knowledge as they become available. At its core, the model characterizes four related but orthogonal elements of indoor spatial environments: their geometric, topological, hierarchical, and semantic aspects. The following sections describe each of the elements in detail.

2.2 Geometric Aspects

2.2.1 Background

GIS provides an extensive framework for describing the geometric properties of geospatial environments [Star1991]. The most typical elements of geospatial geometry are: line segments and **polylines** (serial line segments that contour an area but are not necessarily closed polygons), associated **minimum bounding rectangles** (i.e. bounding boxes) of spanned areas, coordinate system specifications, and geometric transforms (Figure 13). The key purpose of geometry in GIS is visualization (mostly critically, in concert with semantic attributes), and limited spatial analysis. Such geometric properties allow users to answer basic queries of straight-line distance, approximate area, and perimeter length. Additionally, geometry in existing GIS facilitates a simple notion of topology, which is composed of points and line segments. However, the limitations of current geometric representations become quickly apparent when trying to capture geometric features of fine-grain indoor environments.

Figure 13. Geometry in a typical GIS is composed of line segments and polygons, often grouped into layers. The key applications of geometry in GIS are the visualization and basic analysis of land use, zoning and various planning functions. Above, geometry of the MIT campus, transformed and projected into the MA state plane coordinate system, is composed mostly of line segments, and some closed polygons.

2.2.2 Rich Two-Dimensional Geometry

High-resolution GIS does not typically capture closed, polygonal area contours. Large regions, such as those defined by state and national boundaries, may be represented as closed polygons, but finer-grain regions, such as city streets, are usually captured simply as line segments, or polylines at best. This “line segment soup” lacks more meaningful definition that is useful, if not necessary when dealing with fine-grain environments. High-definition geospatial models must represent geospatial entities as discrete objects or nodes, and should therefore describe geometry in clearly articulated, concretely defined ways. For that reason, an indoor geospatial model must capture closed, polygonal area contours (or spatial footprints) that are mutually exclusive and collectively exhaustive (MECE).

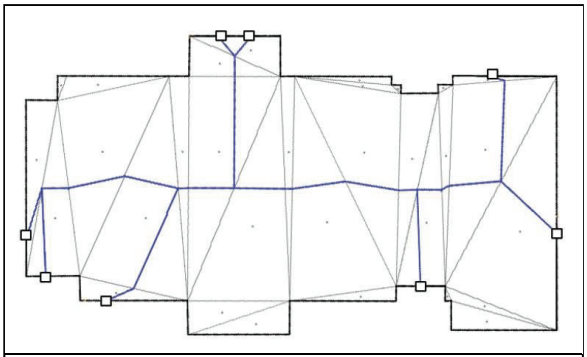

Figure 15. By using a richer geometric model, space polygons can be segmented in useful ways. Above, by triangulating the space area and generating a Voronoi diagram, an approximation of the medial axis, the space's "skeleton," can be produced.

Additionally derived geometric data, such as polygon centroids (i.e. centers of mass), shape convexity, convex hulls, area segmentations (such as constrained Delaunay triangulations and Voronoi diagrams), circumscribed circles, and medial axes (the "skeleton" of an area, or its "mid-polyline") add further utility to a geospatial model, but are not typically represented in GIS models (Figure 15 and Figure 14). Thus, as will be further elaborated on in section 2.3.2, locations and the connectivity relationships between them can no longer be described simply as points and the straight lines that connect them. The new paradigm in geospatial modeling must express locations as two- and in some cases three-dimensional objects, connected by more complex paths that traverse all three dimensions. An indoor geospatial model must therefore support the representation of more meaningful geometric information and its associated derivative geometric properties, which allow more interesting queries, such as natural-walking distance ("walking" distance), to be answered (Figure 16). Furthermore, other fine-grain geometric features, such as ingresses and egresses (i.e. doors), stairs, ramps, and elevator

Figure 14. A richer geometric representation captures closed, semantically meaningful areas. It also characterizes other properties, such as topological relationships (the purple arrows are in/egress points).

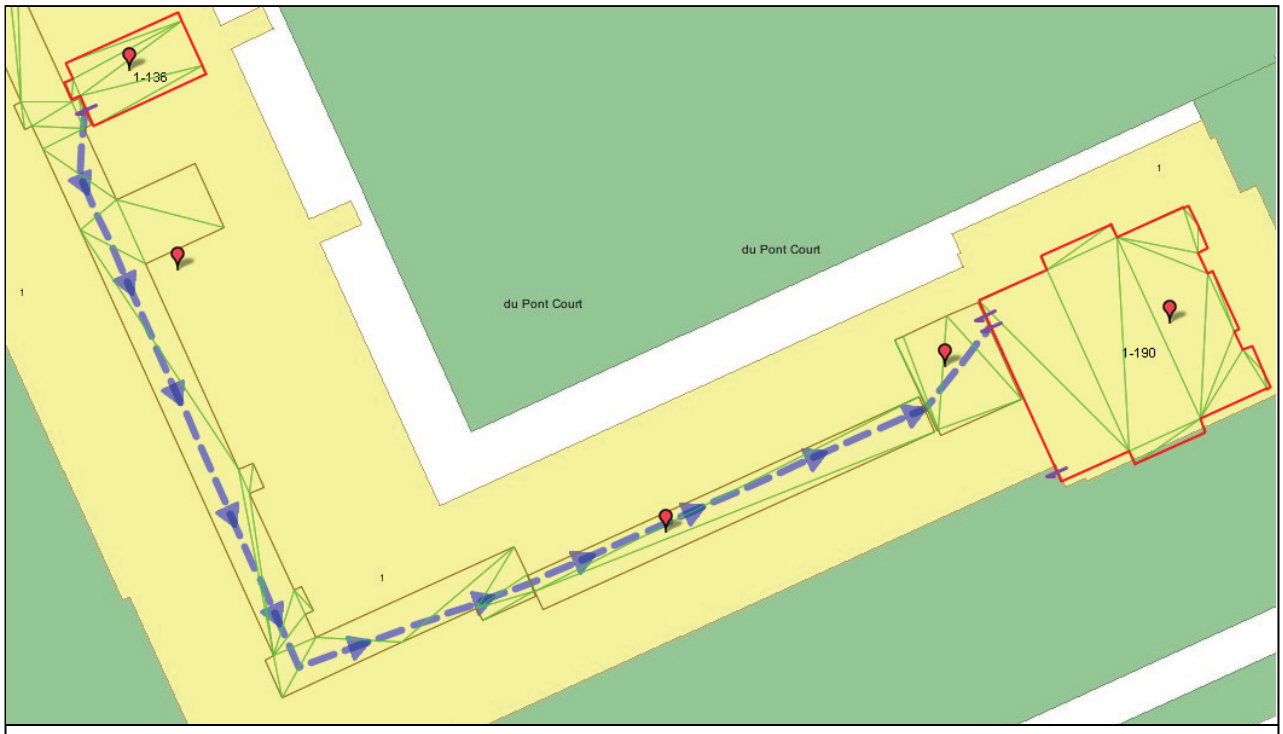

Figure 16. Extended geometric constructs, such as constrained Delaunay triangulations and the medial axis, allow more sophisticated queries, such as natural-walking distance ("walking" distance), to be answered.

spaces, and other spatial features inside or abutting a space contour, must be captured and represented as well. This can be accomplished by specifying points or lines on polygon edges (for doors, for example), and points or areas inside a polygon (e.g. for stairs or elevators).

2.2.3 The Third Dimension

In contrast to the *x* and *y*-axes, existing GIS models do not represent structure variations along the *z*-axis. This is due to the fact that their most commonly represented environment, the broader outdoor environment, does not dramatically vary vertically, nor does it typically vary in interesting or meaningful ways. Properties such as height above sea-level may shift, but the casual GIS user is not concerned with such changes that are slow-changing and not generally actionable. Other height-varying features, such as overpasses and bridges, occur infrequently enough that "flattening" them by projection onto a two-dimensional surface suffices (with the notable

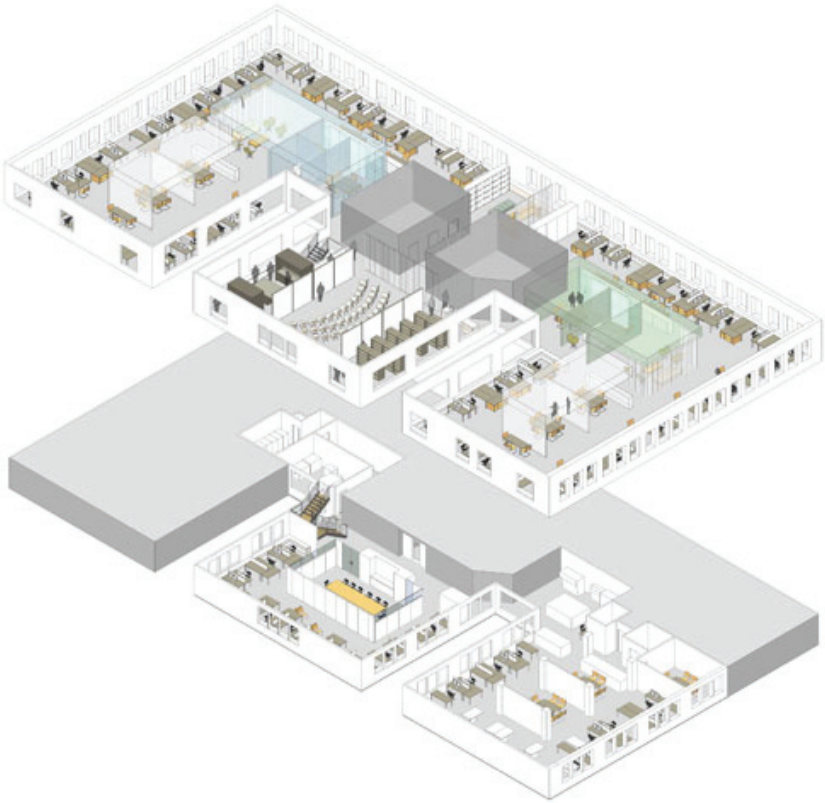

Figure 17. Indoor environments concretely introduce the notion that the *xy*-plane can be shared by multiple regions. In fact, this is the common case in multi-level building environments. As such, the *z*-component must in some way be unequivocally represented in order to properly model indoor environments. [Boora2009]

exception of GPS-based route planning, which is capable of operating in all three dimensions). The present method, however, does not suffice for indoor environments, which are rich in three dimensions. Vertical properties such as space altitudes, notional *z*-values (i.e. floor landings), heights of described areas (volumes, if applicable, e.g. indoors), and height differentials from one point inside or on a space contour to another (to describe vertical movement within an area) are essential features of indoor environments (Figure 17). For this reason, it is important to include all three dimensions in the representation, both metrically (using absolute or relative height values) and notionally (using floor levels such as "ground", "ground+1", etc).

2.2.4 Coordinates Systems and Transforms

Indoor environments present an interesting challenge when it comes to metrical description. Outdoor environments are usually amenable to representation using coarse-grain, spherical coordinate system (e.g. GPS). Indoor environments, however, demand high-precision,

rectilinear, Euclidean coordinates due to their fine-grain nature and because the need for accuracy is high. Thus, indoor geospatial entities should be geometrically expressed on a Cartesian coordinate plane using concrete metrical units such as feet or meters, rather than with non-Euclidean coordinate system (e.g. using degrees or radians). The coordinate system can be local and/or relative, with an associated transform, reference point or reference frame (even of a non-conforming type, such as GPS), but should nonetheless use a high-precision rectilinear unit (e.g. inches).

2.3 Topological Aspects

In addition to geometry, symbolic connectivity, or topology, is an essential component of any geospatial environment. This section evaluates the current state of topological representation in geographic information systems, and outlines the topological requirements of a multi-building indoor environment.

2.3.1 Background

Representing topology in spatial environments is not a novel concept [Jiang2004].

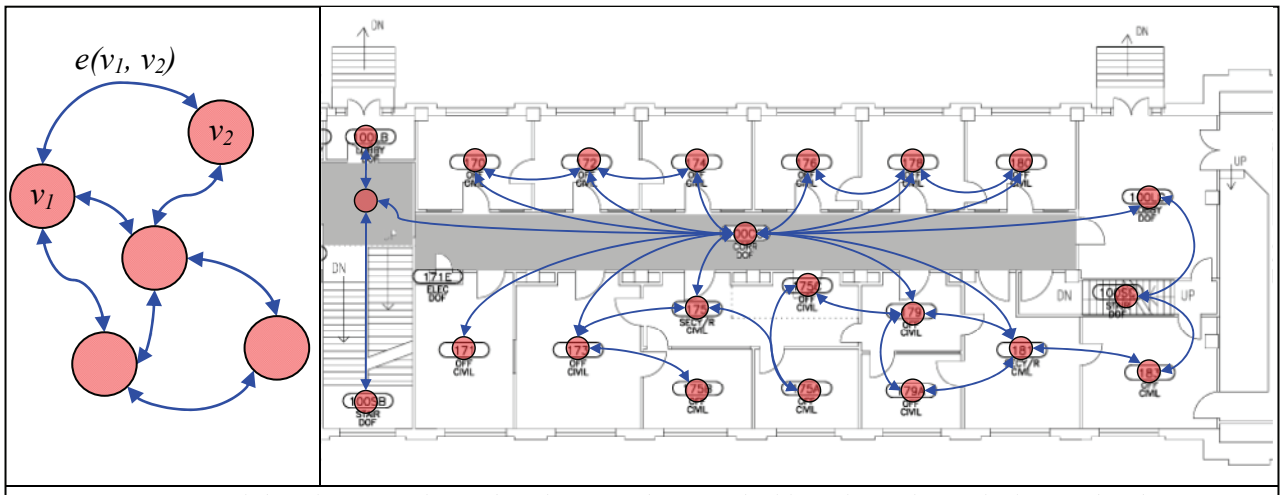

Figure 18. Spaces and the adjacency relationships between them can be likened to a directed edge graph, where spaces are nodes and adjacencies are directed edges. Above, a notional edge graph is compared to a space graph.

Topology, or topological connectivity, is the notion of describing a symbolic (meta-geometric) linkage, also called an **edge**, between two *vertices* or **nodes**. Topological representation of a spatial environment entails generating a connectivity graph where each location is a node on the graph, and each connection between two locations (i.e. a spatial adjacency) is an edge. The resulting network of linkages is what we call a **space graph** (Figure 18). Having a space graph is very useful for a variety of purposes. Most apparent is the ability to find a path of nodes that connect a start node to an end node, as one might do when wanting to produce a route between two locations (cf. Figure 16).

2.3.2 A New Topological Paradigm for Indoor Environments

Figure 19. MIT, as a landmark in current GIS, is represented by a single point, in this case its canonical street address: 77 Massachusetts Ave in Cambridge. Enhanced GIS would model landmarks as areas.

The more sophisticated, yet coarse-grain GIS models support the representation of topology, as evidenced by their ability to generate routes for popular map services both on and offline. Examples of these models are the ones supplied by GIS content providers such as TeleAtlas and NAVTEQ, among others. However, the existing coarse-grain solutions do not provide the full breadth of information that is always useful and sometimes necessary when describing geospatial environments, especially dense urban environments such as those indoors. For instance, the existing primitive topology assumes that nodes (notional locations described in the previous section) are zero-dimensional points of zero area placed on a flat, two-dimensional plane (Figure 19). The edges that connect these nodes are one-dimensional straight-line segments, also projected onto a two-dimensional surface. By linking these one-dimensional line segments, a

basic **path** of line-segments is created between two locations (cf. Figure 7). This representation, called a 1D manifold, works well for low-resolution spatial domains, where locations can be reasonably "summarized" by single points (such as street addresses), and the connectivity network is constrained to roadways that are typically straight lines, move in predictable directions, and do not scale the *z*-dimension in a meaningful way.

When representing indoor spatial domain, however, the connectivity is much richer, in terms of geometric features, dimensionality, adjacency type, and resolution. Indoor landmarks are not simply connected by long stretches of road, and meaningfully lose definition when condensed into points. Therefore, the atomic nodal unit in an indoor spatial model should be a two-dimensional "space" that encompasses an **area** rather than simply a point. This space representation, although flat (for now), exists in a three-dimensional model, where the 2D

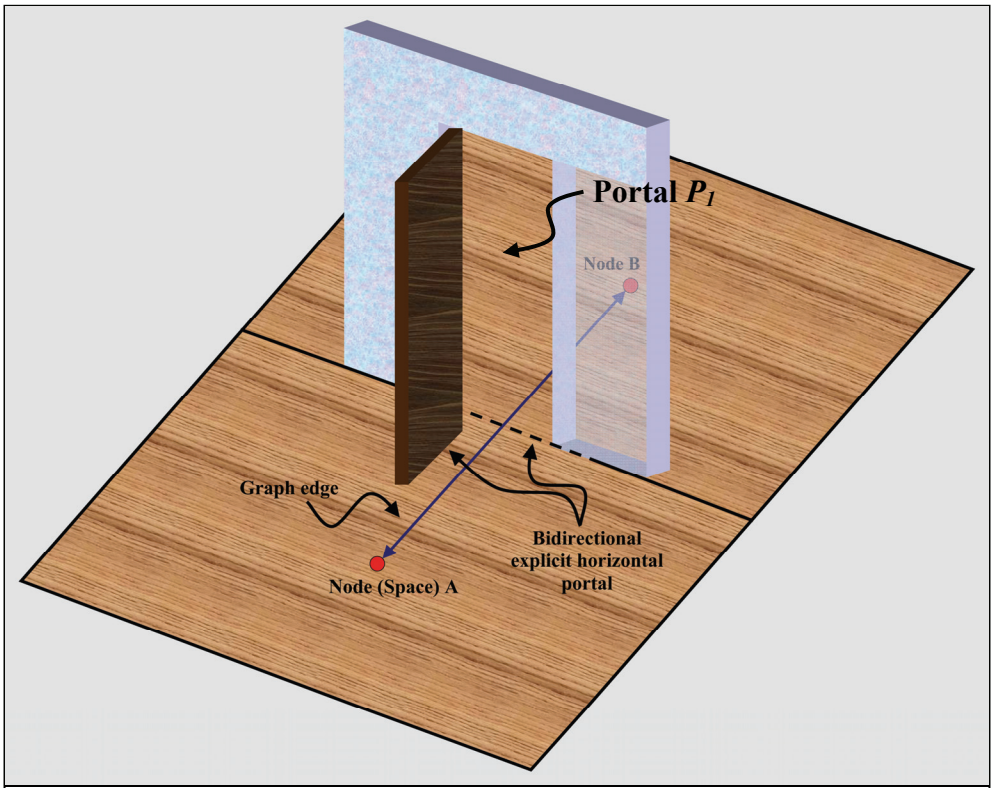

Figure 20. A portal represents both the physical and topological connectivity that exists between two spaces. Above, portal P_1, in the form of a doorway, joins two spaces.

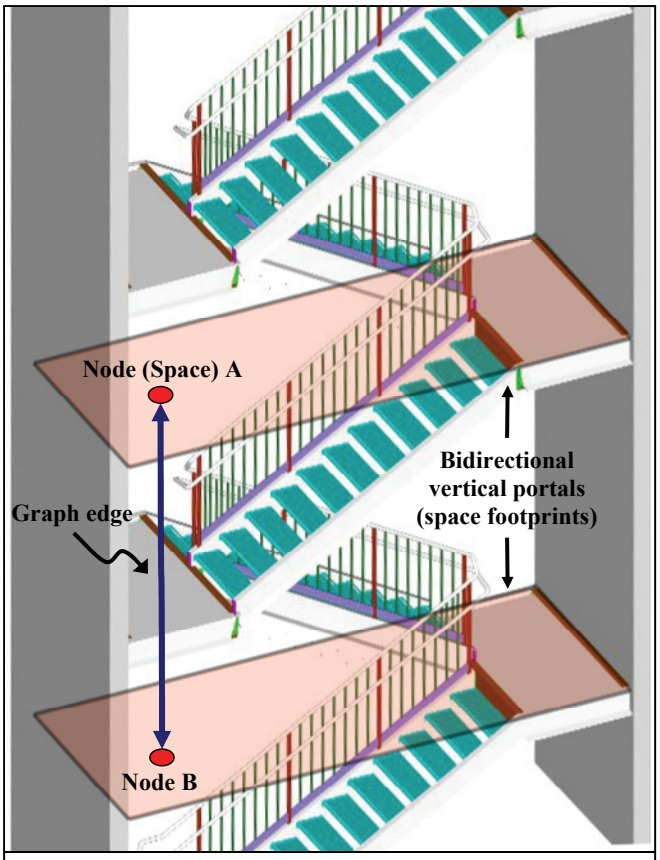

Figure 21. Above, a pair of vertical portals links stair spaces. In contrast to a horizontal portal, which is perpendicular to the *xy*-plane and abuts two space footprints (Figure 20), a vertical portal is parallel to the *xy*-plane and spans a space footprint.

geometry has a *z*-value associated with it, say, the height of the floor above local terrain or sea level, and is represented in a three-dimensional space graph. Edges between nodes represent adjacencies between spaces, namely the doors or other forms of connectivity that join them, or what we call **portals** from one space to another (Figure 20).

The concept of a portal is at the heart of our indoor topological representation. Notionally, it is a directed edge that points from one node (space) to another. Geometrically, it is a two-dimensional plane, either perpendicular to and at the edge of a space's footprint, for lateral portals, or parallel to and coincident with a space's footprint, for vertical portals (Figure 21). Although describing or extending the topological representation of a geospatial environment to geometric terms may seem like conceptual conflation, it is important for the purposes of illustrating how this symbolic representation differs from existing representations. In this model, the graph is a network of indoor spaces that are linked together into a three-dimensional arrangement, laterally and vertically. Paths are linkages of space nodes that traverse both the *xy*-plane and the *z*-dimension (Figure 22).

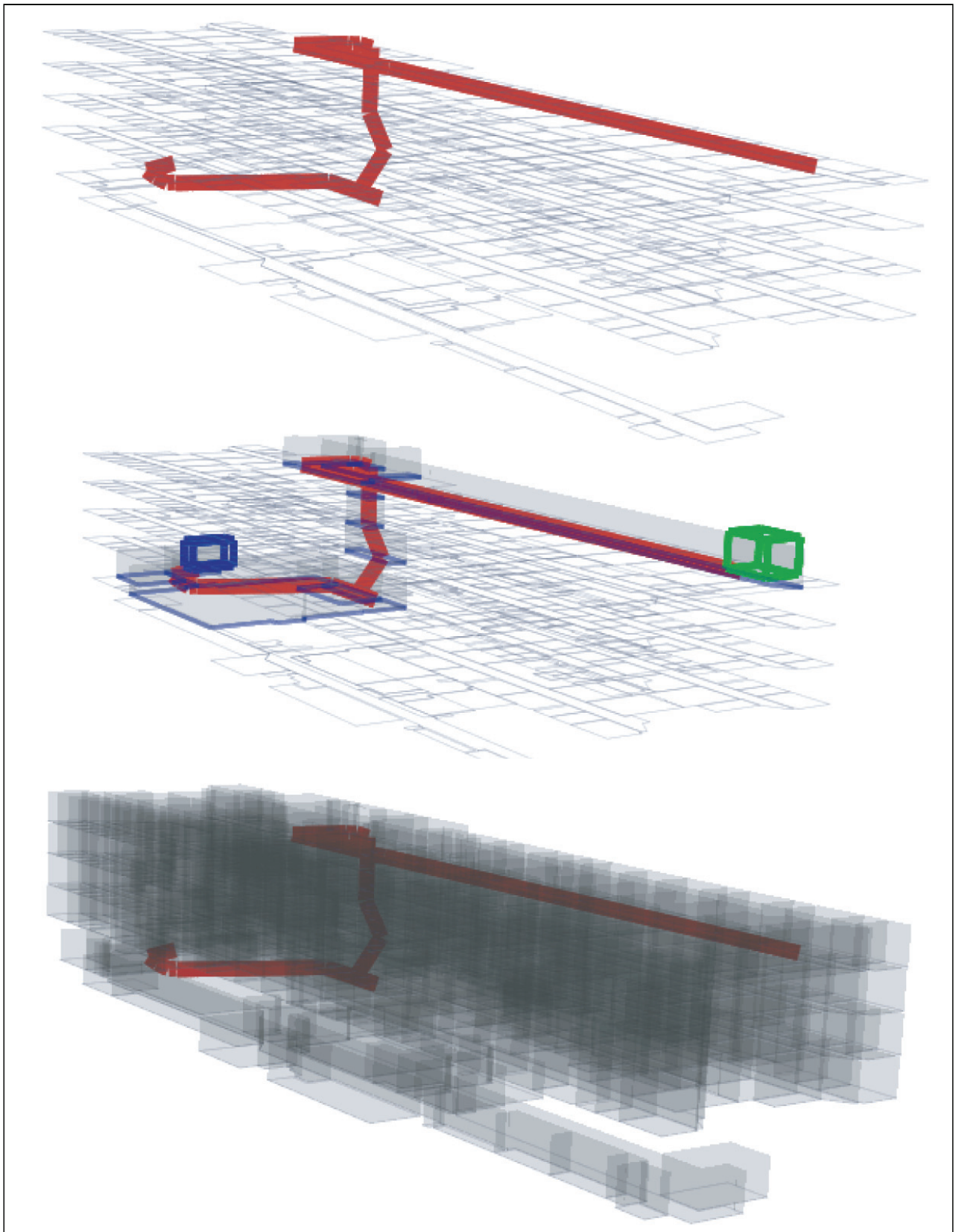

Figure 22. A depiction of the three-dimensional space graph that comprises indoor geospatial environments. In contrast to the typical 1D manifold that geometrically describes current GIS topology, this model represents nodes as two-dimensional areas, with edges that traverse not only the *xy*-plane, but the *z*-dimension, as well. [Nichols2004]

Portals have associated meta-information with them, which describe not only the two spaces they connect, but the symbolic direction in which they admit motion and additional attributes that are useful for understanding aspects of connectivity between spaces. To demonstrate this concept, one can think of **edge direction** as a piece of meta-information that further articulates an edge on the space graph, namely, the *symbolic* direction that it points: from node A to node B, from node B to node A, or bi-directionally. Other types of adjacency information that are considered useful and included with portals are: **dimensional direction**, adjacency type, and precise geometric location. Dimensional direction describes whether the portal links spaces vertically or laterally. More concretely, does the portal link spaces along the *xy*-plane (e.g. through doors) or along the *z*-axis (e.g. through stairs). We say that the dimensional direction describes the **class** of portal. Once enumerated, the portal class can be extended further with the adjacency (i.e. portal) **type**. For example, if the class is vertical, does the adjacency involve stairs, an elevator, or a ramp? If horizontal, is the adjacency explicit (e.g. a physical door), implicit (e.g. an invisible indoor space boundary), or outdoors (e.g. an invisible space boundary on the basemap)? These various semantic attributes are illustrated in Figure 20 and Figure 21.

2.3.3 Geometric Aspects of Topology

Portals also have geometric properties. Certain topologically-relevant aspects of the geometric representation are described in section 2.2.2, but endure further elaboration here. Portals exist in geometric space, and should therefore be precisely geometrically modeled. For example, if we have an explicit horizontal portal (e.g. a door), where does it lie on the space contour? To describe this, we use points or line segments specifying exactly where a portal sits geometrically. Another metrical-topological property that demonstrates utility is the concept of a

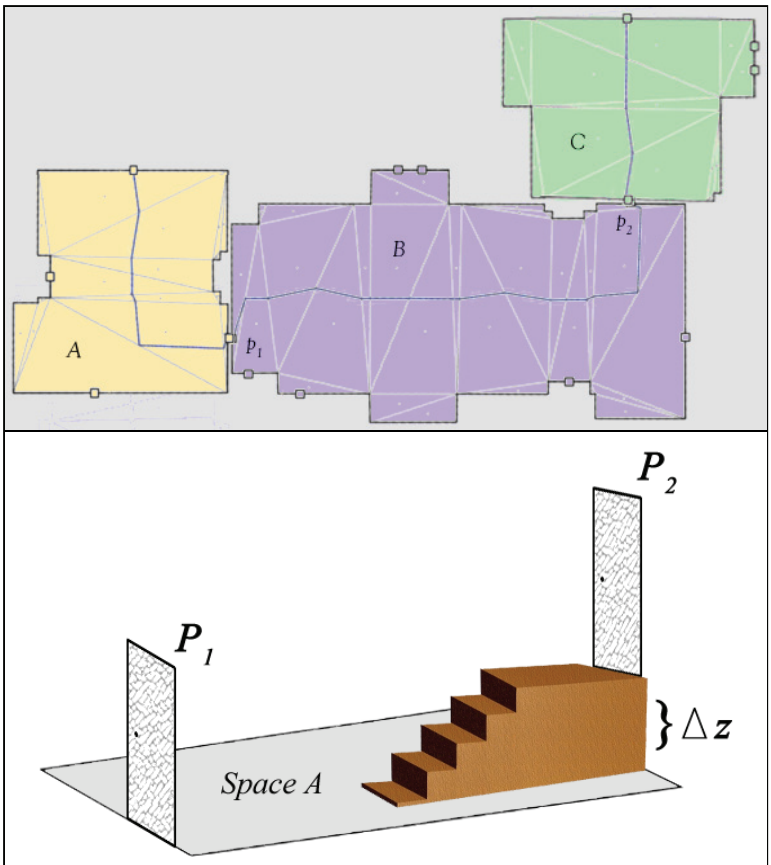

Figure 23. In the upper image, Portal pair *(p1, p2)* is contained in space *B* and links spaces *A*, *B*, and *C*. Although portal pairs are a topological construct, they can be combined with geometric information to increase their utility, such as enabling the approximation of lateral "walking" distance (represented by the darker mid-lines). In the lower image, a portal pair *(p1, p2)* captures a height-differential *z* through the intra-space path of space *A*.

portal-pair. Portal pairs are elements that comprise two **sibling portals** (i.e. two portals originating in or leading to the same space) and thus reference three spaces (Figure 23). Portal pairs are metrically-topologically useful in several ways. Firstly, they are effective in describing intra-space distances that are important in path-finding applications. Rather than basing a cost function on the straight-line distance between space centroids, a value that is often not precisely representative of the true distance between two spaces, one can instead specify inter-space distance as the distance through a portal-pair (i.e. through an intermediate space), thus providing a more precise cost metric when trying to determine the shortest path. Furthermore, portal pairs are similarly useful in encapsulating height or altitude differentials across spaces. Although counterintuitive, space altitudes can vary across laterally-connected spaces, and perhaps even within a single space. To encapsulate this spatial property, *z*-differentials are attached to portal pairs; that is, given two portals that span a space, there exists a height differential *Z* through their intra-space path.

2.4 Hierarchical Aspects

The following section discusses another integral element of geospatial models, namely the notion of containment through hierarchy. Containment relationships provide a logical mechanism for representing and storing spatial objects. Rather than maintain all spatial objects in a "flat" structure, which can quickly grow to an intractable size, hierarchy allows for the nested grouping of objects, thus reducing breadth by introducing depth.

First we evaluate existing models of spatial containment, followed by an analysis of the hierarchical demands of indoor environments. Then we propose an extended containment model to meet those demands.

2.4.1 Existing Representations

One representational abstraction that has existed in geographic modeling since the dawn of such modeling is the notion of hierarchy, or **spatial containment** [Timpf1997]. Containment is the idea that regions can be meaningfully divided into sub-regions and those regions meaningfully subdivided yet further. A simplistic illustration of containment is the partitioning of landmasses into nations, nations into states and counties, counties into cities, and so on. Although elementary in nature, GIS today fails to fully represent this notion of hierarchy and instead provides representational workarounds or limited operators to express such relationships.

One such workaround is the geometric *contains* operator, which returns all elements that lie inside a geometric envelope. This operator can provide one level of containment, but fails to support a true hierarchy of containment with multiple levels of depth, because all geometric elements within the envelope are captured agnostically, regardless of whether they are direct subdivisions or further divisions of subdivisions. Furthermore, the *contains* operator forces a

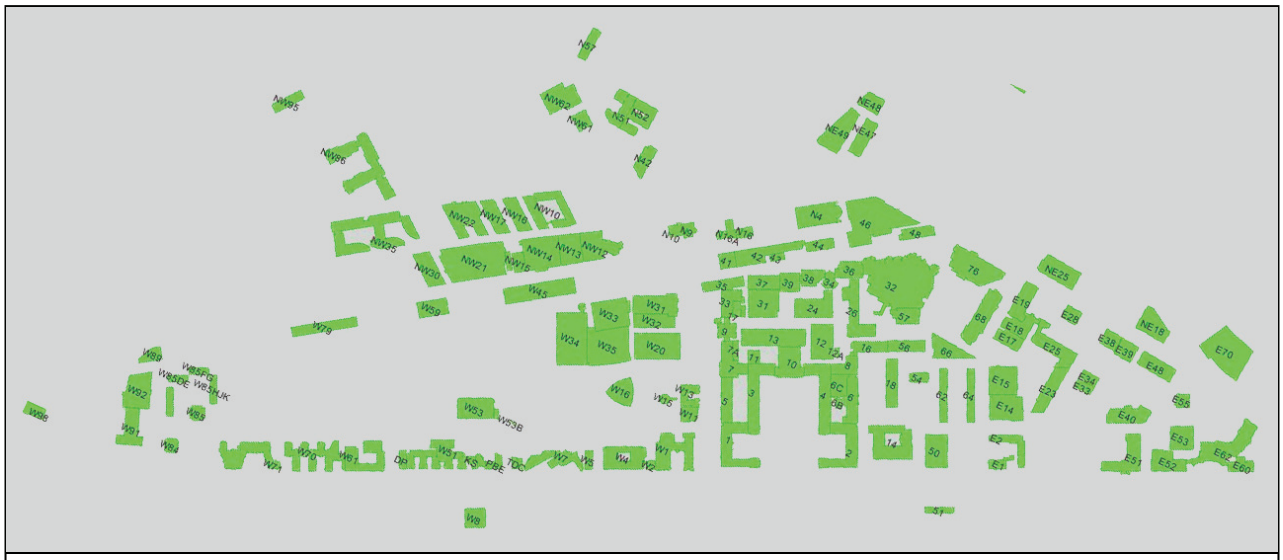

Figure 24. Standard GIS employs a layer-based hierarchy, whereby spatial objects are members of one and only one non-nestable layer. This approach works reasonably well for geometry-heavy outdoor models, but breaks down quickly when trying to represent indoor environments.

geometric dependency on the notion of containment. One can envision a situation where one region exists outside a certain geometric envelope, and yet conceptually "belongs" to the region associated with that envelope. One can think of this as a symbolic or semantic relationship, irrespective of geometry. For example, a campus might be composed of properties that are geometrically disjoint, yet are still contained within the campus region. Similarly, there may be utility in grouping or classifying spaces that share certain semantic attributes: all the bathrooms on campus might be contained in the Bathrooms container. The *contains* operator lacks these representational capabilities.

Similarly, current GIS provides another pseudo-hierarchical abstraction that has the ability to capture semantically-defined relationships. This is the notion of a "**layer**." Geometric objects can be assigned to layers, which then "contain" them (Figure 24). Once again, however, this hierarchical concept is limited in its representational power. Firstly, as is the case with the *contains* operator, layers only provide one layer of depth; they cannot be nested. Secondly, geometric

objects cannot be members of a layer by-reference, meaning that in order to have membership in multiple layers, objects must be cloned and inserted separately into each layer.

The implications of these limitations are substantial. For instance, there does not seem to be a coherent method for storing floor objects. All floors in a multiple-building environment (i.e. MIT) could be stored in the same layer, but this approach does not scale well. Alternatively, all floors of the same level (e.g. ground-level), can be stored together, which reduces the scaling problem, but is still not organizationally natural. The most appealing solution in a layer-based model dictates that all floors in a building be stored in a single layer, with one "floors" layer per building. This approach is not unreasonable, except that the dilemma is reintroduced in a more serious way when discussing rooms within floors. If the same solution were applied to rooms, an untenable number of layers would be required to coherently group them. More importantly, all the layers would be stored in a "flat" structure, which is organizationally suboptimal and again reintroduces the problem of scaling.

2.4.2 Hierarchy Revisited

For these reasons, we introduce an extended abstraction of containment: a tree hierarchy of container entities that supports geometry-independent, nested containers structured directly or by-reference. Section 2.4.7 demonstrates that this hierarchical abstraction supports not only the above-mentioned features, but is even more powerful in its capabilities by providing an advanced method for cataloging and uniquely identifying fine-grain geospatial entities, while minimizing ambiguity and maximizing organizational flexibility.

2.4.3 Hierarchy Basics

We begin by asserting that a geospatial model is in essence a collection of locations, each attributed with extensive meta-information, what might in summary be called a **rich map**. To be more precise, a geospatial model is a collection of **location entities**, where location entities can themselves be collections (i.e. containers) of other location entities. Said differently, location entities are nestable. Metaphorically, we can think of a geospatial model as a tree: some location entities are branches, which themselves can sprout other branches and leaves, while other location entities are leaves that do not sprout any further. That is, branches may "contain" other branches and leaves, while leaves act as termination points on the tree. Similarly, in a geospatial model, non-terminal or **internal** location entities are containers known as **collections**, while **terminal** location

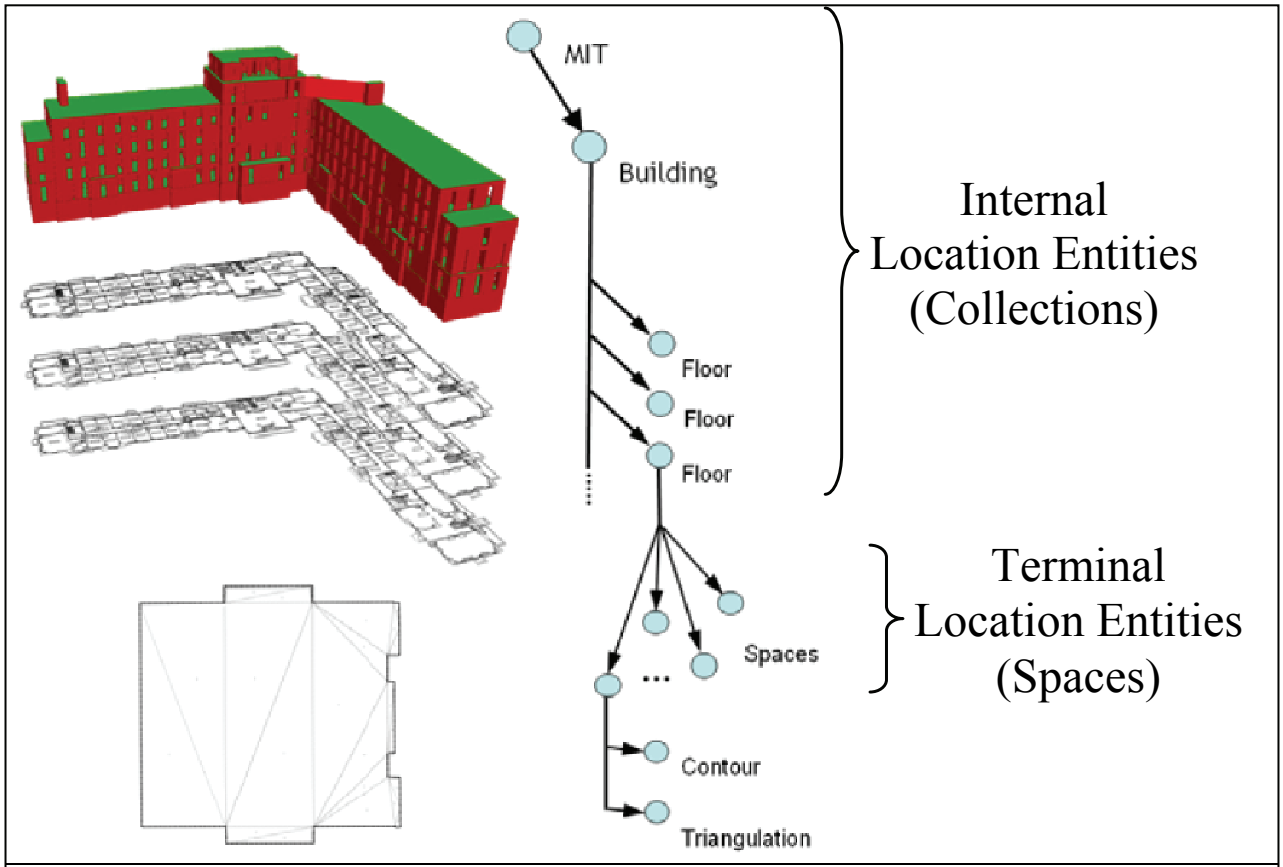

Figure 25. The location entity tree is a composition of collections, sub-collections, and spaces. The principal difference between the location entity tree and layer-based GIS is that location entities are nestable, which provides unlimited hierarchical depth. Other differences exist, as well, and are enumerated in the remaining parts of section 2.4.

entities are known as **spaces**. Collections may contain both "sub" collections and spaces. Spaces, on the other hand, are leaves or termination points for a hierarchy of collections (Figure 25).

To summarize: geospatial models are composed of location entities. Location entities can either be terminal or internal (non-terminal). Internal location entities are nested collections of "sub" location entities, both terminal and internal. Terminal location entities are spaces, the atomic (i.e. not subject to further subdivision) spatial unit, and the nodal component of the topological space graph (further described in section 2.4.4).

Internal location entities, or collections, are highly-flexible container structures. Another apt metaphor in illustrating this hierarchical relationship of location entities is by using familial relationships: internal location entities – collections – are parents, and the contained location entities, both terminal and internal, are the children. Similarly, location entities belonging to the same parent are said to be siblings to each other. This metaphor can be further extended when necessary to include all aspects of familial relationships (cousins, ancestors, and so on). Furthermore, it is useful to point out that terminal and internal location entities differ only in whether they may contain other location entities. Both types of location entities share similar geometric, topological, and semantic features.

2.4.4 The Topological Connection

At this juncture of the model, it is fruitful to link the hierarchical and topological abstractions: in reality, a terminal location entity, a space, is simply the nodal unit of the space graph described in section 2.3. Generally speaking, collections can be used to house location entities that are related in some arbitrarily-defined way, either that they are topologically-adjacent

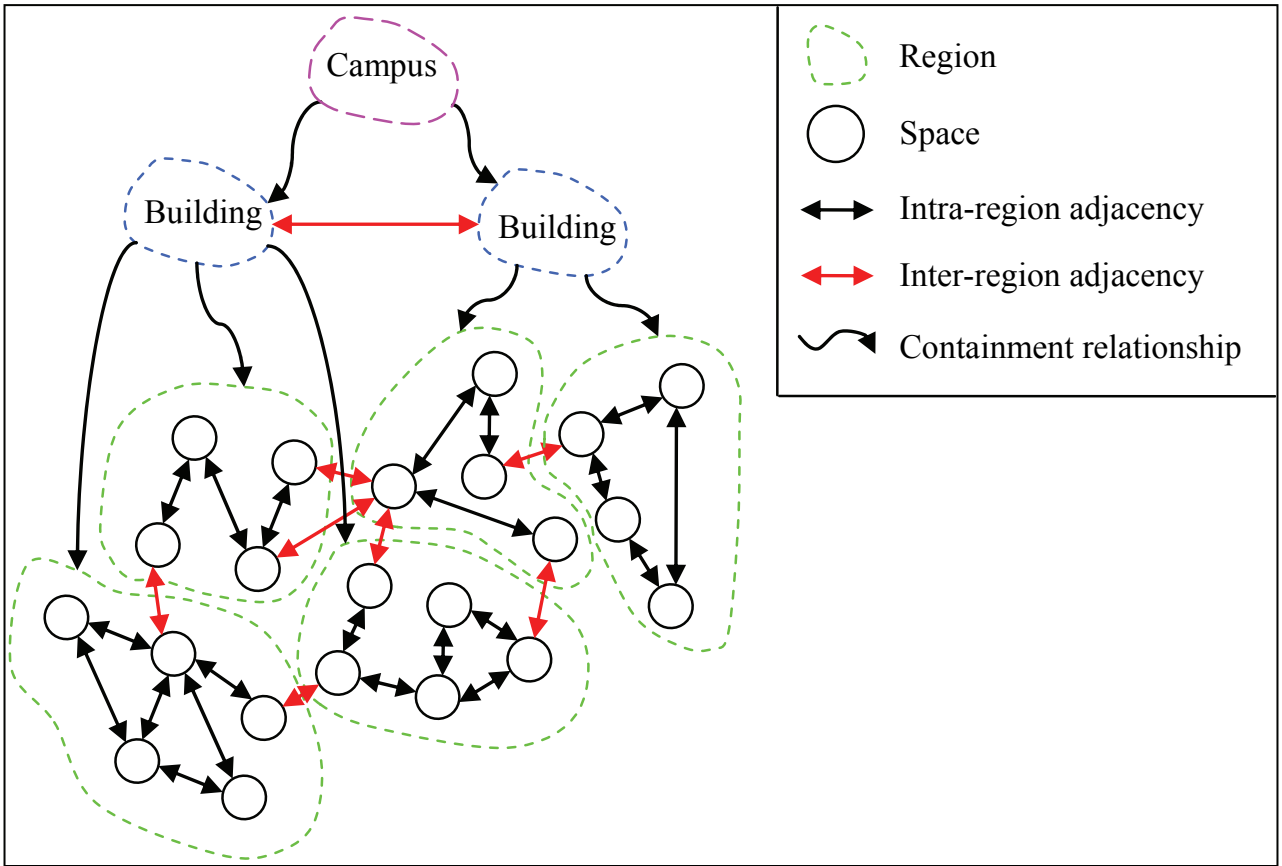

Figure 26. The hierarchical and topological constructs can be merged by noting that leaf nodes on the tree are actually topological nodes on the space graph. The tree hierarchy is superimposed "above" the network of interconnected nodes.

or otherwise semantically-related in some way. This is in contrast to the previously described *layer* notion, which has a geometric dependency. Collections that host topologically-connected location entities are called **regions**; there exists one **connected component** from the space graph within a region and each location entity within a region is topologically-adjacent to at least one other location entity within that region. For example, a building is a region that houses floor regions, and each floor is topologically-adjacent to some other floor in the building. Floor regions subsequently house spaces, and each space is topologically-adjacent to some other space on the floor. The spaces in this example, the metaphorical leaves, are the termination points on the hierarchical containment tree; they are the **atomic spatial units**. An illustration of the unified hierarchy-topology structure can be seen Figure 26.

It is important to point out that being a sibling to another location entity (meaning both are members of the same collection) does not imply that they are topologically related. Topological relation between location entities only exists when both are members of a *region*, which is defined above as a connected component. Similarly, being adjacent to another location entity does not imply a sibling relationship, as evidenced by vertically-adjacent spaces on different floors. In this respect, therefore, the space graph is orthogonal to the tree hierarchy. Said another way, we can conceptually think of the space graph as being laid out on a large flat surface, with a tree hierarchy superimposed above it. The root of the tree is at the top, and the tree descends downward, level by level (e.g. campus, building, floor, etc). The leaf nodes (i.e. spaces) are descendants of regions at the foot of the tree, but also inhabit the space graph laid out on the surface.

2.4.5 The Semantic Connection

As discussed in section 2.4.4, internal location entities, or collections, play host to sub-entities that share some arbitrarily defined relationship. When that relationship is topological, such as in the case of spaces on a floor, we call the floor collection a *region*. However, there are many reasons to group location entities into collections, and these reasons are not necessarily topological in nature. One example is the scenario described in section 2.4.1, where a campus might be composed of geometrically and/or topologically disjoint locations. In that case, the campus collection contains location entities that are only related semantically. For example, they may both be owned or managed by the same organization.

2.4.6 Representational Structure and Organization

Having an arbitrarily flexible hierarchical representation is useful, but containment relationships can benefit from some structure. For example, although it is true that all terminal

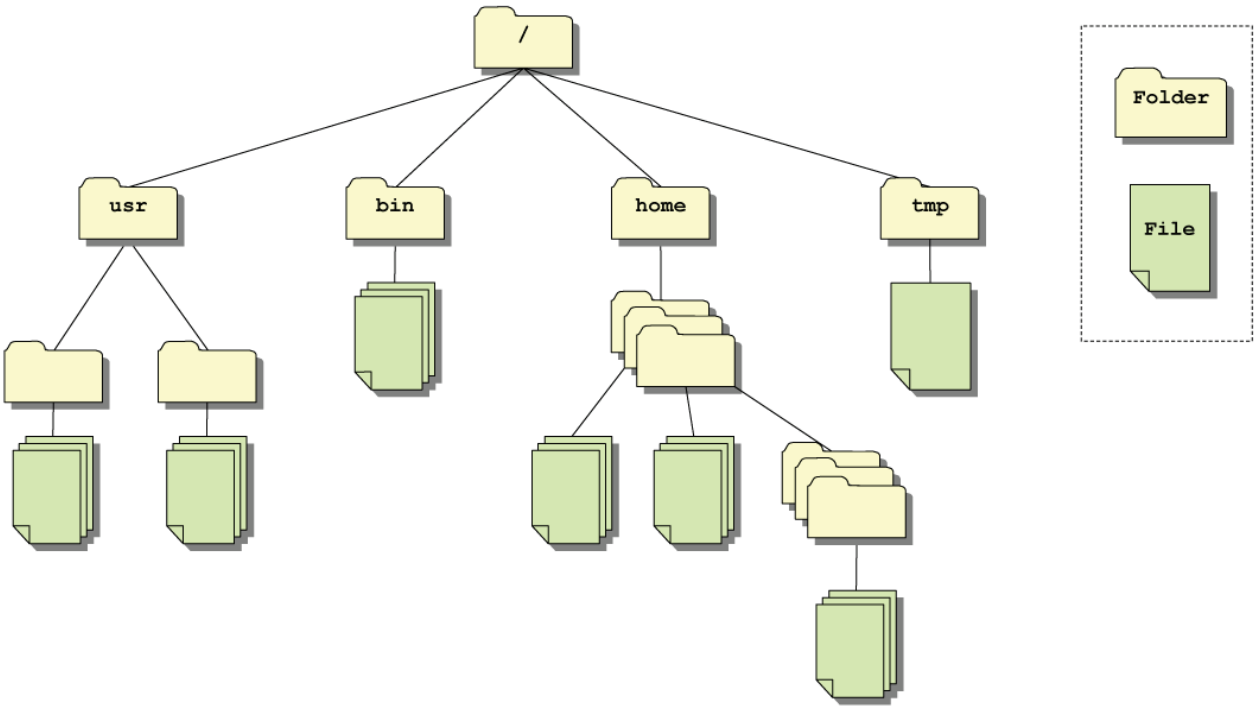

Figure 27. The UNIX file system architecture is a time-tested, flexible and efficient hierarchal representation, with containers (folders) for storing objects (files).

location entities (spaces) can exist in one unified "flat" collection (i.e. the entire space graph), and all other collections (such as building and floor regions) can point to those spaces **by-reference**, this is not the optimal representational arrangement. To draw an analogy with the UNIX operating system, imagine if all files on the volume existed under the root ("/") folder, and the directory tree was simply populated with symbolic links that pointed to individual files directly under the root. This would be suboptimal both from a representational and data storage standpoint. File system directories are a convenient organizational abstraction that make data management easier. There usually exists a logical directory to place a file. Even if a file can belong in more than one directory, there usually exists a primary directory for which its placement makes the most sense (Figure 27).

The same holds true in spatial models. An "authoritative" or primary tree of collections is useful in providing a concrete organizational structure to a geospatial environment. But what is the most logical arrangement when constructing a primary tree? Although the specification is flexible enough, and each organization is free to decide how to represent its own environment, the recommended approach to primary hierarchical organization is via a topologically-oriented tree. That is, the authoritative tree should be composed of regions, sub-regions, and spaces. Other collections may certainly exist within the geospatial model. However, their containment of spaces should be purely *by-reference*, negating the need to duplicate information and to maintain consistency. A simple example of this is a collection that represents all bathrooms within an organization. This *bathrooms* collection, which would contain "symbolic links" that reference *all* bathroom spaces within the geospatial model, might logically be placed under the root of the model. Separately, bathroom collections within specific sub-regions of the model (such as individual floors), would reference only the bathrooms that exist within those specific sub-regions.

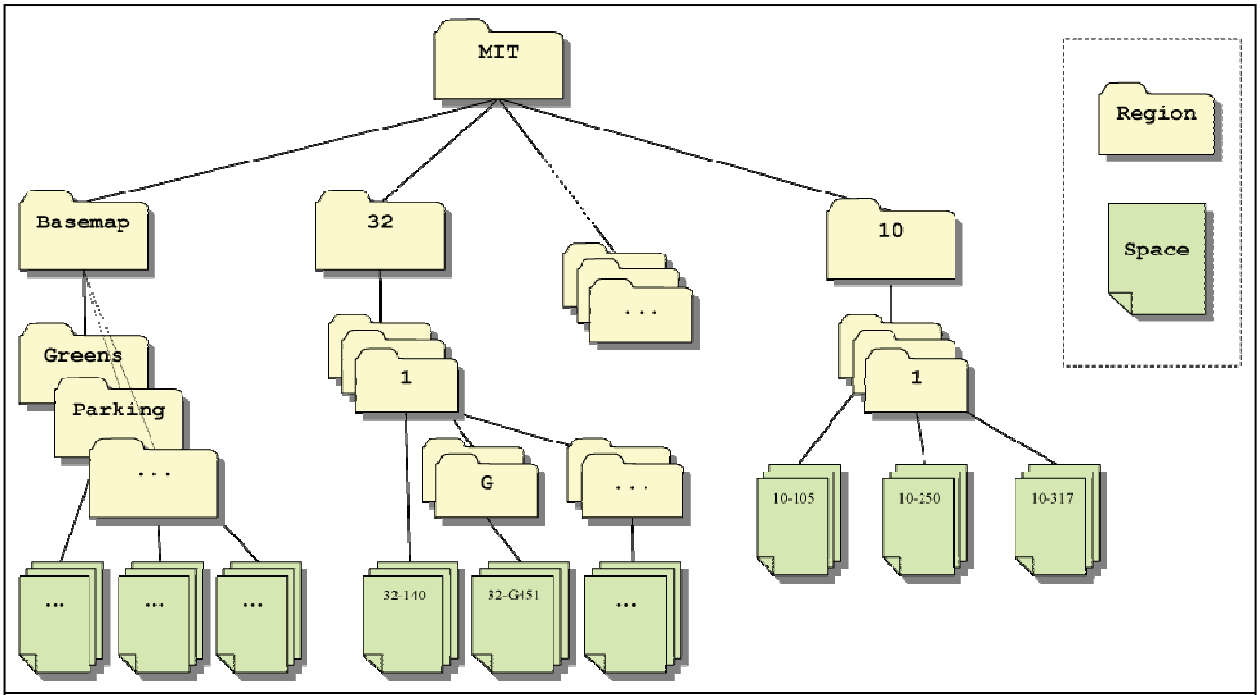

Figure 28. Like the UNIX file system architecture, an indoor geospatial model can be structured as a tree. Containers are specified based on topological association, where a "folder" in the tree represents a region (see section 2.4.4).

Ultimately, however, each bathroom space is hosted by and stored in its parent region. (For a richer discussion of how these "semantic" or *predicate-derived* collections are composed, and how symbolic, *by-reference* links work, see section 2.5.1.)

This proposed hierarchical architecture, with an "authoritative" backbone not unlike file system directory structure, is required in some form but is not strictly limited to topological trees (i.e. regions). The topological configuration, however, is most intuitive (Figure 28). It makes geospatial object modeling more tractable and data storage far more manageable. Furthermore, this representational architecture not only has representational benefits, but systems benefits as well, as will be described in section 2.4.10.

2.4.7 Representational Identification

As described in the previous section, *organizing* geospatial objects using a system similar to a file system is useful. Internal locations entities, or collections, have their parallel in file system directories. Terminal location entities, or spaces, have their parallel in file system files. Space objects can be *referenced* in much the same way as actual files, without being duplicated, using a system of symbolic links (Figure 29). The discussion until now has pertained to the *representation* of location entities, with an emphasis on coherently structuring, storing, and referencing such geospatial data objects. This section proceeds to the topic of representational *identification*. Namely, we evaluate the requirements of an identification protocol and present a solution for our indoor geospatial model.

In order for the topological space graph described in section 2.3 to function properly, each node (space) must have a **unique identifier**. As the geospatial model grows and the number of

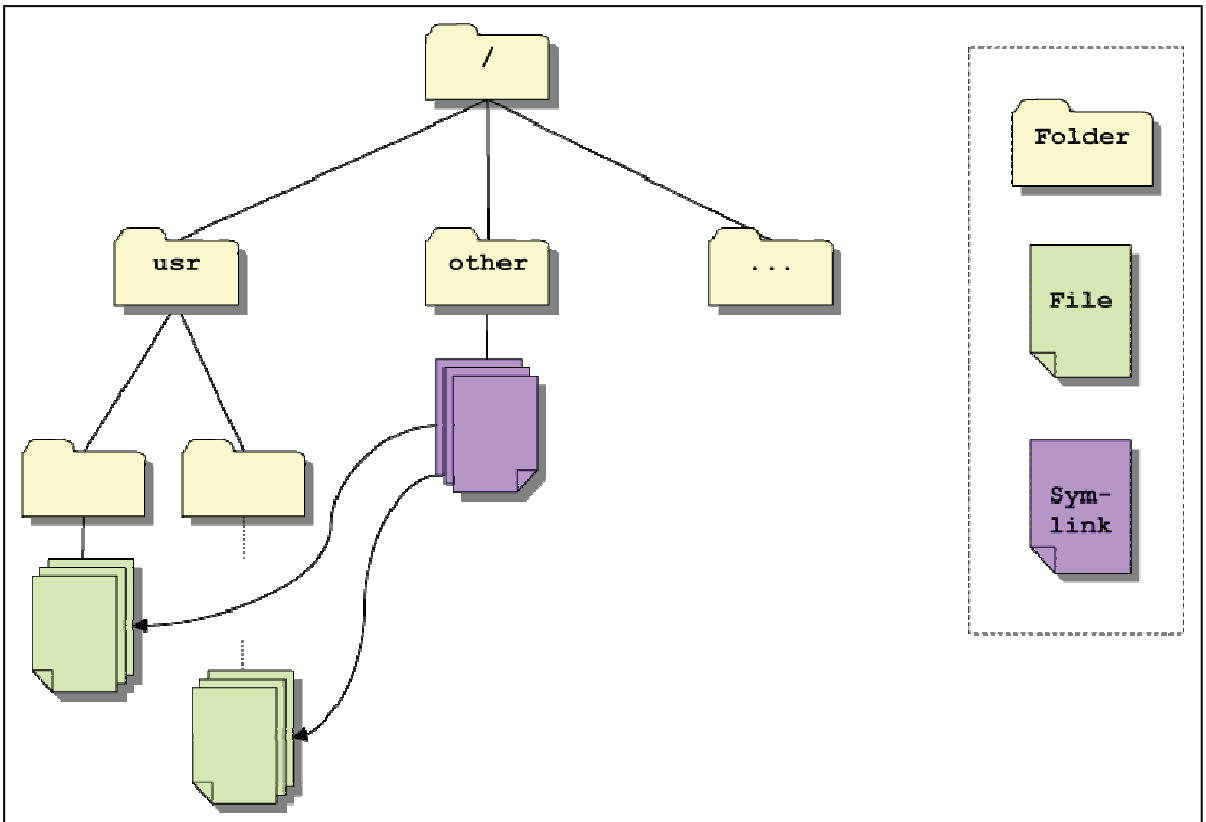

Figure 29. The notion of symbolic links offers a construct to our geospatial model that standard GIS does not. This is the ability to access geospatial objects *by-reference*. Currently, objects are stored in layers and assigned semantic meaning by them. To take advantage of multiple layers, an object must be replicated to each layer separately. Symbolically-linking an object obviates this replication step. See section 2.5 for a detailed discussion.

spaces increases, imposing a single namespace on the universe of spaces in the model becomes increasingly infeasible. At MIT, for example, nearly 40,000 distinct spaces exist. Naming each of those spaces without a coherent scoping mechanism would be highly difficult, and not user-friendly. Incidentally, the Facilities department at MIT had the foresight to use the implicit scope inherent in building identifiers to name each space. That is, each space name is composed of its building, floor, and a unique identifier within the floor. Thus, space "32-337" lies in building 32, floor 3, and room 37.

However, MIT is somewhat unique in so tersely numbering each of its buildings, and therefore composing a name from a building identifier is a relatively simple and straightforward

process. Not all organizations have such clean namespaces and location identifiers. For example, two spaces in separate buildings within a single organization might both be named "337". If both buildings are to be represented in the same geospatial model, a generalized method for disambiguating the two spaces must be devised, most logically involving building identification. Unique space identification, therefore, is a nontrivial issue that needs to be explicitly addressed when specifying an indoor geospatial model. We now describe a scalable and coherent system of *identification*, which provides a mechanism for uniquely *naming* and *addressing* location entities. (The conceptual difference between *naming* and *addressing* is discussed in section 2.4.9.)

Let us suppose that the formal name of a building is the "Ray and Maria Stata Center," and a room within the building lives on the third floor and is identified simply as "337". We are told in advance that the name of the room includes the floor identifier. We could borrow from the MIT naming syntax and state that the fully-qualified name of the space is "Ray and Maria Stata Center-337". This syntax, however, is slightly awkward, in that it creates an arbitrator separator (i.e. the hyphen) between the building and floor, but not between the floor and the room. Relatedly, it is not obvious to an outsider that the room resides on the third floor, even though that piece of information is embedded in the space name. A more informative name might be "Ray and Maria Stata Center-3-37," or perhaps "Ray and Maria Stata Center-3-337" (as there may be organizational value in maintaining the given floor identifier).

Alternatively, we can extend this idea one step further by once again borrowing from the UNIX file system. A fully-qualified name patterned after a file system path might look as follows: "/MIT/Ray and Maria Stata Center/3/337", which is actually the concatenation of a *path to* and the *name of* a terminal location entity. Even for an organization like MIT, which already uses a

workable identification system, the proposed method provides additional benefits. Our previous example of "32-337" becomes "/MIT/32/3/32-337" (once again, maintaining the actual room name for completeness) and has the following additional benefits: 1) it natively follows the directory-like organization that the model already uses; now, both the *structure* and *naming* are based on the file system metaphor; and 2) it clearly articulates the separation of descendent location entities, both internal and terminal, by using "/" tokens. This makes it easy to trace a path down the tree hierarchy, directly and uniquely to a location entity.

Before we proceed to describe how the naming mechanism is useful in the topological domain, we first present several terms to crystallize the previous discussion on naming. As demonstrated earlier, namespaces and scoping are critical in large-scale datasets that require unique names. For that reason, a new **namespace** is defined in each collection (i.e. internal location entity) that descends from the root collection. For example, floor 3 in building 32 has its own namespace. It may duplicate room names from other floors or buildings, but all names on its floor must be unique. These unique names are said to be the **canonical names** of location entities. All distinct objects within a collection, both terminal and internal, must have a canonical name that uniquely identifies them in that collection.

As an aside, the reader may recall from section 2.4.6 that location entities can be symbolically referenced. In order to comply with namespace constraints, symbolic links must also be uniquely named within a collection, even though they are not canonical in nature. Furthermore, the reader may wonder whether the notion of object aliases, such as **colloquial names**, exists. Aliases, synonyms, and symbolic links are discussed in detail in section 2.5.

Returning to our discussion, the **fully-qualified canonical name**, as described above, is actually a concatenation of canonical names separated by slashes. The **canonical path** portion is composed of the canonical names of the ancestral collections (e.g. regions) in their respective namespaces, and the final string portion is the canonical name of the space in its namespace. Together, they form the *fully-qualified canonical name* of a location entity. The reader may recall from section 2.4.6 that we described an "authoritative" tree representation, which acts as the hierarchical backbone of the geospatial model. This primary tree is by definition the canonical tree representation, composed entirely of canonical paths. And as suggested earlier, this primary hierarchical structure should follow from the *topological* structure of the geospatial model.

Said another way, canonical paths should be composed of regions. For example, a canonical path might be "/MIT/32/3/32-337", whereas "/MIT/Stata/3/32-337" would not be. In this case, "Stata" is merely a symbolic reference to "32," defined as a colloquial name for convenience. Paths that include any non-canonical components (for example, symbolically-linked regions) are not canonical.

2.4.8 The Topological Connection Revisited

As the reader may recall, one motivating reason for the naming design described above is unique node identification for spaces in the topological space graph. From section 2.3.2, we recall that adjacent spaces are linked together via portals. A portal is essentially a directed edge (or vector) that points from one space to another. Concretely, this linkage is easily facilitated using the hierarchical architecture proposed above, and the associated representational naming mechanism. A portal can be described as a data structure that, at its most basic, simply enumerates an ordered pair of fully-qualified location entity names; two spaces are linked together via their paths on the

tree hierarchy. For instance, *e("/MIT/32/3/32-337", "/MIT/32/3/32-334")* represents a single directed portal between those two spaces. Reversing the two paths would reverse the direction of the portal.

Location entity references can even be made via relative or symbolically-linked paths. Because location entity identifiers follow the file system metaphor, object references can be made via either absolute or relative path naming, or any other resolvable reference (i.e. symbolic link). If two location entities exist within the same collection, they can reference each other directly by their canonical names, without needing to specify full hierarchical paths. For instance, in the example above, both spaces belong to the same collection. Therefore, the portal can be reduced to: *e("32-337", "32-334")*, which utilizes the relative position of the spaces to each other on the tree. This further streamlines the topological operation of graph "walking," or traversing the network of nodes with the goal of reaching a specific *destination* node, and goes hand in hand with notion of **out-of-core** processing described in section 2.4.10.

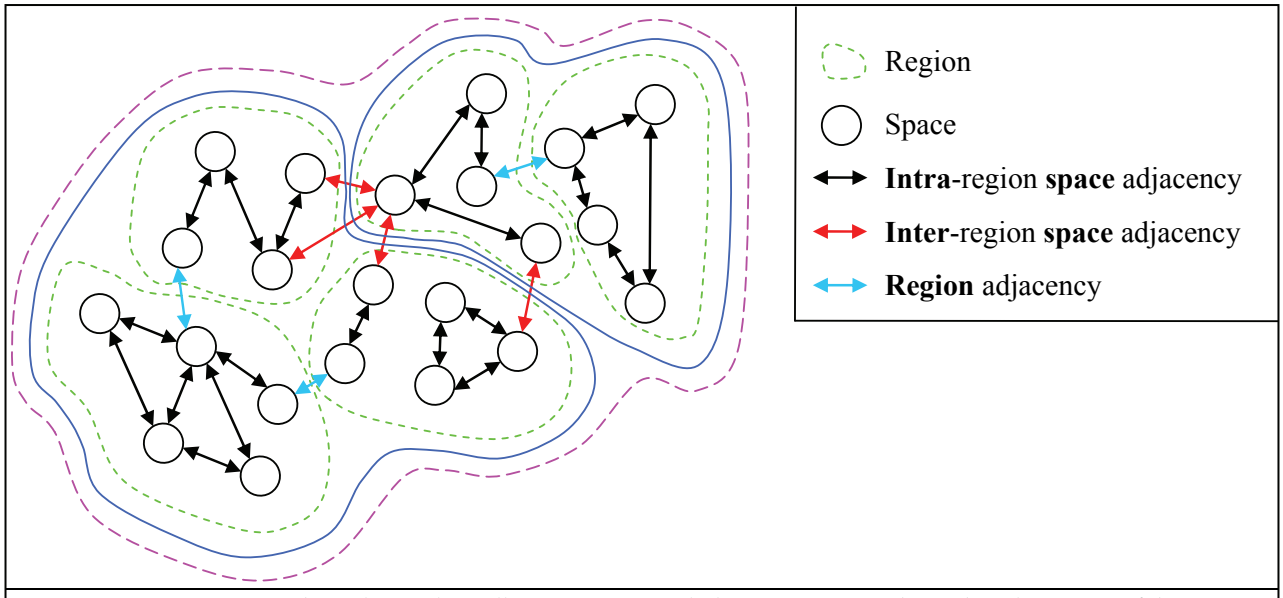

Figure 30. Spaces are not the only topologically active geospatial objects. Regions inherit the adjacencies of their descendent spaces and become supernodes that are topologically connected to adjacent regions. The adjacent regions are themselves supernodes by virtue of their descendant spaces.

A further point to note about tree hierarchy and topology is that the network of nodes is not strictly limited to spaces. As described earlier, terminal location entities (i.e. spaces), are the atomic spatial unit and represent the formal nodes in the space graph. However, the reader may recall that the authoritative tree hierarchy for a geospatial model is the topologically-oriented one. Sibling collections in the primary tree (i.e. regions) are, by definition, topologically connected, and thus a graph network of regions exists, as well (Figure 30). The region as a node can be thought of as a "supernode", which is the convex hull of all the nodes that are contained within it. This supernode inherits all adjacencies that cross its hull, to and from other nodes and supernodes in the graph. For example, a floor region is the supernode of all spaces that reside in it. This floor region inherits all adjacencies that intersect its boundary. Those adjacencies might be to other floors (e.g. vertically), other buildings (e.g. laterally), or outside (e.g. on a ground floor).

This concept of a supernode is useful for a variety of reasons. Firstly, an application may be interested in determining only a "low-resolution" path between two points. For example, it may want only the buildings that connect two locations. Similarly, this feature can be used as an optimization in graph-walking, by quickly determining a skeletal, low-grain path between two destinations [Whiting2007]. This process heavily prunes the graph to a small subset of possible high-grain paths. Only then do we refine the path by searching a finer-grain version of the redacted graph, which by now contains only the pertinent elements of the network.

2.4.9 Hierarchy in Distributed Access

A feature that is further elaborated in chapter 3, but is pertinent in the current discussion is the amenability of the tree hierarchy to be represented using the internet addressing system. As geospatial data distribution increasingly occurs over internet networks, it is important that the

geospatial representation be easily extendable to internet environments. Conveniently, the file system metaphor of hierarchical representation readily lends itself to being translated into a system of internet URLs (as well as, of course, to standard file system paths).

More precisely, the tree hierarchy and file system metaphor can be mapped exactly to the internet naming and addressing architecture of **U**niversal **R**esource **N**ames (URNs) and **U**niversal **R**esource **L**ocators (URLs), collectively known as **U**niversal **R**esource **I**dentifiers (URIs) [Mealling2002]. In the geospatial case, the URL would be any hierarchical path that leads to a location entity. This includes actual (canonical) paths and symbolic (non-canonical) paths. For example, the bathroom space colloquially referred to as "32-317" would be referenced not only

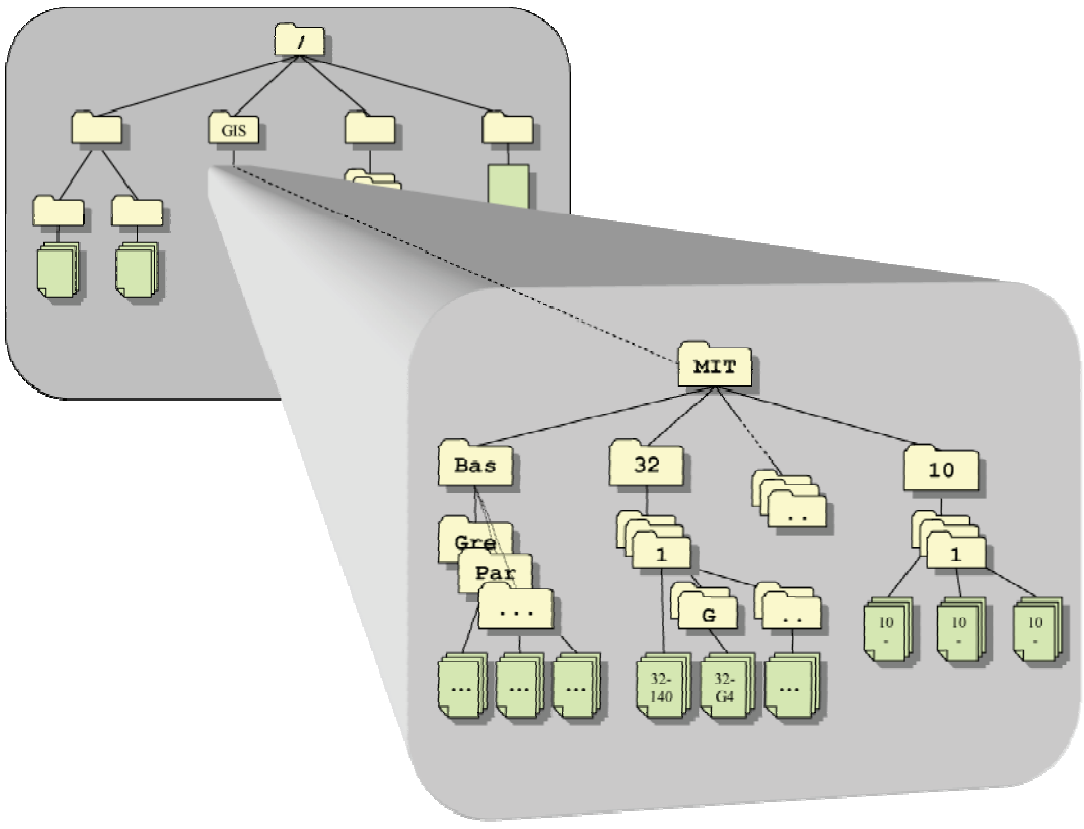

Figure 31. Because our geospatial model is predicated on the UNIX file system metaphor, providing access to geospatial data is simple. In fact, the model can be mapped directly into the file system and geospatial objects can be accessed with minimal interface overhead. Above, data for our geospatial model can be accessed using the "`/GIS`" path. E.g. "`/GIS/MIT/32/3/32-337`".

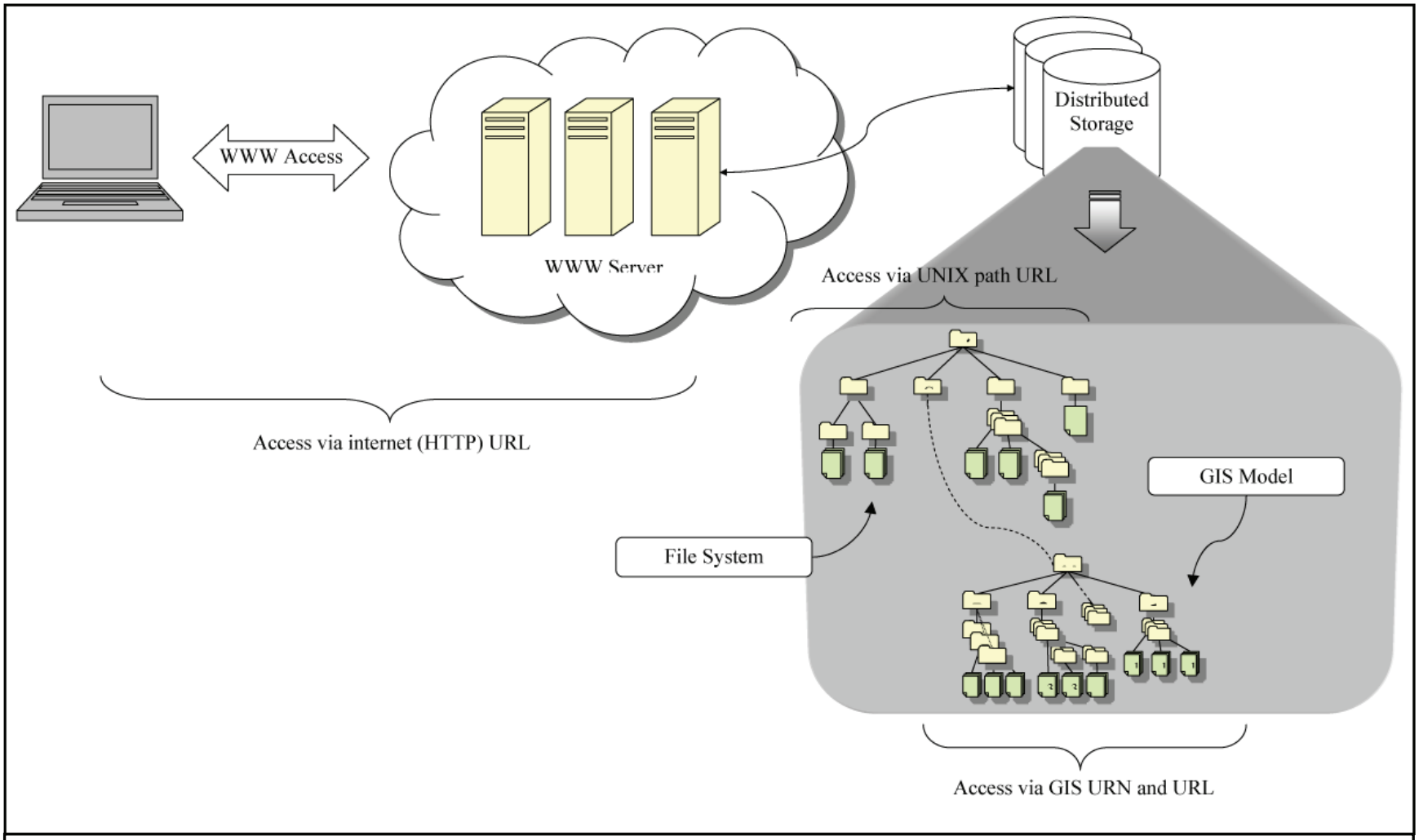

Figure 32. The transfer of distributed geospatial data is greatly optimized by its hierarchical representation. In particular, the hierarchical structure and naming of our indoor geospatial model exactly mirror that of the UNIX file system, and by extension, the internet addressing and data distribution schemes. Above, the path of a query is illustrated as it originates from a user, is accepted by a server, and is applied to a GIS dataset stored on distributed media.

directly, via the fully-qualified canonical name "/MIT/32/3/32-317", but perhaps also symbolically by "/MIT/Bathrooms/32-317" and "/MIT/Stata/Bathrooms/32-317." Each of these is a URL, which "locates" or addresses a geospatial data object. Additional pieces of addressing information, such as an HTTP server or data storage device, can be further prepended to a geospatial URL to allow distributed access to such data (Figure 31 and Figure 32). For example, a complete URL might look as follows: "http://maps.csail.mit.edu/models/MIT/32/3/32-317". A URN, on the other hand, is a single distinctive *identifier* for an object, which supplies all pertinent namespaces required for total disambiguation, but does not necessarily address an object, nor does it guarantee access to an object even it does coincide with an address. A separate URN-to-URL resolver translates a unique URN into a potentially unlimited number of URLs, all referring to the same object – in our case a geospatial object. For the URN value, we recall that the fully-qualified canonical name, the chain of canonical names that compose the path and space name, can serve exactly this purpose. Thus, the URN in the above example is "/MIT/32/3/32-317", absent any further storage- or access-specific elements, such as mount points or web servers. These details become relevant when discussing distributed geospatial data access.

2.4.10 Storage and External Memory Access Optimization Through Hierarchy

Concretely, from a systems perspective, data storage (via representation) and access (via naming) is far more tractable using a tree-descent architecture. Large-scale data management and access often benefit from mechanisms that logically partition a data space, either by frequency of use or by meta-relationship. Examples of these mechanisms are *memo-izing*, *swizzling, prefetching,* and *out-of-core* operation [Varadhan2002]. These mechanisms aim to either remember (e.g. cache) pieces of information with a policy dictating how much to remember and when information expires, or

to predicatively and intelligently load only part of a dataset from **external memory** into resident memory, for reading and manipulation.

Incidentally, the above optimization tactics are conceptually similar, and all benefit from hierarchical partitioning; a tree representation, especially a topologically-oriented one, facilitates these optimizations by segmenting data into logical and manageable pieces (Figure 33). Collections are a form of memo-ization and facilitate swizzling [White1994], pre-fetching, and out-of-core operation by making it easy to fetch, store, and otherwise handle a relevant part of the geospatial model. For example, if a user is interested in exploring a certain room, it makes sense to recall,

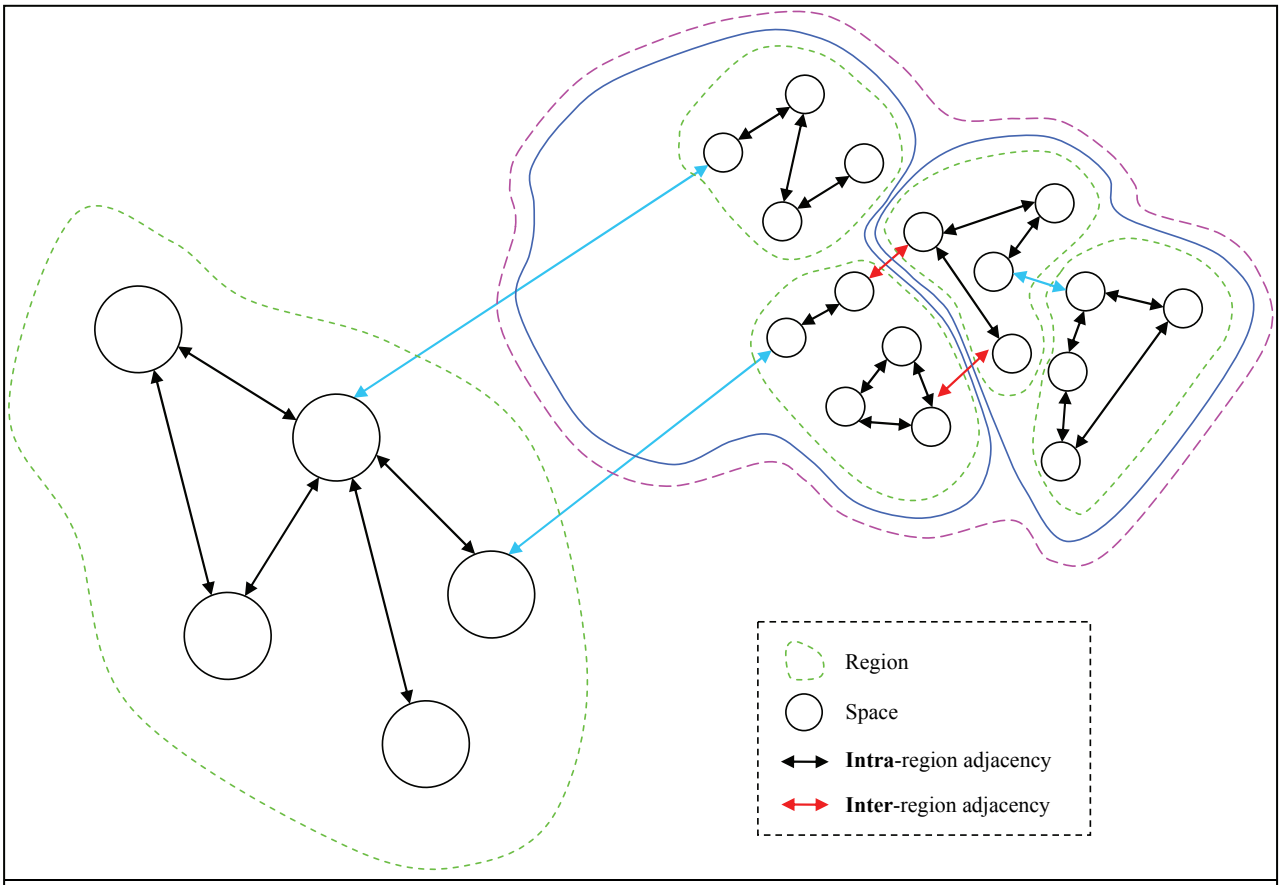

Figure 33. For large geospatial datasets, the ability to load only part of the model into memory is a critical aspect of GIS. The sheer amount of information makes handling an entire model at once infeasible, if not impossible. Furthermore, the user is generally interested only in a subset of the model at any given time. Thus, providing a means to logically partition the data into blocks is useful and the hierarchical organization of our geospatial model makes this convenient. As illustrated above, a region can be imported from external memory and operated on independently.

perhaps even in advance, a portion of the floor near that room. Conversely, we would not necessarily want to load the entire building into memory, which is unnecessarily resource-intensive. The fact that each floor is segmented independently makes this access method possible.

Another example demonstrating the data-management efficiency of the tree-based representation is to imagine that an organization (such as MIT) compiles (either formally or organically) a collection of all the *landmarks* on campus. Rather than search through the entire model each time, looking for locations that meet certain criteria, landmarks can be remembered and assigned to collections via symbolic links that point to the fully-qualified names described above (the actual location entities, as they are stored in their parent region). In other words, the tree hierarchy also facilitates the memo-ization of predicate-based collections. In this way, all landmarks on campus can easily be explored "out-of-core", that is, without needing to deal with the entire geospatial model each time. The competing "flat" representation, as described earlier, becomes increasingly infeasible for the operations described here, as the model-size and dataset-scale grow.

2.5 Semantic Aspects

The final component of a geospatial model embodies the semantic aspects of spatial objects. Semantic attributes define the purpose, meaning or identity of an entity, and are thus of critical importance to any spatial representation.

The following sections describe in detail various semantic notions, and propose a flexible and robust representational architecture that captures such knowledge.

2.5.1 Introduction

An essential component of geospatial data is the multi-layered and multi-faceted semantic knowledge that is often "attached" to it, the rich "meta data" that gives location entities their meaning. In the preceding sections, extensive discussion focused on several major properties of geospatial data, namely their topological, hierarchical, and geometric structures. Semantic data is essentially "everything else". How do people colloquially refer to space X? What type of space is X? What events have occurred or will occur in X? Who owns, administers, and/or uses X? These questions transcend the more traditional attributes of physical space (absolute location, geometry, adjacency, and so on) that conventional GIS systems typically encapsulate, and instead seek to capture the way people relate to spaces and the meanings they assign them.

The discussion of geospatial hierarchy in the previous section introduced some notions of geospatial semantics. For example, collections are one way of capturing semantic properties of location entities or creating semantic relationships between location entities. Predicate-based collections, which symbolically reference all geospatial objects with semantic property X, are effectively semantic containers. The hypothetical *bathrooms* collection was a heavily cited example. More concretely, a predicate-based collection is a collection composed entirely of child entities that are the result of a search query. The query works as follows: a search predicate such as TYPE="bathroom" is presented as input, together with a source path to search. The system searches all location entities under the source path, looking for entities that match the predicate, and creates a new collection that is composed of symbolic links that reference the matching location entities. In this case, the collection would be composed of all location entities that have a

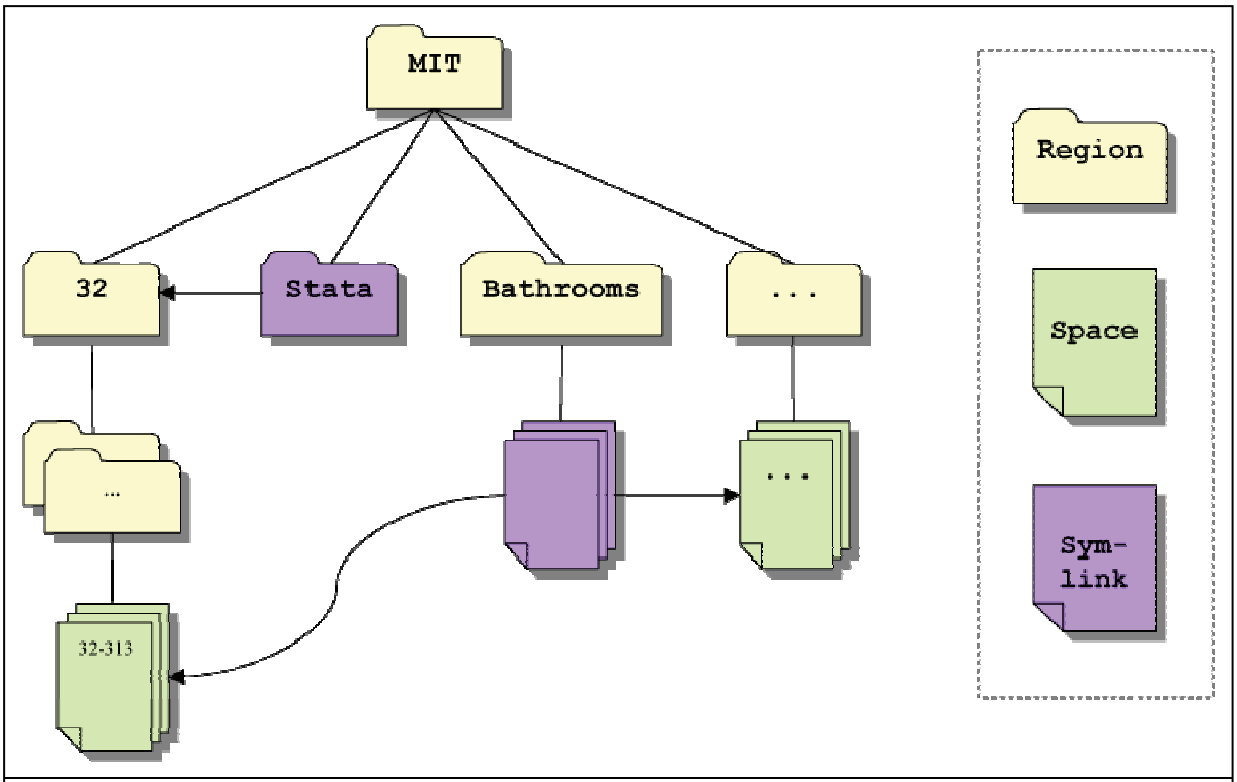

Figure 34. Geospatial objects can be symbolically-linked in much the same way as files and folders. This linking is a flexible (but not exhaustive) way of capturing semantic properties. For example, colloquial names (such as Stata) can be represented as symbolic-links and spaces of a certain type (such as bathrooms) can be grouped in this way.

type equal to *bathroom*. This is similar to the concept of a virtual or "smart" mail folder, which contains all mail that matches a certain search predicate.

Furthermore, symbolic links are themselves a kind of semantic attribute (Figure 34). The symbolic path "/MIT/Stata" is a pointer to the canonical path "/MIT/32". "Stata" is a common nickname for the "Stata Center" or the "Ray and Maria Stata Center," or MIT Building 32. The MIT Facilities department canonically refers to the Stata Center simply as "32", as do we in our exemplar model. Semantically speaking, however, Stata and its companion names are used to colloquial identify "32", as does "Dr. Seuss Building" and other affectionately-coined nicknames. Each of these handles is a piece of semantic information that is attached to the geospatial object contained at the canonical path "/MIT/32," and must be captured in the geospatial model.

2.5.2 Encoding Semantic Information in the Model

As described above, symbolic links are one way of expressing semantic information associated with a location entity, by creating a pointer to a location entity and meaningfully naming (e.g. “Stata”) or grouping (e.g. “Bathrooms”) said pointer. However, the model supports another mechanism for encoding semantic information. Some types of meta-information are arbitrary in nature and do not lend themselves well to the structures of simply naming, grouping, or typing; other pieces of information are highly subjective. Thus, representing them through

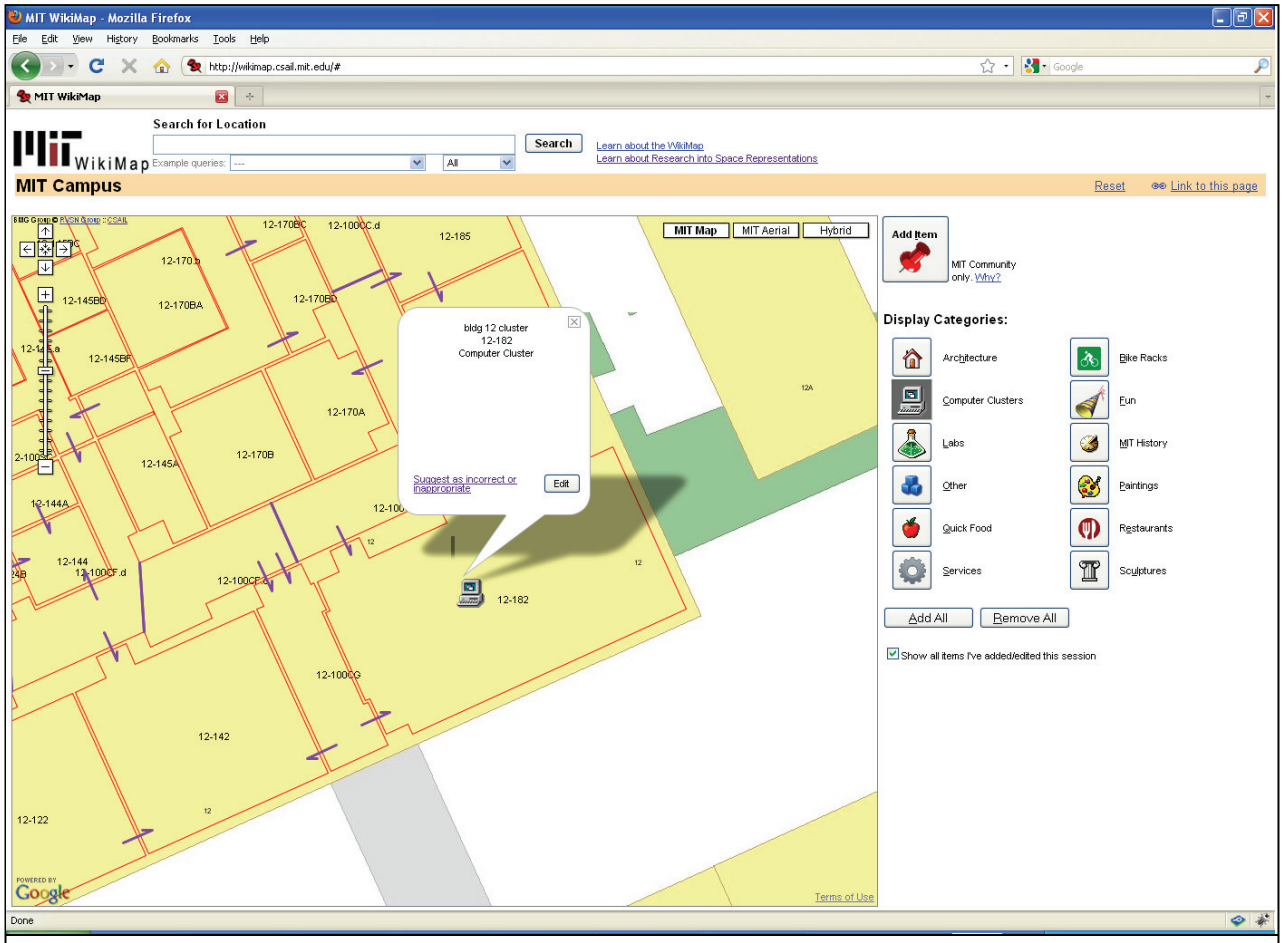

Figure 35. In Figure 9, we demonstrate how semantic information, such as that gathered from Wikipedia, can be integrated into Google Maps for the purpose of semantically annotating geospatial data. Here we present a similar concept, as illustrated in the MIT Wikimap. The difference in this representation is that the annotations belong not to a single point (as they do in current GIS), but to two-dimensional areas. The representational richness of our indoor geospatial representation (geometrically, topologically, and hierarchically) offers more precision and depth when geo-tagging features, and broadens the range of semantic information that can be captured.

hierarchical embedding, although technically possible, is prone to pollution and becomes unwieldy as the geospatial model is "tagged" with more data. This class of arbitrary data is instead most appropriately stored directly with (or "in") the geospatial object, or keyed by the geospatial object's canonical path, and perhaps further annotated with author, date, data quality, and access control.

Thus, a geospatial object is composed not only of its geometric elements and topological relationships, and is contained in a hierarchical family, but also maintains semantic details about itself. For instance, space X is owned by MIT, operated by Facilities, and used by the Robotics group; space Y usually smells like maple syrup; and space Z has a wonderful view of the river (with an accompanying photograph). Spaces can be annotated with these details "expertly" (by an authority of some sort, like the Facilities department) or organically (e.g. by individuals), and the authorship information (which relates to the data's fidelity) can itself be part of the semantic set of information. The idea here, however, is that as arbitrary and unstructured as semantic knowledge can be, ultimately it is linked directly to a geospatial object, just as are other features (Figure 35).

3 Implementing an Indoor Geospatial Model

The previous chapter discussed in detail the conceptual elements of a geospatial model that are essential to the proper representation of indoor environments. This chapter describes an operational implementation of such a model. The system and method introduced here take as input a semi-structured corpus of geospatial data in the form of individual digital floor plans, which cumulatively represent a large-scale, multi-building indoor environment, automatically extract and infer indoor geospatial features and relationships, and generate a model that conforms to the indoor geospatial representation described in the previous chapter. The output of the system is a highly-structured indoor geospatial model that is: at once human-readable and readily machine-parsable; currently comprehensive and highly descriptive, yet extensible in format to support the addition of geospatial features that are not currently represented; and as a representational architecture is light-weight, interoperable, easily maintained and distributed.

3.1 The Building Model Generation Pipeline

The Building Model Generation (BMG) pipeline, or simply *the pipeline* as it is more commonly referred to, is a sequence of cascading geospatial-data processing stages. Geospatial output from one stage is incrementally daisy-chained as input to the next stage, with the process continuing until the desired result is achieved. More concretely, the pipeline is an effectively automated sequence of modules that are linked together to produce a fully-populated indoor geospatial model from the aforementioned corpus of digital floor plans, with the output ultimately for use in the myriad applications that can utilize such a model (Figure 36). The modules involved in this sequence must aggregate an inventory of buildings and floors for a given environment,

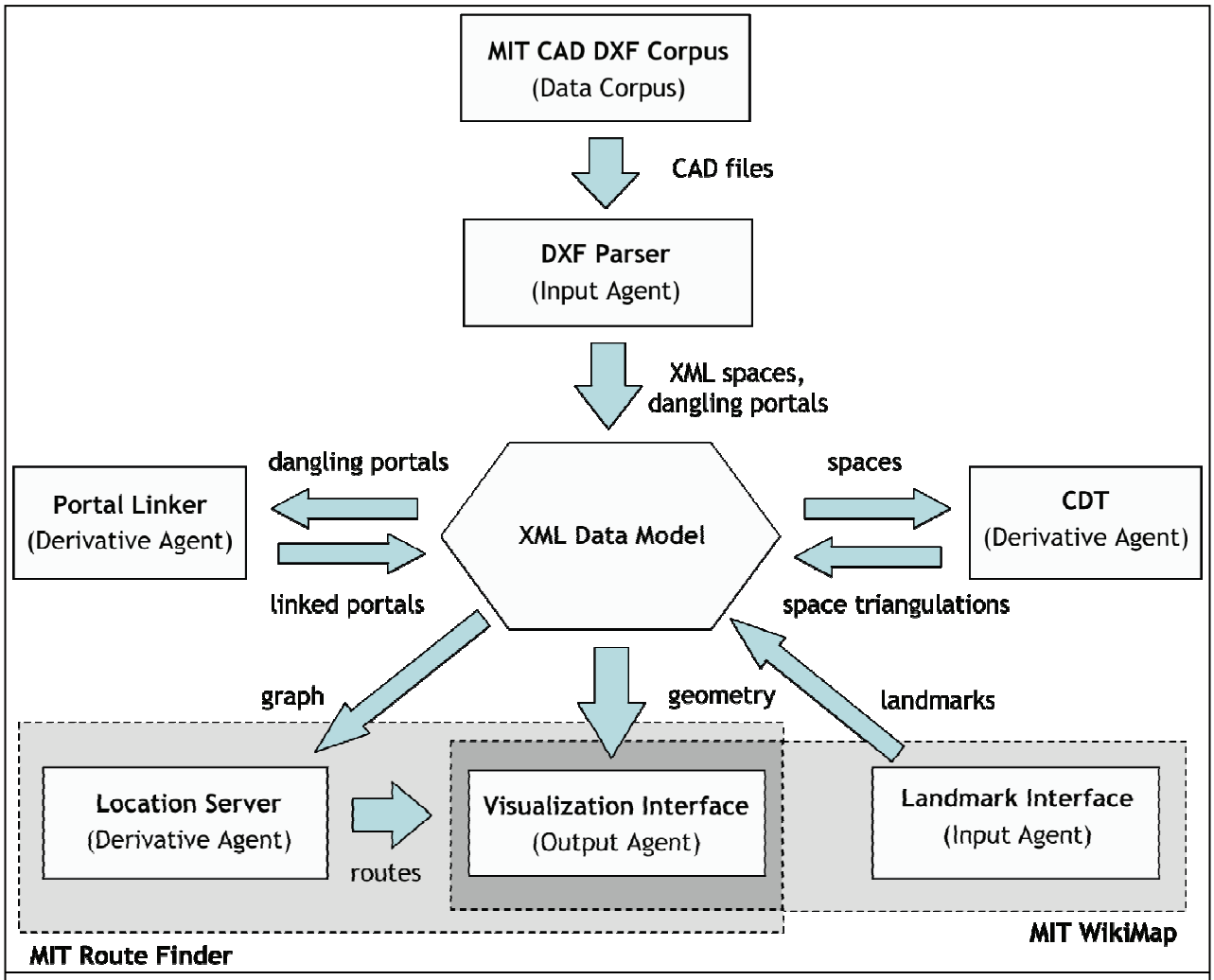

Figure 36. The Building Model Generation (BMG) pipeline is a data pipeline that aggregates a corpus of legacy input data, in our case, a collection of semi-structured floor plans, and processes it through a sequence of cascading stages that extract meaningful geospatial elements. The extracted data is then used to populate an indoor geospatial model, which is made available to location-based services, such as route finders, map visualization tools, and localization services.

create a geospatial hierarchy based on this inventory similar to the one described in section 2.4, collect and parse the relevant digital floor plan files, extract and/or infer relevant geospatial features, and generate a concrete model that is compliant with the indoor geospatial representation defined in chapter 2. The following sections fully describe the various inputs of the system, the internal workings of each of the modules, resulting outputs of the system, and an elegant set of data formats and schemas for storing and operating on the resulting geospatial model.

3.2 System Input Basics

A geospatial model is merely a collection of representational constructs. In order to be useful, it must be populated with actual data. Although future data collection schemes may operate autonomously, past and current methods require manual, human inspection of physical environments and the transcription of such observations. Here we describe an assortment of inputs to the BMG pipeline, which are comprised of various expert-generated sources of legacy geospatial content. These sources, described in more detail in the coming sections, are building inventories, architectural floor plans, and empirically-derived transform parameters for projecting geometry onto real-world coordinate systems.

3.2.1 Background

The geospatial framework described here is demonstrated to work by using the MIT campus as an example input environment. MIT has an indoor environment composed of 848 floors in 162 buildings, covering over 12 million square feet [Facilities2009A]. Each floor has an accompanying digital floor plan that is maintained by the MIT Department of Facilities (henceforth Facilities) and is regularly reviewed and updated by Facilities to reflect ground truth. Furthermore, Facilities maintains (1) current *space accounting reports*, which comprehensively document building and floor inventories, and (2) geometric transformation parameters, which allow a consumer of Facilities' floor plan data to transform floor plan geometry into globally registered geographic information. The following sections describe in more detail the space accounting reports, floor plan data, and transform parameters. In conjunction with the discussion of transform parameters, the topic of coordinate systems will be revisited.

3.2.2 Geospatial Inventory via Accounting Reports

Space accounting reports are records maintained by Facilities to document building, floor and space inventories, their usage, and allocation. The mining of these reports makes possible the programmatic assembly of a representational hierarchy and the subsequent aggregation of floor plan files [Facilities2009B]. These reports are accessible in two ways. First, the reports are available for download on the Facilities website and can subsequently be parsed to extract the relevant building and floor information (Figure 37). Second, inventory information is also maintained in a Facilities database, which can be similarly mined to learn building and floor inventories. For instance, using either of these methods, an inventory analysis tool can learn that there exists a

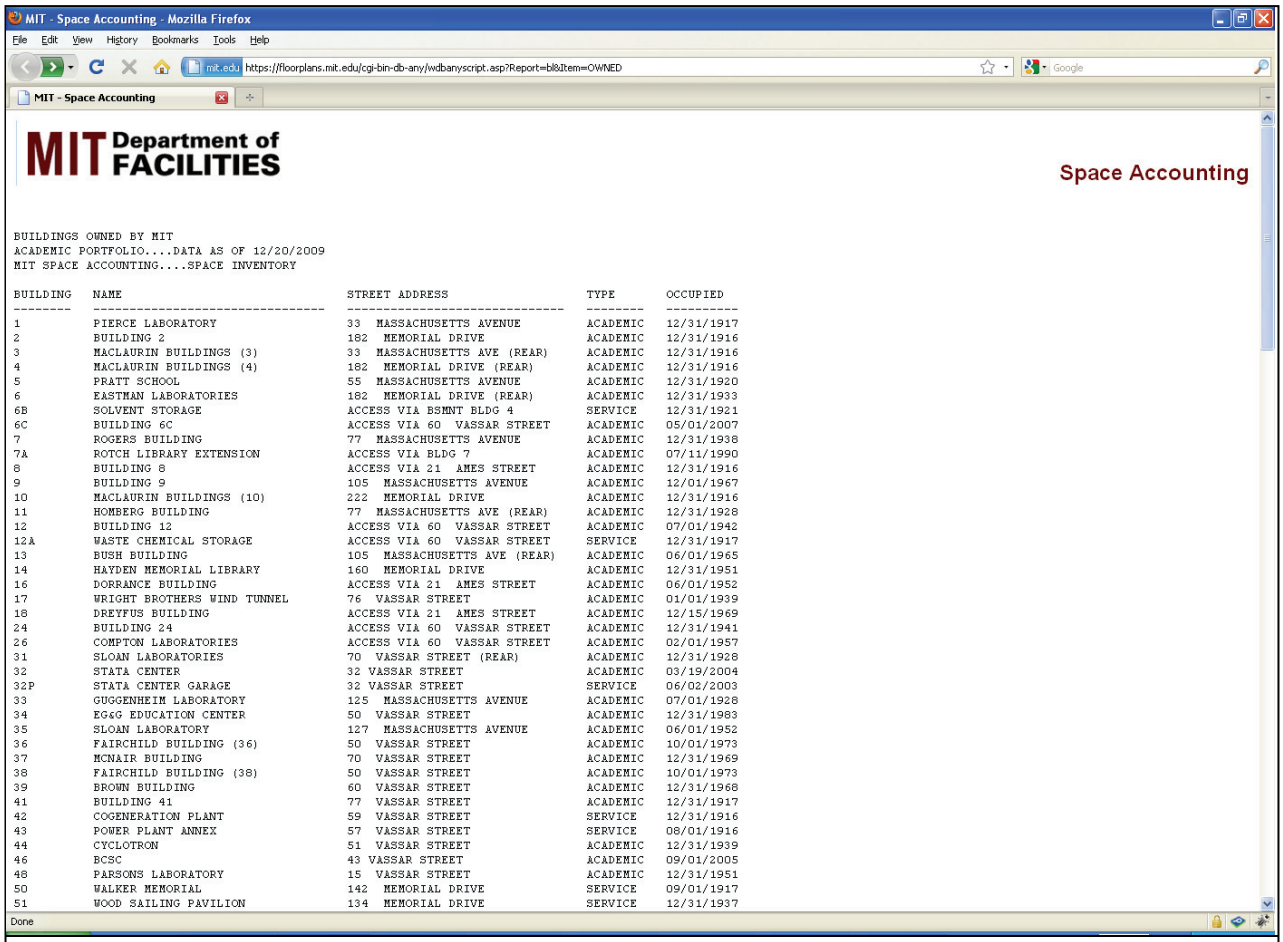

Figure 37. A space accounting report as published on the Facilities website. The above report, a building inventory, enumerates each building on campus by number, gives its full name, address, space use, and date of first occupation.

building *10* on the MIT campus, and that it contains nine floors. Once ascertained, this information can be applied to the next stages of the pipeline, namely a hierarchy-generating module and a floor plan retrieval module. Both of these modules are contained within the *Floor Inventory Manager*, fully described in section 3.4.1.

3.2.3 Legacy Floor Plans

3.2.3.1 Background

The general notion of a floor plan is simple: to describe the precise architectural layout of a given floor in a building. Before the advent of digital drafting, such technical plans were produced

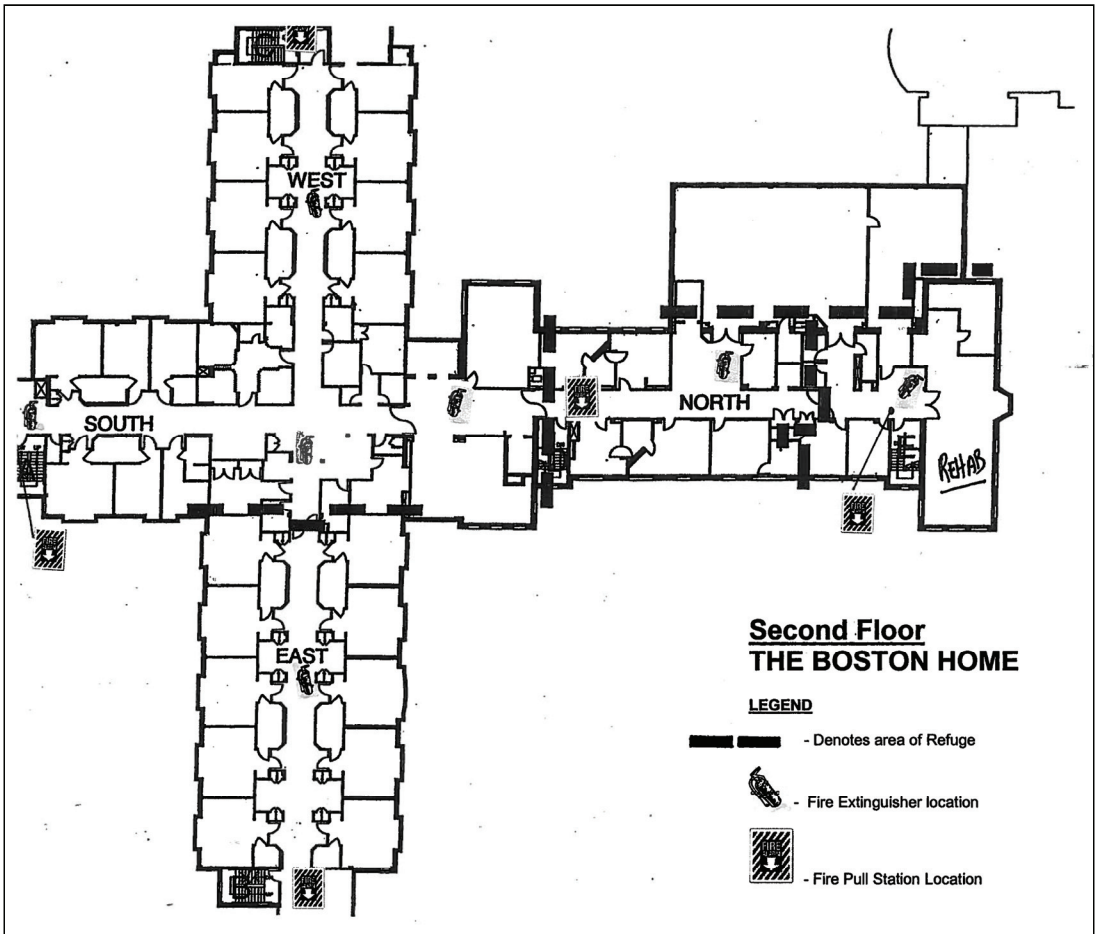

Figure 38. A paper-drafted floor plan of an assisted living facility near Boston. Illustrated are geometric and semantics elements of the indoor environment. Until recently, this low-grade model of the interior was the principal method for capturing and describing the inside of the facility.

on paper (or some other lasting material), and were commonly referred to as blueprints [Ware1999]. With the introduction of digital drafting, however, the landscape of technical drawing has changed substantially. Firstly, digital aids offer a degree of precision and accuracy not easily achievable with earlier drafting methods (Figure 38 demonstrates such an antiquated drafting method). Furthermore, the relative speed with which such technical drawings can be produced and replicated is substantially higher than before, allowing for an even richer architectural composition. Lastly, and most relevant to this author, digital drafting imparts architectural content in an easily exploitable representation. Depending on the nature of the drafting format, the content can range from a fully-structured and easily parsable representation, to completely unstructured representation, which is rather difficult to interpret digitally. Ultimately, the drafting software that is used and the effort put forth by the drafter determine the true depth of abstraction (i.e. degree of meaning) built into the drawing and the associated digital model.

To illustrate the degree of structure that can be present in a drawing, a few simple examples are described. On the unstructured end of the spectrum, a floor can be represented in a raster format, composed of monochrome pixels that represent floor geometry in a flat, 2D grid. Interpreting a drawing in this bitmap representation is non-trivial. The geometry must first be extracted from the image and accurately converted into a vector representation. The vector geometry must then be separated and grouped in semantically meaningful ways. For instance, various geometric constructs must be translated into their architectural equivalents (e.g. arcs for doorways) and similar geometric constructs must be coherently grouped such that elements of one geospatial landmark are not confused with another (e.g. lines composing the outlines of one space must be grouped together and not mistaken for a nearby space).

Some of these complexities can be solved by introducing more structure. For instance, even if we maintain the raster representation, we could instead replace monochrome pixels with colored pixels, where similar geometric constructs share a color (e.g. space contours are one color, doorways are another, and stairs yet another, etc). Color-coding would simplify (but not solve) the step of grouping geometry once it is converted into a vector format, because the architectural properties are effectively codified in color. However, color-coding can be taken only so far.

Another way of introducing structure is by replacing the raster representation with a vector one, because the error-prone and often difficult step of geometry extraction is obviated. In this case, geometry is stored in its native vector representation. Space contours are represented digitally as lines or polygons rather than pixels. The more geometric abstraction that is built into the model, the easier it is to parse, mine, and ultimately interpret. For instance, even if a technical drawing is represented as vector geometry, it can be composed of raw, primitive geometry, such as pure line segments, or it can be composed of more meaningful geometry, such as closed polygons, which implicitly embed semantic information such as space boundaries (Figure 39). In the most difficult case, all the geometry would be represented in a single mass of line-segment "soup". Even though the geometry is provided for "free" as a vector representation, making sense of it is still

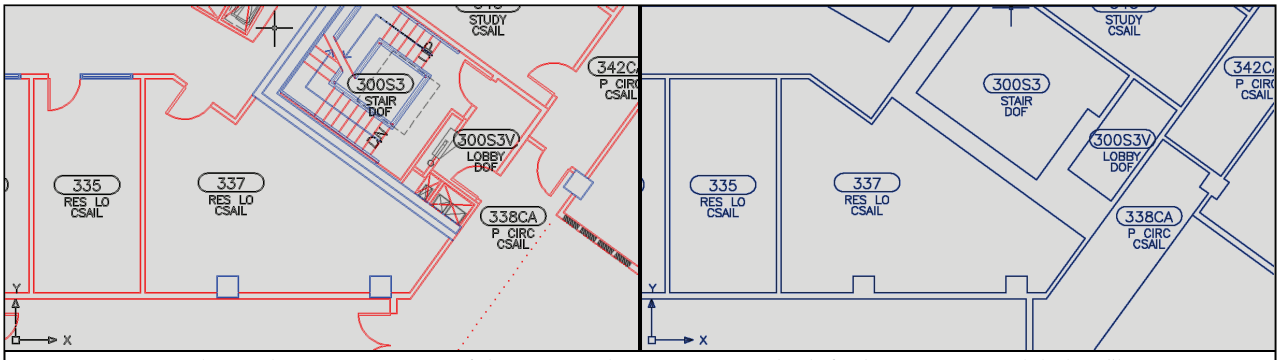

Figure 39. Two identical representations of the same indoor space. On the left, the space is modeled as "line-segment soup." Although architecturally color-coded, it can be difficult to discern which geometry belongs to which space or even where spaces are defined. On the right, each space is modeled as a closed polygon with a defined space footprint.

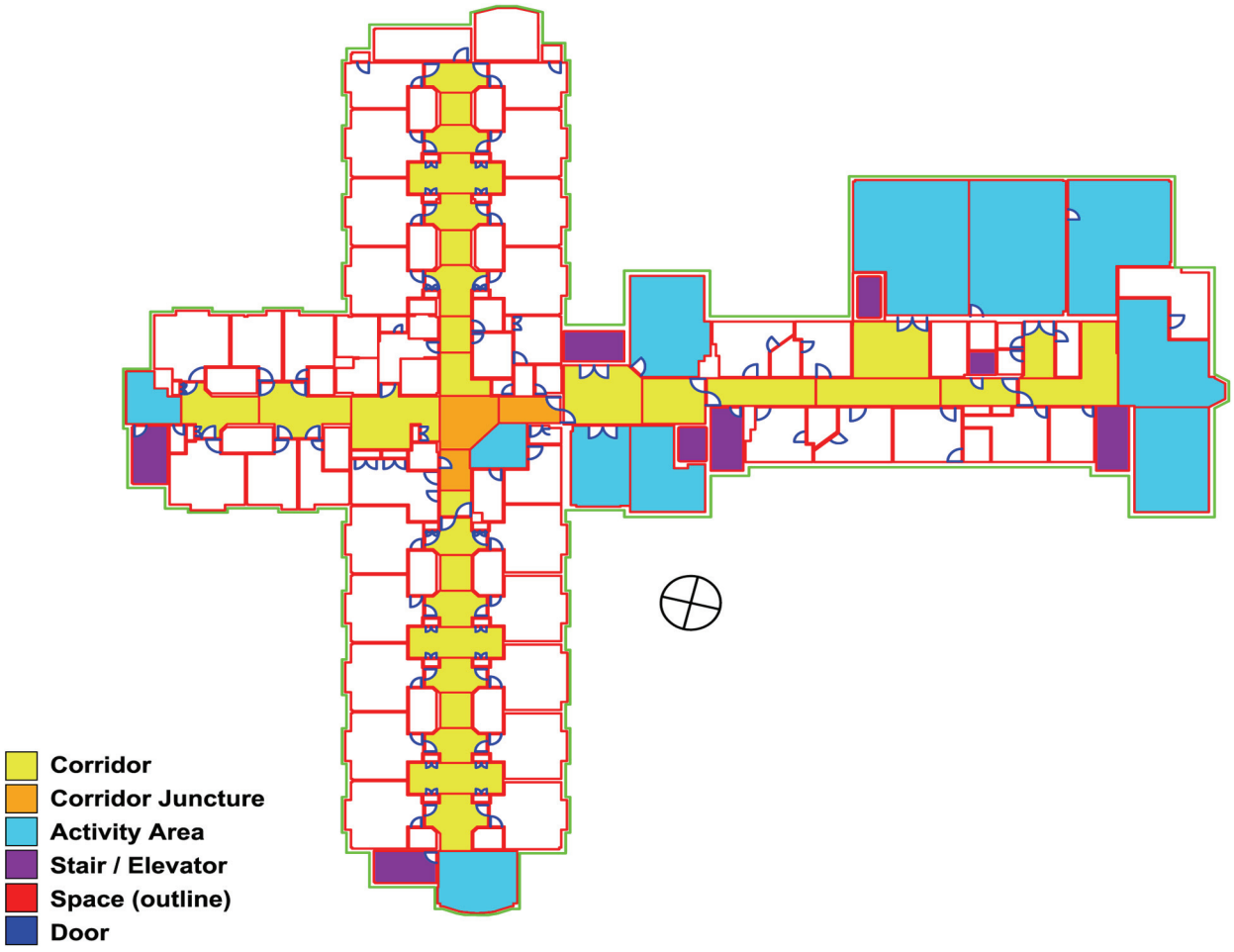

Figure 40. In contrast to Figure 38, the same assisted-living facility is modeled digitally using a geometrically and semantically-abstracted representation of the interior environment. It should be noted that color-coded areas in the floor plan above are introduced in a post-processing step. Space types are textually annotated in the floor plan file.

quite difficult. Alternatively, a better scenario is where line segments are grouped into polygons, and geometry is stored in distinct "layers," which indicate additional semantic value (such as doorways or space contours). The more abstraction the author incorporates into the drawing, the easier it is to populate our indoor building model.

The aforementioned notwithstanding, however, it is necessary to point out that different organizations uphold different motivations for drafting and maintaining floor plans. Among them one will find administrative, logistical, technical, political, financial, educational, and social reasons, as well as others. The drafting specification, the effort put forth, and the overall quality

and richness that an organization devotes to creating and maintaining its floor plans is ultimately tied to its motivations for having the plans to begin with. With that, we proceed to investigate some common cases for floor plan use, and the resulting implications in mining such data.

3.2.3.2 Space Management Standards and CAD

There are many useful, if not essential reasons for drafting floor plans, both with and without the use of computers. Apart from providing a comprehensive, interior view of indoor environments, which can inform all aspects of Facilities management, decision-making, and maintenance, organizational space allocation is today one of the most important aspects of cataloging and inventorying indoor environments. The pioneer in creating an industry-wide standard for facilities management and administration is the United States Department of Education's National Center for Education Statistics, which together with the National Working Group on Postsecondary Facilities, published the *Postsecondary Facilities Inventory and Classification Manual* (FICM).

This document describes at length standard practices for initiating, conducting, reporting, and maintaining a postsecondary institutional facilities inventory [Cyros2006]. As the abstract of the document states, "[The document] reflects the perspective that along with human resources and financial assets, space is one of the primary resources of a postsecondary educational institution. It provides updated definitions for building area measurements, space and room use codes, and other data elements that are useful for including in a facilities inventory." The key elements of this document stress the importance of maintaining clear records of space usage, occupation, ownership, and administration, as well as the value of precise space area measurements.

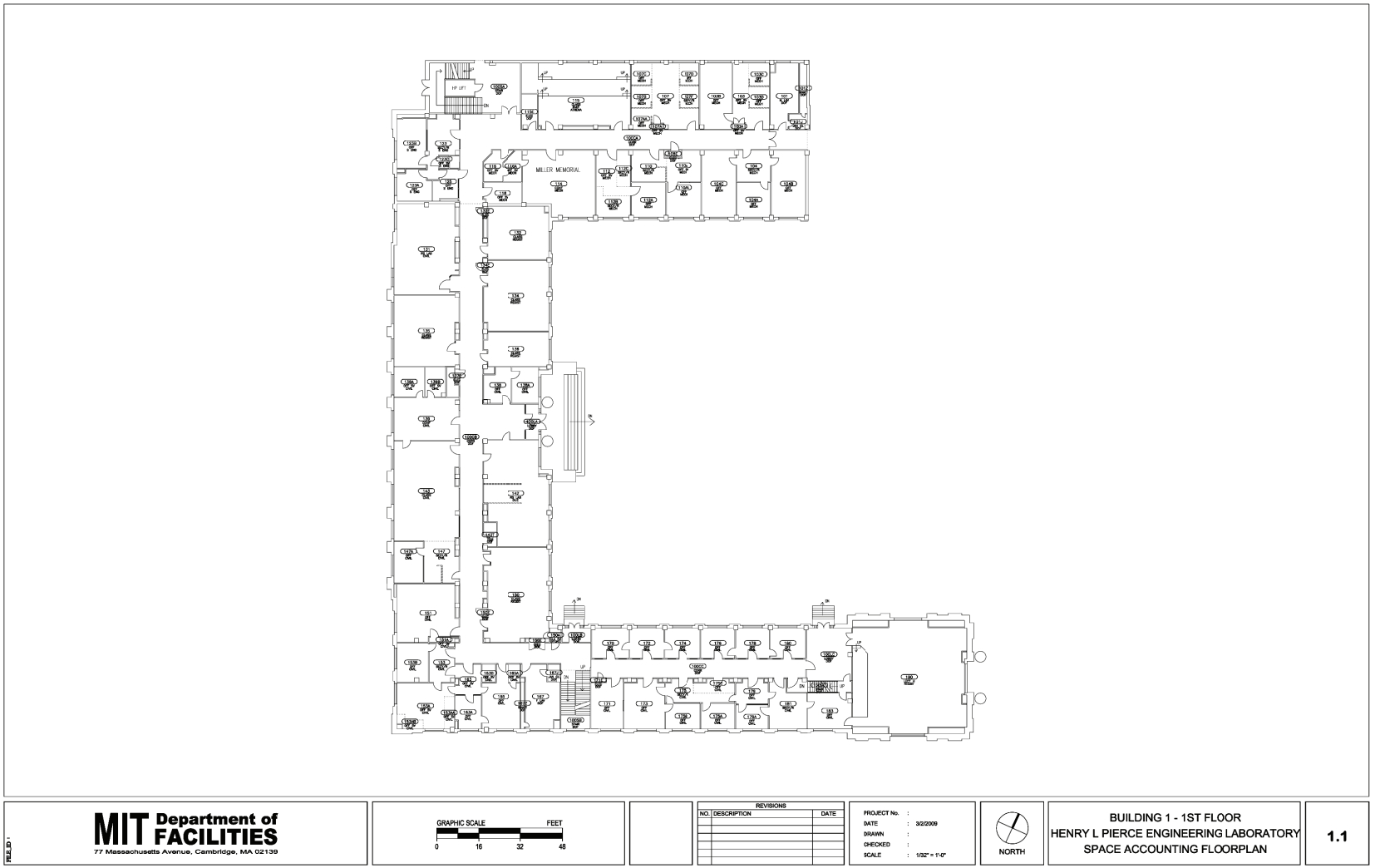

Figure 41. An example floor plan, as drafted by the MIT Department of Facilities using the AutoCAD authoring and converted to a bitmap image. The first floor of Building 1 is represented.

In punctuating the need to collect, report, and maintain these details of indoor environment, the document describes various building and space measurement practices, data reporting, and an exhaustive list of space types, with a special focus on postsecondary facilities. FICM is useful because it provides organizations with both the framework and incentive for systematically collecting and reporting precise, accurate, and meaningful indoor spatial data. Although this data is not in its most generalized form, one that would look similar to the indoor geospatial representation described in the previous chapter, it provides an excellent corpus of content to mine, parse, and use as the substrate for populating a more comprehensive indoor geospatial model.

As a postsecondary institution that relies on government aid, MIT is obliged to survey its indoor environments in a FICM-compliant way. One method among several that MIT employs in accomplishing this reporting is by leveraging the capabilities of CAD packages such as AutoCAD. By integrating a tool such as AutoCAD into the space accounting pipeline, Facilities is able to quickly and programmatically create space allocation and administration reports, as well as richly annotated AutoCAD floor plans (Figure 41). A beneficial consequence of this reporting strategy is that the resulting DXF files contain semi-structured, semantically-meaningful geometry that can be extracted and used as input to an even richer geospatial representation.

For example, in necessitating the precise measurement and reporting of space areas, Facilities represents and drafts the geometric objects that describe rooms as closed polygons with defined geometric areas, rather than simply as lines that visually appear to illustrate rooms. By choosing to represent rooms (and floors) as enclosed geometric entities, Facilities enables floor plan users, such as this author to capture and model discrete spatial entities, not only

geometrically, but by inference also semantically, topologically, and hierarchically. As simple a notion as it may seem, closed polygons create discrete spatial nodes that facilitate the processing of the floor plan data in ways that would otherwise be very difficult. Space labels and descriptions, which are geometrically situated inside or near space polygons, are easily associated to the space object through various straightforward polygonal tests. Topological adjacency information can be similarly inferred or extracted by casting rays from door arcs and seeing where they fall. These various methods are described in section 3.4.2, but they are mentioned here because they are greatly assisted through the adherence of FICM standards.

Ultimately, however, the floor plan only describes a single floor; the drafter can only embed so much information in it. Other pieces of information must be leveraged in order to construct a more comprehensive multi-floor and multi-building indoor geospatial representation. The *Space Accounting Reports* described in 3.2.2 make possible inter-floor and inter-building linkages. Once the space accounting reports are analyzed and a comprehensive assessment of floor inventories is made, the next step is to aggregate the associated floor plans and reconstitute them into a multi-building indoor geospatial model.

3.2.3.3 Digital Floor Plan Formats

The industry-accepted format for creating and storing geometrical floor plans is AutoCAD Drawing Exchange Format (DXF), published by Autodesk, Inc. DXF is a powerful, proprietary drafting format that is designed to capture and represent geometric entities for numerous technical domains, including architecture, manufacturing and prototyping. DXF is the more open sister format to the native AutoCAD format DWG. Whereas the DWG format does not have published specifications or open support from Autodesk, the DXF format is openly published and supported

by Autodesk, with existing mature libraries for reading and writing DXF objects [Autodesk2001]. Thus, the preferred format for handling CAD content outside of the native AutoCAD environment is DXF rather than DWG, , due to its more transparent nature. The specifications for modern versions of the DXF format are published and regularly updated by Autodesk, and the format is generally represented as ASCII (although binary encodings exist, as well).

The open nature of the DXF format by no means implies a simple structure. DXF files are highly complex in their organization, arcane in their formatting, and generally difficult to parse and interpret. When reading DXF files, the published specification becomes something of a bible, which is essential in translating the overwhelming number of properties, generally represented as numerical codes, into meaningful values. Furthermore, CAD files, and especially DXF files, tend to be extremely large in their representation of geometric entities, making file-handling all the more difficult. With that said, there exist numerous libraries that simplify the process of extracting the lowest-level objects and structures in DXF files for higher-level processing. These libraries support most, but not all of the seemingly countless drafting properties that are supported through AutoCAD software, leaving out the more arcane attributes. For our purposes of extracting principal geometry, these libraries work well. The library of choice for this project is known as Dime, which is developed by Coin3D [Coin3D2009].

3.2.3.4 MIT Floor Plans

With this knowledge in mind, we venture into the details surrounding one specific implementation of a digital architectural drawing: the MIT floor plan. The following sections discuss the basic construction and drafting protocol behind MIT's space management model.

3.2.3.4.1 Organization and Structure of the Campus Collection

Floor plans produced by MIT Facilities share a nearly uniform set of conventions: (1) Each MIT floor plan describes a complete floor; that is, a floor plan file retains essentially all spatial information contained inside the floor contour, but nothing beyond it; furthermore, (2) the floor's representation is entirely local in nature. A local representation, as discussed earlier, implies that the information described in the floor plan has no external context. In other words, the data is not *globally registered*, not even within the scope of the MIT campus. This is true on several levels: (1) hierarchically (i.e. placement of the floor on the containment hierarchy), (2) topologically (i.e. placement of the collection of spaces on the space graph), and (3) geometrically (i.e. placement of floor geometry in relation to other floors, buildings, and the world at large). Before the floor plan data can be collected and coalesced into a geospatial model, it must be cataloged and reconstituted into a form that, at the very least creates context between it and other floor plans. Fortunately, Facilities provides enough clues to make this possible.

3.2.3.4.2 Creating Campus-wide Context

First, in terms of hierarchical organization, MIT floor plan files are named and stored in a semantically-meaningful way, which facilitates both their aggregation and their organization. For example, the floor plan for floor 1 in building 10 is named `10_1.dxf`. This semantic annotation makes possible the automatic retrieval of this file and allows the pipeline to hierarchically position the culled spatial data. Specifically, floor plans are stored in a centralized online repository that is downloadable to members of the MIT community: each floor plan is named according to its building and floor numbers, and is suffixed with the DXF extension [Facilities2009B]. Thus, with a

floor inventory, the programmatic retrieval of an entire campus of floor plan data is time- and bandwidth-consuming, but nonetheless straightforward. Using the semantic naming, aggregated files can then be organized into a hierarchically meaningful way. For instance, all floor plans belonging to the same building can be stored in a directory named according to the building name. Further processing of the floor plans would similarly utilize this hierarchical organization.

Next, the local geometric data contained in each floor plan can be straightforwardly transformed from the local coordinate system into a "global" (that is, more contextually useful) coordinate system by way of a matrix transform. This matrix transform, specific to each floor, is similarly maintained by Facilities and rarely needs to be modified or reacquired. Transforming the geometry into a uniform coordinate system is essential for a number of reasons. Firstly, it creates a geometric relationship between all the floor plans. If we are to produce a unified multi-building geospatial model, all the geometry contained in such a model must share their geometric representation. Secondly, and relatedly, maintaining a uniform coordinate system is necessary for properly inferring another relationship: topological connectivity. Because topological linkage is inferred through geometric adjacency, geometric registration into a uniform coordinate system is a prerequisite for such a task. In other words, topological context and connectivity, both between floors (inter-floor, intra-building) and between buildings (inter-floor, inter-building) is dependent on geometry.

3.2.3.4.3 Floor Plan Composition

As discussed in section 3.2.3.3, the industry-standard format for drafting and storing architectural technical drawings is the AutoCAD Drawing Exchange Format (DXF), published by

Autodesk, Inc., and its sister format DWG. MIT floor plans are authored using the AutoCAD software package, and are stored in both the DXF and DWG format.

The drawings are semi-structured in their composition. As described earlier, the structure of a floor plan file can vary widely. On one end, an unstructured plan can be composed simply of raw pixels (essentially just an image), without any further abstraction built into it. On the other end, a highly structured plan might have rich geometric abstractions, adjacencies that are explicitly marked, and geospatial objects that are directly tagged with semantic data. MIT floor plans fall somewhere in the middle of this spectrum. They make moderate use of geometric abstractions (e.g. polylines) and semantic abstractions (e.g. layers). They are annotated, although not explicitly, with some semantic properties (e.g. space names). And the drafting style is for the most part maintained across the entire corpus of floor plans. The latter point is critical when dealing with large scale automated processing. Even though the data is semi-structured, which makes feature extraction relatively straightforward, attributes that are inferred, such topological adjacency and space name labeling still rely on invariant properties across floor plan file.

MIT floor plans owe their uniformity and structure mostly to compliance with the FICM specification. For example, space and floor contours are composed of closed polyline segments so that areas can be easily computed for the purpose of space allocation. Geometry is grouped into layers for similar reasons, thus allowing geometric objects to be easily extracted and conveniently processed. Other elements of structure are likely owed to a drafting discipline that emerged at some point in the history of Facilities' drafting practices. For example, as a rule, space labels always lie within space contours (even for very small spaces), overlapping boundaries of space contours signify the existence of a "ghost" wall (what we refer to as an implicit portal), and door arcs do not

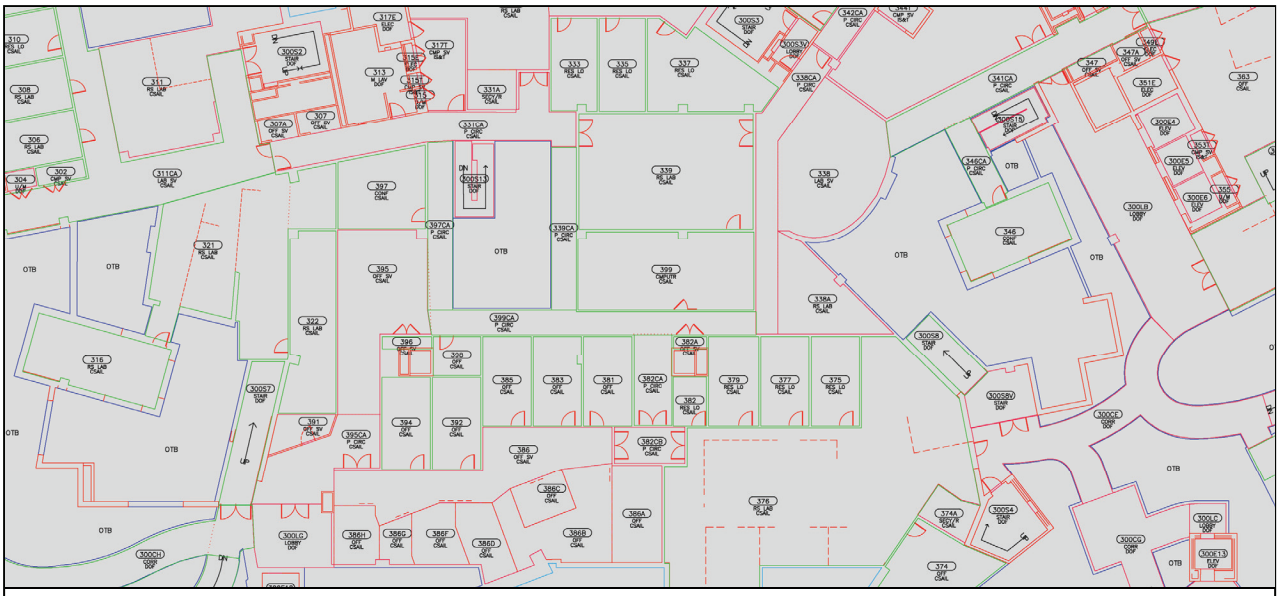

Figure 42. A subsection of an MIT floor plan. The various structural constructs used here are: closed polylines for space foot prints, layers (although not obvious, space contours, labels, door arcs, etc., are each on different layers), consistent space labeling, and the use of implicit and explicit portals. Each of these is discussed at length in section 3.4.2.

exceed a certain epsilon distance from a space boundary (Figure 42). These rules are not set forth in FICM, nor are they enforced by AutoCAD, yet they exist across the corpus of MIT floor plans as invariants, thus making feature inference possible. In rare situations, these invariants are violated, sometimes as bona fide exceptions, other times as errors. If an unexpected result occurs frequently enough, a new invariant is implemented to specifically handle its existence. Floor plan processing and the implementation of invariants are discussed in section 3.4.2.

3.2.4 Coordinate Systems and Transforms

A final piece of the input puzzle relates to the information that allows local geometry contained inside individual floor plans to be translated into a more global context. Each floor plan is represented using a local coordinate system. On the XY plane, this means that (1) the spatial data is usually rotated to be axis-aligned, e.g. the walls of a regularly-shaped floor are roughly parallel with the X and Y axes, and (2) an origin is chosen at the bottom left of the spatial data,

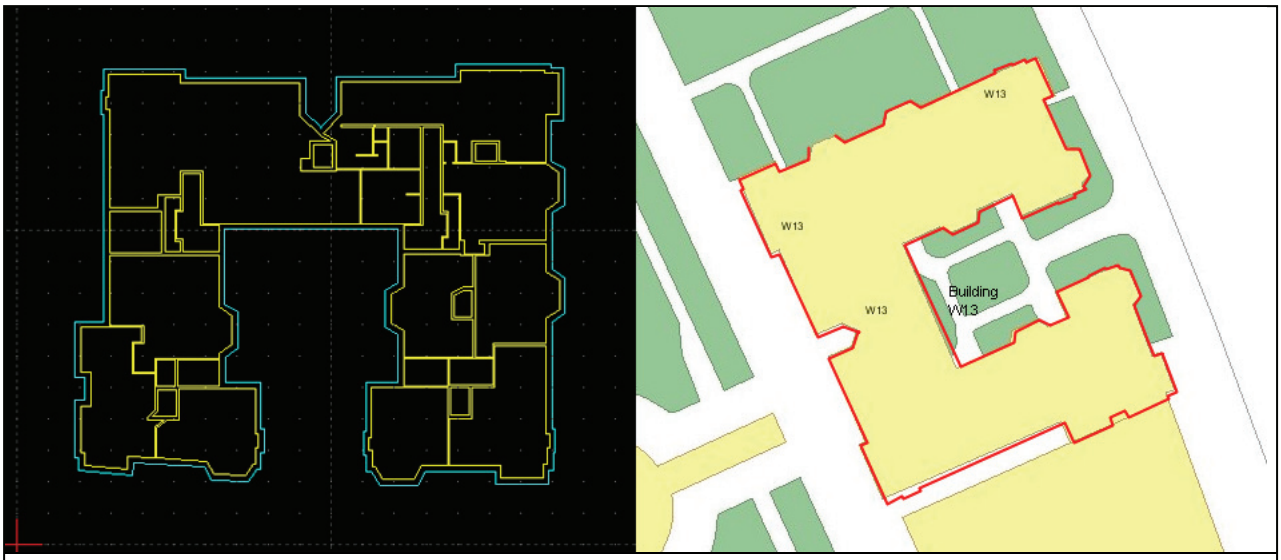

Figure 43. A floor on the MIT campus, as viewed in a floor plan using a local, axis-aligned coordinate system (left), and in a transformed, globally-registered coordinate system (right).

such that all points are in the positive quadrant of the XY plane. In order to globally register the floor plan geometry, each floor must be rotated so that it is properly aligned with other floors in the building, across the campus, and ideally with the world at large. Secondly, a common reference point or reference frame must be chosen so that at the very least, all floors have a relative geometric relationship between each other, and ideally again with the world at large (i.e. global registration). Refer to Figure 43.

Global registration can be accomplished in two ways. First, floor plan geometry can either be transformed into a rectilinear campus-wide coordinate system (i.e. sharing a common origin) with feet or meters as the basic unit. Or second, each floor can maintain its own local rectilinear coordinate system, and otherwise provide only a single global reference point, which "grounds" (i.e. registers) the data in an existing, perhaps non-Euclidean reference frame. As described in the chapter on representation, the coordinate system and units of measure are important. When dealing with indoor spatial environments, it is important that the basic representational unit be

high-precision, rectilinear, and Euclidean. Units of fractional feet or meters fit these requirements well, and thus at the very least, each floor should be represented in a Cartesian coordinate system, with those units as the basic units of measure.

Whether the common reference frame must itself fit these requirements is not clear. For example, could GPS coordinates be used as the common reference frame? Having a common Cartesian coordinate system certainly makes the task of geometric registration between floors and buildings easier, but it also has higher representational demands, is limited in its representational reach given that it is still only a local (e.g. campus-wide) coordinate system, and lastly, requires extra steps to interface with more global (and hence, more common) coordinate systems, like GPS.

On the other hand, using a global reference frame provides the benefits of expanded representational reach and gives the ability to easily register with other data sources. However, it still makes the tasks of local spatial processing and management more difficult. Furthermore, a universal reference frame simply does not meet the basic requirements of an indoor coordinate system (see section 2.2.4), especially when it necessitates on-the-fly localized registration in cross-floor geometric analysis (e.g. when topologically-linking floors).

For these reasons, the framework described here uses a common rectilinear coordinate system for all floors on the MIT campus. The specific coordinate system is the Massachusetts state plane (MASP) coordinate system, which was created by the Massachusetts Office of Geographic and Environmental Information (MassGIS), and was adopted by the MIT Department of Facilities for use in surveying MIT property. MASP is a rectilinear coordinate system that uses a Lambert conformal conic projection to project the curved Massachusetts surface onto a two-dimensional plane (Figure 44) [MassGIS2006]. Its origin is to the southwest of the state, and depending on the

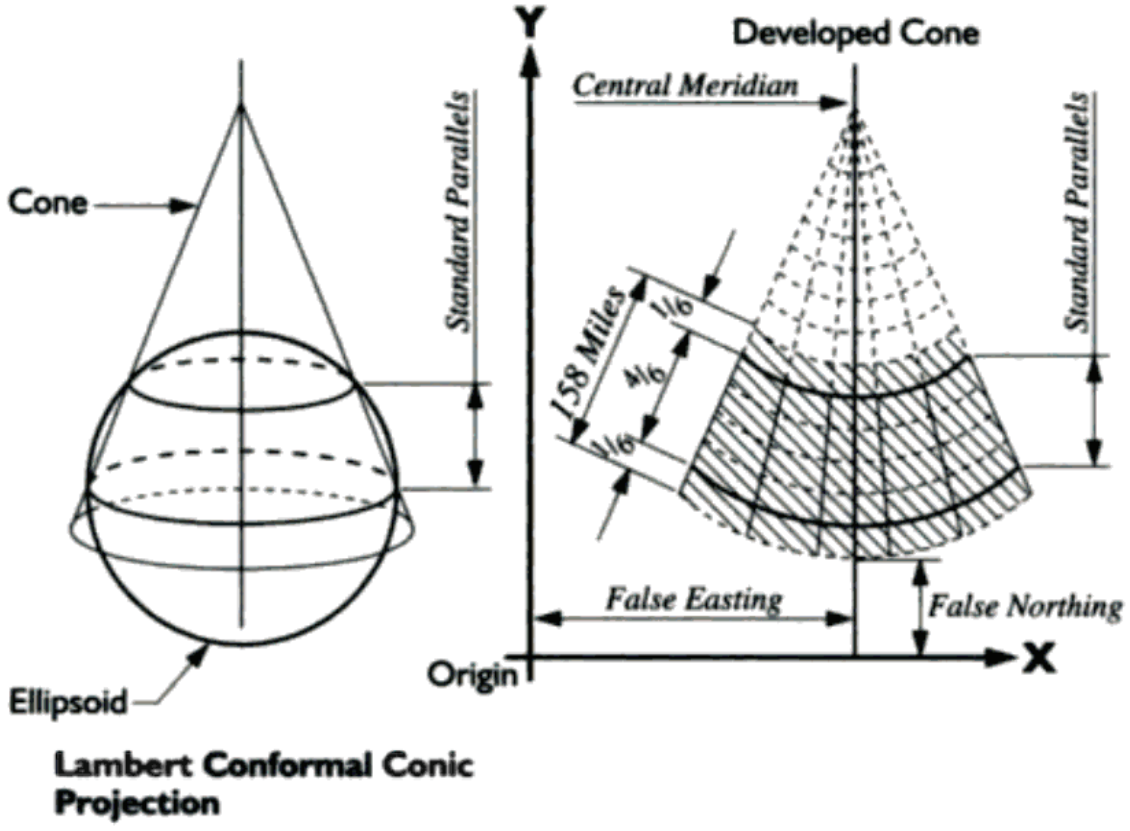

Figure 44. The Lambert Conformal Conic (LCC) projection is the projection utilized by the state of Massachusetts to create a rectilinear (i.e. state plane) coordinate system. With this projection, a cone intersects a sphere at two parallels and the intersected part of the sphere is "flattened" onto the cone. [VanSickle2004]

specific flavor of MASP, the basic unit of measure is either feet or meters. MIT Facilities uses fractional feet as its unit of measure, and the main MIT campus is enclosed in a minimum bounding rectangle defined roughly by ($693x10^3$, $489x10^3$) feet to the southwest and ($723x10^3$, $504x10^3$) feet to the northeast.

In order to transform each floor into the MASP coordinate system, three pieces of information, unique to each floor, are required as input: the degree of translation in X and Y from MASP to the local plane of the floor plan, the degree of rotation between the floor as it rests on the Massachusetts state plane and the axis-aligned geometry of the floor plan, and lastly, any scale factors between MASP and the local coordinate system. The first piece of information

(translational data) was provided by the MIT Department of Facilities in the form of a legacy document, which contains the minimum bounding rectangle (essentially two points) of one space on each floor in MASP. With even only one of these points, translation is easily ascertained given the same point in the floor plan's local coordinate system. Rotational information is conveniently embedded in the floor plan file itself, in the form of a bearing compass that shows exactly which direction the entire floor must be rotated in order to coincide with the global plane (Figure 45). This value, extracted during floor plan parsing, is described in section 3.4.2.

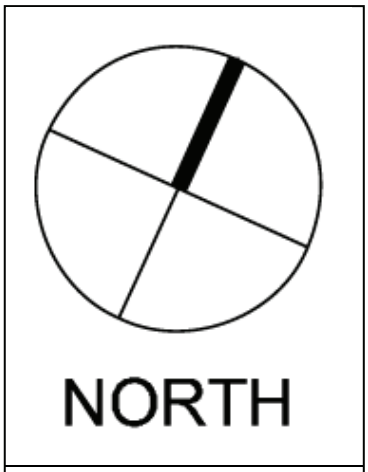

Figure 45. A compass embedded in each DXF indicates the degree of rotation required to register a floor plan with the basemap. Here, the compass indicates a required rotation of 335° CW.

Figure 46. The first floor of building W11, as seen transformed from the floor plan's local coordinate system into the Massachusetts State Plane coordinate system. In particular, it is rotated by roughly 25° CCW, scaled by a factor of 1/12, and translated by the MASP origin.

Lastly, the scale factor is simply a coefficient that, although can be different for each floor plan, usually does not change from one floor to the next, because drafting standards, and hence floor plan unit measures generally remain consistent within an organization. For MIT, floor plan coordinates are in units of inches. Therefore, in order to transform those values into MASP feet, they need to be divided by a factor of 12. A complete floor plan transformation can be seen in Figure 46.

3.3 Data Storage Structures

The previous chapter outlined a geospatial model for indoor building environments that is descriptive and extensive, yet extensible to new forms of geospatial knowledge. Discussed at length were four related but orthogonal elements of indoor spatial environments: the geometric, the topological, the hierarchical, and the semantic. With the conceptual components of an indoor geospatial model fully articulated, and a pipeline to reconstitute non-uniform geospatial objects into members of our newly minted model, we must now develop a concrete data representation to capture and encapsulate the full extent of this model, which demands a flexible and extensible, yet structured and descriptive representational format. With the previous section discussing the inputs to our BMG pipeline, and the next section describing the inner workings and output of the pipeline, namely, the geospatial model itself, here we crystallize the various data structures that are best suited for storing and representing the different data elements of our geospatial model.

3.3.1 Hierarchy

Section 2.4 discussed in detail the properties of a geospatial model *hierarchy*. One of the ideas presented there was the notion of file system directories acting as a metaphor for internal location entities (i.e. collections). If location entity collections can be metaphorically thought of as file system directories, then perhaps an appropriate data structure for *storing* them might very well be file system directories. That is, for each internal location entity (the figurative branch on our tree), we create a corresponding directory, which virtually represents that location entity and has the ability to store its *contained* (descendant) location entities, such as sub-regions and spaces.

The file system directory is a classic demonstration of a tree-like data container. It allows for fast traversal, efficient and distributed storage, and is flexible in its ability to store descendent and symbolic data. Other tree-like containers certainly exist, such as specialized databases and various markup languages, but the file system directory is an easily deployable, highly flexible, architecture- and platform-independent, time-tested solution that reasonably suits our needs.

This data structure, the file system folder, works wonderfully at describing trees. However, it lacks the ability to capture any other properties that are relevant to the container, in this case a location entity collection. For instance, where are a region's semantic and meta-data stored? How are topological and geometric properties of such a collection represented? The file system directory does not address all of these issues by itself. For that, we will need to explore other data structures in order to more completely represent geospatial objects in our model.

3.3.2 A More Comprehensive Representation

Location entities have other properties beyond the notion of containment that must be captured in some way, but cannot be represented through the mere existence of a file system directory. For example, both spaces (terminal entities) and regions (internal entities) have semantic, geometric, and topological properties. Where are those properties stored? Although the possible answers to this question are numerous, two solutions stand out for their respective strengths.

3.3.2.1 A Common Format

One solution dictates that the properties of each location entity are stored in that entity's own, highly-structured file, which captures all relevant geospatial information. This electronic file

would be composed using human-readable character codes (such as ASCII), rather than raw binary, but would also be organized in a highly-structured markup language, so as to be readily machine-parsable. The natural candidate for such a file format is the extensible markup language (XML), a flexible set of rules for encoding documents electronically [Bray2004]. XML is a tree-based file format structure, which means that informational elements can be intricately nested for maximal representational power. (As an aside, the reader may wonder why this tree-based file cannot be used to encode the tree structure of out geospatial model. The response is that from an implementation standpoint, such a method would present scaling difficulties, as it would force the entire model to be stored in a small number of files.) Furthermore, XML informational elements can be richly annotated with attributes, thus providing additional representational capabilities. For example, a space known as 32-337 can be very basically described in XML as follows:

```
1   <space name="32-337" type="OFFICE">
2       <geometry><centroid x="710483" y="496481" /></geometry>
3   </space>
```

Both content-nesting and annotation are demonstrated in the above example. Various properties (geometry in this example) are inserted into a nested tree and subsequently annotated with values that describe the particular location entity.

This storage mechanism, in concert with the file system directory described in section 3.3.1, is a powerful representational combination. The combination fully supports the canonical and symbolic path and naming scheme presented in chapter 2. More concretely, section 2.4.4 illustrated how hierarchy and topology are representationally integrated in our geospatial model. Location entities may be adjacent to each other but are not siblings, and therefore must topologically reference each other across the tree hierarchy. They do so by enumerating each others' path along the tree. For instance, the space 32-300CA is topologically adjacent to the space

36-200LA. The fully-qualified canonical name for each, also known as the URN (see section 2.4.9) would be `/MIT/32/3/32-300CA` and `/MIT/36/2/36-200LA`, respectively. Their file system URLs might be `/opt/models/MIT/32/3/32-300CA.xml` and `/opt/models/MIT/36/2/36-200LA.xml`. Critically, the two spaces would topologically reference each other from within their XML, using the other's URN:

```
/opt/models/MIT/32/3/32-300CA.xml
1   <space name="32-300CA" type="CORRIDOR">
2     <geometry><centroid x="710317" y="496371" /></geometry>
3     <portal class="horizontal" target="/MIT/36/2/36-200LA" type="explicit" />
4   </space>

/opt/models/MIT/36/2/36-200LA.xml
1   <space name="36-200LA" type="LOBBY">
2     <geometry><centroid x="710168" y="496287" /></geometry>
3     <portal class="horizontal" target="/MIT/32/3/32-300CA" type="explicit" />
4   </space>
```

The URN reference, being a tree-based construct, may be provided in either an absolute or relative form, and a software resolver would do the work of mapping the URNs to the appropriate file system URLs for access to the actual geospatial objects. This example demonstrates that the file system structure, coupled with the collection of highly-structured XML files described above can be used to fully capture the representational capability described in chapter 2. The XML format is a natural candidate for encapsulating geospatial properties, both within a single location entity, and across multiple location entities.

3.3.2.2 Optimized Formats

The second, and perhaps complementary solution to the storage needs of a geospatial model, is a database structure that allow for the optimized storage and lookup of certain types of information queries. The previous section described a text-based encoding mechanism that is very

flexible and powerful in its representational capabilities and is easy to maintain, but lacks efficient searchability and transmission optimization, due to its descriptiveness and bulky text-based nature. For these reasons, it is useful to incorporate complementary storage mechanisms, such as specialized databases designed with various optimizations in mind, into an indoor geospatial representational framework. For example, a geospatial database such as PostGIS, which is optimized for geometric information queries, can be used to quickly access and operate on geometric geospatial data. Similarly, semantic data, such as location names and aliases, benefit greatly from a database mechanism that can quickly perform text-based lookups for searching and matching text-based semantic knowledge. Content for these optimized database schemas would be derived directly from the aforementioned XML representation of the geospatial model, with the databases periodically synchronizing to the XML for content consistency.

3.4 Populating the Model – Running the Pipeline

The previous chapters and sections detailed the conceptual representation of a geospatial model, various inputs to the geospatial framework, and the concrete data structures to be used in storing the model. This section discusses in detail the initial stages of the pipeline, namely the acquisition of building inventories, retrieval of electronic floor plans, and the construction of a geospatial hierarchy. Furthermore, it describes the core elements of the pipeline process, namely the parsing of semi-structured digital floor plan files, the extraction and inference of geospatial properties, and population of the geospatial model.

3.4.1 The Floor Inventory Manager

The reader may recall from the previous chapter on representation that buildings are collections, or more specifically topological regions, and contain floors. Floors, too, are collections,

and contain spaces. These nested regions compose a hierarchical geospatial tree. In analyzing the space accounting reports, the Floor Inventory Manager can apply the building and floor inventory information to the creation of the authoritative ("skeletal") tree hierarchy described in section 2.4.6. For instance, before any buildings are discovered, there exists only the root collection of the tree, in our case canonically named "MIT". As buildings are discovered in the reports, they are inserted into the root collection as *children*. Contained floors are subsequently inserted into their specific building collections as children. Once the process of inventory mining is complete, the resulting structure is an authoritative hierarchy of the MIT campus (but for now only a leafless tree), composed of building regions and their associated floor regions. Paths that lead to each region are entirely canonical, and the skeleton of the model is ready for further population, namely by using the floor plan files that are also aggregated in the inventory mining process.

Space accounting reports are maintained by Facilities and can be mined in multiple straightforward ways. As a textual corpus that can be "scraped" from a web server, these inventory reports are simply text files containing spatial members and annotated with other pertinent information, such as space type and square footage. With a limited number of key URLs, the entire campus hierarchy can be straightforwardly traversed by "walking" the URL tree, and a space containment tree can be generated. Furthermore, relevant floor plan files can be acquired using this information. This process is briefly described here. For brevity, notional URLs are used to illustrate the process. A fuller explanation of the process can be found in the Appendix.

Say, for instance, that a list of buildings on campus can be found at the URL `http://floorplans.mit.edu/buildings`. This URL might return a table of data that contains a building number (e.g. 32) in column 1, a building name in column 2 (e.g. Ray and Maria Stata

Center), an address in column 3 (e.g. 32 Vassar Street), and so on. As fixed width text, the data is easily parsed and used as the substrate for the geospatial model. Each building name is used to subsequently access the next level of inventory data. For instance, a list of floors for a given building might be accessible using the URL `http://floorplans.mit.edu/floors?building=X`, where `X` represents a building number. This result, too, is an easily parsable table of data. Column 1 consists of floor numbers (e.g. 1) and the remaining columns contain other pertinent information, such as administrative ownership of a floor (this other data can either be stored and used later for enriching the geospatial model, or it can be ignored). The list of buildings acquired in the previous step is iterated through, and a list of floors for each building is amassed. This floor data provides the next level of depth in our tree hierarchy. It facilitates access to the last, and most interesting part of the Facilities data: the floor plans. For instance, floor plans might be accessible using the URL format `http://floorplans.mit.edu/dxfs/X-Y.dxf`, where `X` and `Y` are the respective building and floor numbers. By further iterating through each building, and each floor in each building, floor plan data can be programmatically acquired. Simultaneously, a concrete canonical tree of regions (i.e. building and floors) is incrementally built, awaiting further population.

Facilities maintains space accounting reports in another form, namely a SQL database that can be queried for building, floor and space information. A geospatial tree can be similarly built using this method – and floor plan files retrieved (via URL) – in much the same way that is accomplished using the web scraping process described above. A discussion of the database-driven approach is left for the appendix.

In summary, the floor inventory manager is a tool designed to mine Facilities' space inventories either via crawling and parsing the online repository or by directly querying the

Facilities database. For each discovered building, and its accompanying floors, the tree of regions is extended and the associated floor plan file is downloaded. This file is stored locally and added to the queue of files to be parsed in the subsequent pipeline stage of floor plan processing. Finally, a summary of the inventory exploration is generated, indicating which buildings and floors were discovered and whether the associated floor plan files were successfully retrieved or not.

3.4.2 Low-level Floor Plan Parsing

With a collection of system inputs to our BMG pipeline, and a concrete representational architecture for digitally storing our resulting geospatial model, the following sections discuss the actual process of aggregating and parsing the input for the purpose of populating our multi-building indoor geospatial model.

3.4.2.1 Preliminaries

As mentioned in section 3.2.3.3, the Dime DXF parsing library was selected as the library of choice in parsing DXF files and extracting the low-level geometry that is present in the floor plan files. The open-source library was written in C++, and is no longer undergoing active development. Code developed for the geospatial framework by this author, used in concert with the DXF library for interpreting floor plan geometry, was authored in C++. Beyond the fact that the underlying DXF library was itself written in C++, this programming language was selected for its runtime speed, flexibility and control in memory management, and ability to easily handle extremely large files. The decision proved wise, as a typical floor plan, containing 50 spaces on average, is processed in under three seconds.

In processing floor plan files, there are several goals that must be achieved, and which have a direct correspondence in the geospatial model. Firstly, and most obviously, floor and space

geometry must be extracted. The reader may recall that generally speaking, floor plans represent the spatial information of one individual floor or parts of an individual floor. In the case MIT, each floor plan represents exactly one floor. Thus, in parsing and processing a floor plan file, the resulting output relates to one whole floor. Geometrically, this includes the floor contour, individual space contours, door arcs and door posts. After extraction, these geometric data are temporarily stored in memory for further processing, awaiting the extraction of other relevant data.

The next element of processing involves the extraction of semantic information. This include space labels (the names of individual rooms), space types (e.g. office), and other semantic information such as space owner, administrator, and user. These data, too, are stored in memory. Lastly, miscellaneous information is extracted from the floor plan, such as the bearing compass that indicates the precise bearing of the floor with respect to North. Once the floor plan is fully parsed and all pertinent information is extracted, the various data are combined, analyzed, and processed in order to yield semantic and topological relationship.

3.4.2.2 Floor and Space Geometry

As discussed in section 3.2.3 and illustrated in Figure 39, space geometry in FICM-conforming floor plans are represented as closed polygons, and stored in semantically-meaningful layers. In particular, contours are captured using the LW_POLYLINE geometric primitive of the DXF standard, which is essentially an ordered list of (x, y) points. Space contours are stored on the A-AREA-FICM layer, and the floor contour is stored on the A-AREA-FICM-GROS layer.

Polylines stored in MIT floor plans have several properties that must be kept in mind during processing. First, points contained in polylines are oriented in a counterclockwise fashion.

Second, due to drafting anomalies, it is possible to have duplicate points in a polyline. Therefore, it is useful to check for duplicates and eliminate them. Lastly, DXF polyline convention dictates that the first point not be duplicated at the end. In other words, technically speaking, the polyline is not closed. However, for our purposes it is useful to close the polyline, and thus we duplicate the first point at the end.

Contours are extracted by traversing the geometric entity list of each layer of interest (i.e. A-AREA-FICM and A-AREA-FICM-GROS), and subsequently extracting the points from each encountered LW_POLYLINE object. Polyline vertices are stored in memory in their local coordinate system, until the relevant transform information (most notably, the rotation parameter) can be ascertain and applied.

3.4.2.3 Space Semantics: Labels, Types, and Other Semantic Attributes

Semantic information regarding spaces is textually stored in the DXF file, in addition to geometric properties. The most useful elements of semantic information are the space names (i.e. the *labels*). Each label is stored in a text block together with the space type (e.g. OFFICE), the space owner (e.g. MIT), and the space user (e.g. EECS). Most importantly, however, the textual block is semantically-linked to a given geometric area by being geometrically situated *inside* the area (Figure 47). In particular, the empirically-derived invariant for linking textual labels to geometric space objects states that the center point of each text block lies inside its parent space contour. Thus, each textual label can be corresponded to a single space contour by iteratively performing a *point-in-polygon* test of its center on each LW_POLYLINE object (gathered as described in the previous section) until a matching contour is found.

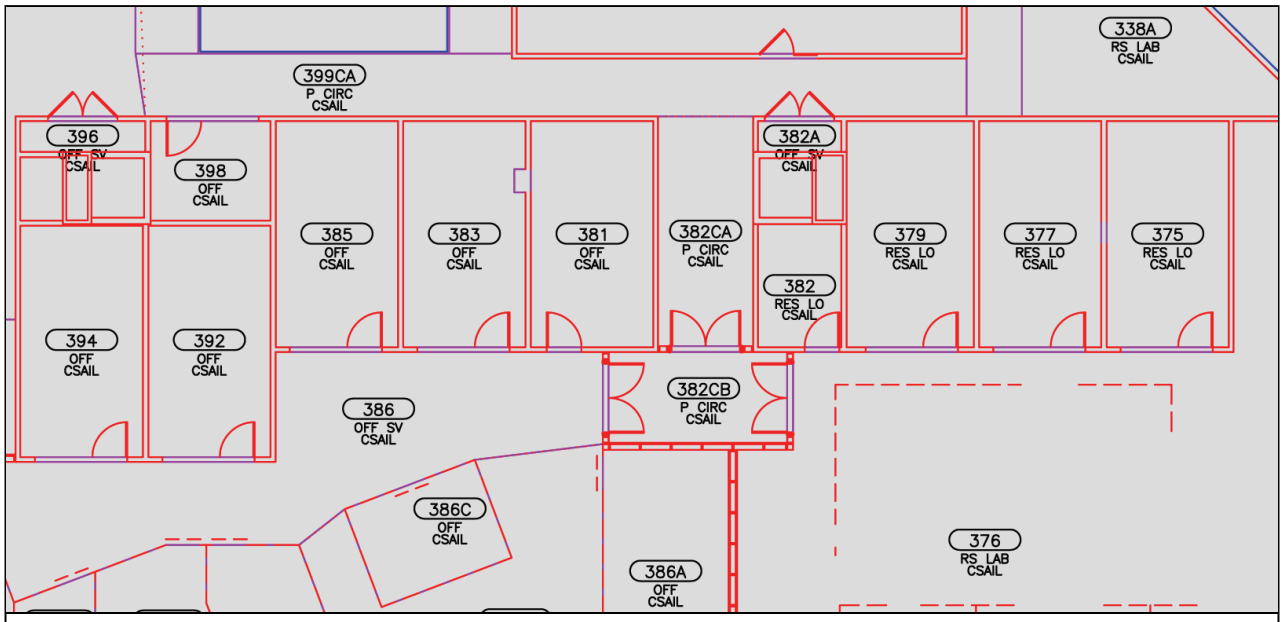

Figure 47. One of the empirically-derived DXF processing invariants specifies that space labels are *always* geometrically situated inside their parent space contour. Notice that even for very small spaces, the center of the space label (the text enclosed by the oval) always lies inside the associated space polygon.

Once complete, this series of tests allows the parser to assign a semantic handle to each geometric space contour. Now, instead of having a collection of anonymous polygons, we have labeled space objects, keyed by name and referencing the geometric contour, space type, and other pieces of geospatial information.

3.4.2.4 Explicit Portal Detection

The parser uses door arc and door post geometry to infer explicit adjacency relationships between spaces. These adjacency relationships represent not only geometric adjacency but topological adjacency, as well, and are thus critical in helping to build the space graph described in 2.3. By deducing the geometric placement and direction of door arcs in relation to space contours, we can iteratively assemble a connectivity graph between spaces.

As seen in various figures such as Figure 42 (p. 75) and Figure 47 (p. 90), doors are represented using 90° arcs that represent a door "sweep," together with a door post represented as

a line segment. The door post is important because it indicates the orientation of a door. If a door arc is situated near two perpendicular walls, it may be difficult to discern the wall to which the door corresponds.

Together, these two elements form a quadrant, and the geometric basis for portal detection (refer to Figure 48 for the remaining portion of this discussion). At the center of this quadrant (45° along the arc, with half the radial magnitude) lies a point referred to as P_1. This point rests inside the originating space, *Space A* of our portal. By running a *point-in-polygon* test (using P_1 as our test point) on the floor's corpus of space polygons, we can readily determine which space contour P_1 lies in and thus assign that space as *Space A* of the portal. Once P_1 is determined and *Space A* is assigned, we can cast a ray from P_1 to determine the destination space. This ray (1) must be parallel to the line segment that represents the door post and (2) should be of the same magnitude (i.e. the magnitude of the door arc radius). The tip of this ray, point P_2 lies inside *Space B*, the adjacent space. Once again, by performing a *point-in-polygon* test using P_2 on the corpus of space contours, we can determine which space is *Space B* in the portal. By iterating through all door arc objects on the floor, explicit portals linking adjacent spaces can be generated and the space graph for that floor is thus formed.

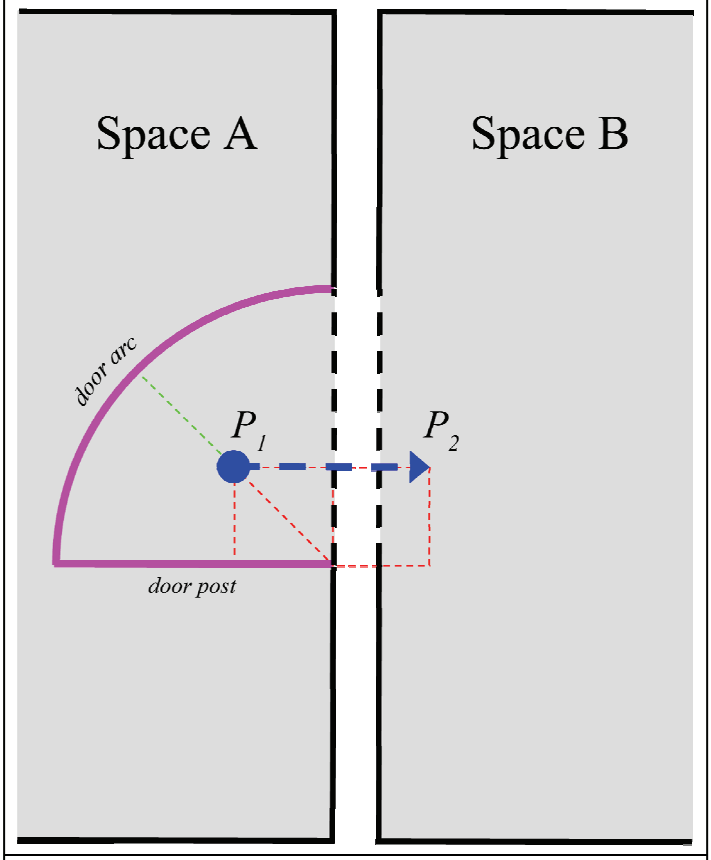

Figure 48. Portals are discovered through a series of geometric tests. Here, the process described in section 3.4.2.4 is illustrated.

3.4.2.5 Implicit Portal Detection

In addition to explicit portals, there is an additional notion of horizontal space connectivity, and that is the implicit portal. An implicit portal indicates the existence of an administrative boundary, but not a physical one. This construct can be useful in a variety of architectural circumstances, such as open lab areas, provisional cubicle areas, large spaces, and so forth. In these situations, there is reason to divide the geometric space into numerous logical partitions, even in the absence of physical partitions.

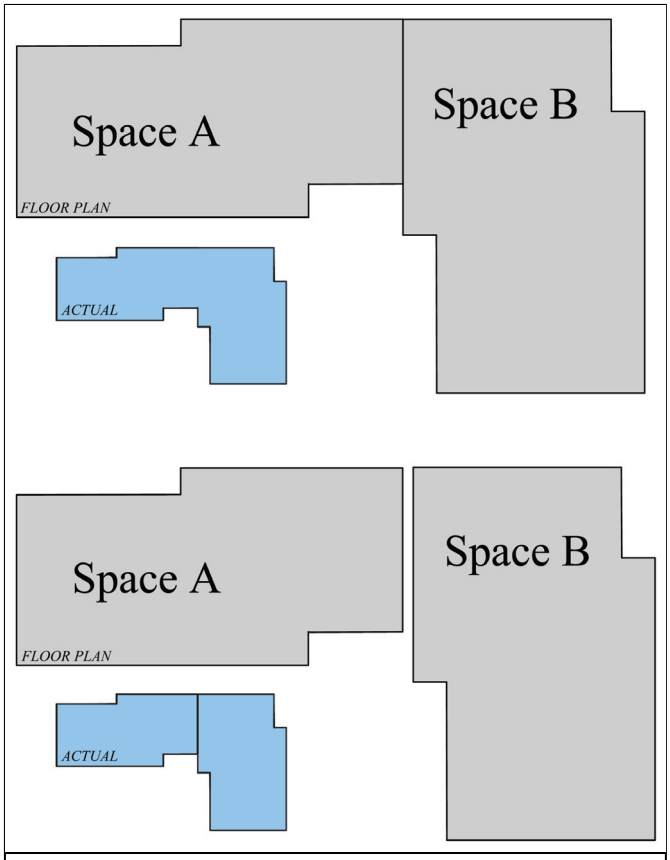

Figure 49. Implicit portals represent a non-physical, "logical" boundary between two spaces. Because spaces are represented as closed polygons, a unique and unambiguous method for representing implicit boundaries is employed: overlapping space contours indicate the absence of a physical, but still semantic boundary between two spaces.

Because implicit portals represent the absence of something physical, creativity is required in geometrically representing them in a FICM-compliant way. The reader may recall from section 3.2.3.2 that, in accordance with FICM specifications, space areas be should be represented as closed polygons, in order to enable the precise measurement of space allocation. For distinct physical spaces, closed polygons make representational sense. However, for open spaces that are merely administratively segmented, a closed polygon would cause representational confusion, because a physical division might be presumed to exist where one does not actually

exist. For that reason, MIT Facilities chose to represent an administrative boundary in a unique way: by overlapping the space contours at the logical boundary.

In other words, when two spaces are adjacent and separated by a physical barrier, their space contours are proximal but not overlapping. However, when two spaces are only administratively divided, but do not have a physical partition, then their space contours overlap at the edge(s) of segmentation (Figure 49). Detecting these implicit space divisions, and hence the implicit portals between them, is conceptually straightforward. If we note that polygons are composed of line segments, then implicit portal detection requires checking for overlapping line segment between space contours. However, this can be a computationally expensive problem, as it requires that each edge of a given polygon X be checked against the edges of all other polygons on the floor (a process that must be done for all space contours). An optimization step can be introduced to minimize the number of polygons that polygon X must be checked against. If two polygons have overlapping space contours, then by definition their bounding boxes must intersect.

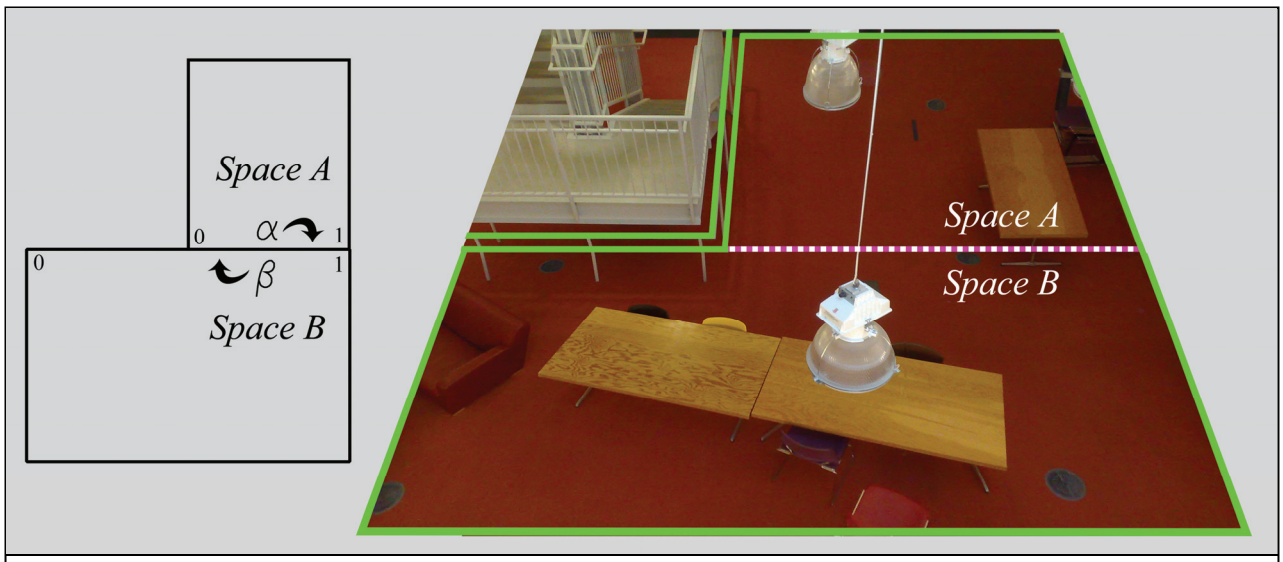

Figure 50. Where an implicit portal exists, the portal is said to lie on the respective edges of overlap of the two involved space contours. In some situations, only part of an edge will overlap with another space contours. Here, edge β of *Space B* partially overlaps only with edge α of *Space A*. In this case, t_1 and t_2 for edge α will be 0 and 1, respectively, whereas t_1 and t_2 for edge β will be 0.5 and 1, respectively.

Checking for bounding box intersection is a computationally cheap test that greatly narrows the number of polygons that must be checked against polygon X for overlapping edges. Furthermore, once polygon X is found to overlap with another polygon Y, then the reverse is true, as well. When polygon Y becomes the subject space, no further comparisons against polygon X will be necessary.

Once two polygons are found to have overlapping edges, an implicit portal is said to exist on their respective edges of overlap. Two parameterized values between 0 and 1 (i.e. t_1 and t_2) are defined, indicating the fractional range of the edge representing the implicit portal (Figure 50). For example, if only a portion of an edge overlaps, then t_1 will be larger than 0 and t_2 will be smaller than 1. If the portal spans the entire edge, then the two values will be 0 and 1, respectively.

Otherwise, the typical elements of a portal (i.e. start space A, end space B, directionally, etc.) exist for implicit portals just as they do for explicit portals.

3.4.2.6 Coordinate Transformation

One final step that is necessary before the floor plan processing is complete and the XML file is generated relates to the coordinate system of the geometric data. The reader may recall that each floor plan maintains its own local coordinate system. The geometry that is extracted from the floor must be scaled, rotated, and translated in order to conform to the designated coordinate system of the geospatial model, which in the case of MIT is Massachusetts State Plane (MASP). As mentioned in section 3.2.4, rotational information is stored in the floor plan file itself, and is extracted and stored in memory during the parsing stage. Scaling is simply a division by 12, in order to convert from the floor plan unit of inches to the MASP unit of feet. Translation, as the reader may also recall from section 3.2.4, involves the use of legacy information in the form of the minimum bounding rectangle (MBR) of one reference space per floor. A legacy data file provided

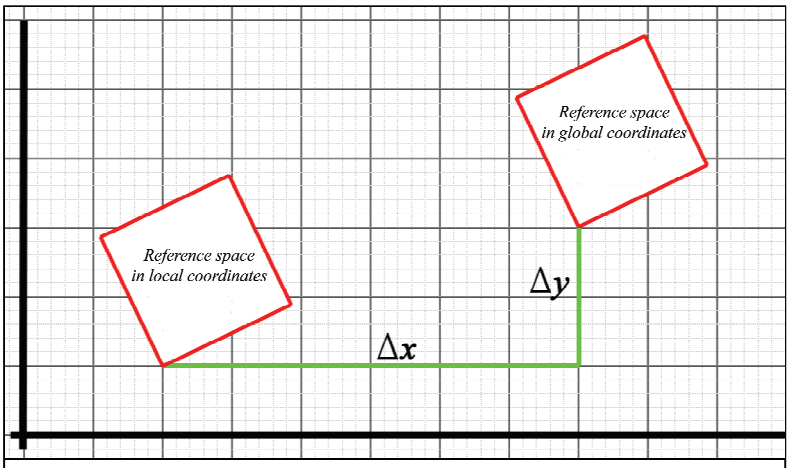

Figure 51. Once a geometric object is scaled and rotated using the appropriate parameters, the final step in coordinate system registration is translation. By using a reference space with the correctly translated (global) values, a (△x, △y) translation point can be deduced and subsequently added to each local point on the floor.

by the Department of Facilities contains a list of MBRs, referencing one arbitrarily chosen space for each floor on campus. When a given floor plan is processed, the MBR data file is summoned and the MBR of the reference space for that floor is loaded into memory. Once all the geometry on the floor is scaled and rotated appropriately via a matrix transformation using the rotation and scale parameters described above, the bottom left corner of the reference space's local MBR, a two-dimensional point, is *subtracted* from the bottom left corner of the space's groundtruth (global) MBR. This yields the two-dimensional translational value for the floor, which must be *added* to each point of geometry on that floor (Figure 51).

As a simplistic illustration, for a given reference space, if the bottom left point of the local (transformed but untranslated) MBR is (10,15) and the bottom left point of the global MBR is (110,215), then the translational value that must be added to each point on the floor is (100,200). Once this step is complete, the floor plan geometry is transformed to the correct coordinate system.

3.4.3 Populating the Model and Producing XML

The previous section discussed the low-level processing of floor plan data that is necessary to yield geometric, topological, and semantic geospatial data. Once the geometry is transformed,

the space semantics (e.g. name, type, owner, etc.) are learned, and spatial topology is inferred via explicit and implicit portals, this information must be transcribed into the XML representation discussed in section 3.3.2.1.

As each digital floor plan is processed, its contents are reconstituted into what becomes the basis of a new XML file. In other words, each floor is encoded in the geospatial model as its own XML file, with the features extracted from the DXF floor plan providing the substrate for the file. In particular, within the XML, the floor is organized as a spatial hierarchy (recall from section 3.3.2 that XML is a tree-based file format and supports, to a limited degree, the direct representation of our hierarchical spatial representation), with individual spaces being stored as "children" of the floor. Geometry, topology and semantic attributes are added to each space in accordance with the format specified in 3.3.2.1, once all dependent information is aggregated and rendered into its final form. When the processing of an individual floor plan is complete, the XML file for the floor is transcribed, and the next floor plan is processed.

The XML file is stored in accordance with the canonical naming scheme described in 2.4.7, and immediately accessible to location-based services that are interested in utilizing its contents. A typical XML file can be seen in the appendix.

4 Applications of Indoor Geospatial Models

The previous chapter discussed in detail the process through which legacy floor plans containing semi-structured, non-standard geospatial objects are reconstituted into a human-readable, readily parsable, coherent geospatial representation, which supports indoor geometry, topology, hierarchy, and semantic attributes. The following sections describe several uses of such a model, from something as basic as a visualization and annotation tool, to a more advanced indoor GPS-like software service.

4.1 MIT WikiMap

MIT WikiMap is an indoor geospatial visualization and annotation tool. One of the first services of its kind, it delivers fine-grain, interactive (i.e. pannable and zoomable) illustrations of over 160 buildings, 800 floors, and 37,000 spaces. It also allows for indoor route-finding and graph-walking queries and provides an interface for contributing and browsing crowd-sourced semantic annotations. The following sections discuss each of these features in detail.

4.1.1 Geospatial Visualization

As discussed in chapter 3, the geospatial model is concretely stored in XML files. These XML files are easily distributed over a network and interpreted by any number of languages. As a markup language, XML is ideally suited for internet transmission and browser processing. Although not ideal due to its lack of structural conciseness, which requires more bandwidth to transmit, XML's format is readily parsed by web applications (as an aside, the simple solution to XML's lack of succinctness is to compress it using base64 or to easily convert it into a more streamlined format, such as JSON).

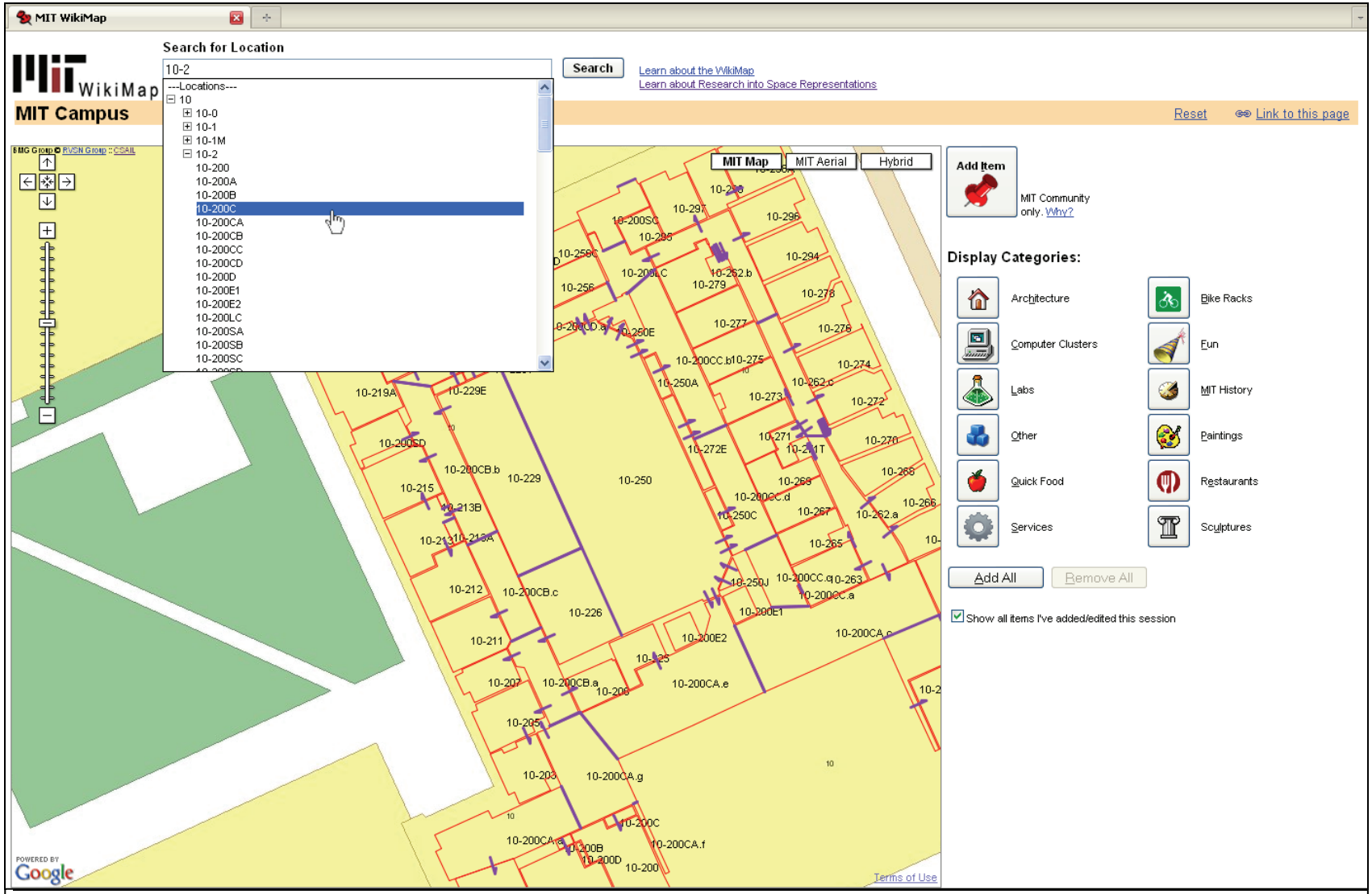

Figure 52. The MIT WikiMap service, as used to apply the geospatial model composed in chapter 3. As can be seen in this demonstration, geometric contours, topological adjacencies, and semantic attributes are all viewable and explorable using the WikiMap.

MIT WikiMap thus utilizes and extends this XML-based indoor geospatial model. The WikiMap web application, available at http://wikimap.csail.mit.edu, directly imports the post-processed XML pipeline product, and uses it to visualize buildings, floors, and spaces, portals and topological paths, and user-contributed semantic annotations. Using JavaScript, geometry is conveniently extracted from the model and displayed in the browser. Users can search for specific geospatial objects (using both canonical and colloquial names), or they can "walk" the region-based spatial containment hierarchy (Figure 52).

Furthermore, users can explore spatial topology by mousing-over portals. The corresponding destination space is displayed, and with the click of the mouse, the adjacent space is selected (Figure 53). In such a fashion, the user can iteratively walk the space graph. Thus, the full breadth of the geospatial model is exposed to the user via this web application.

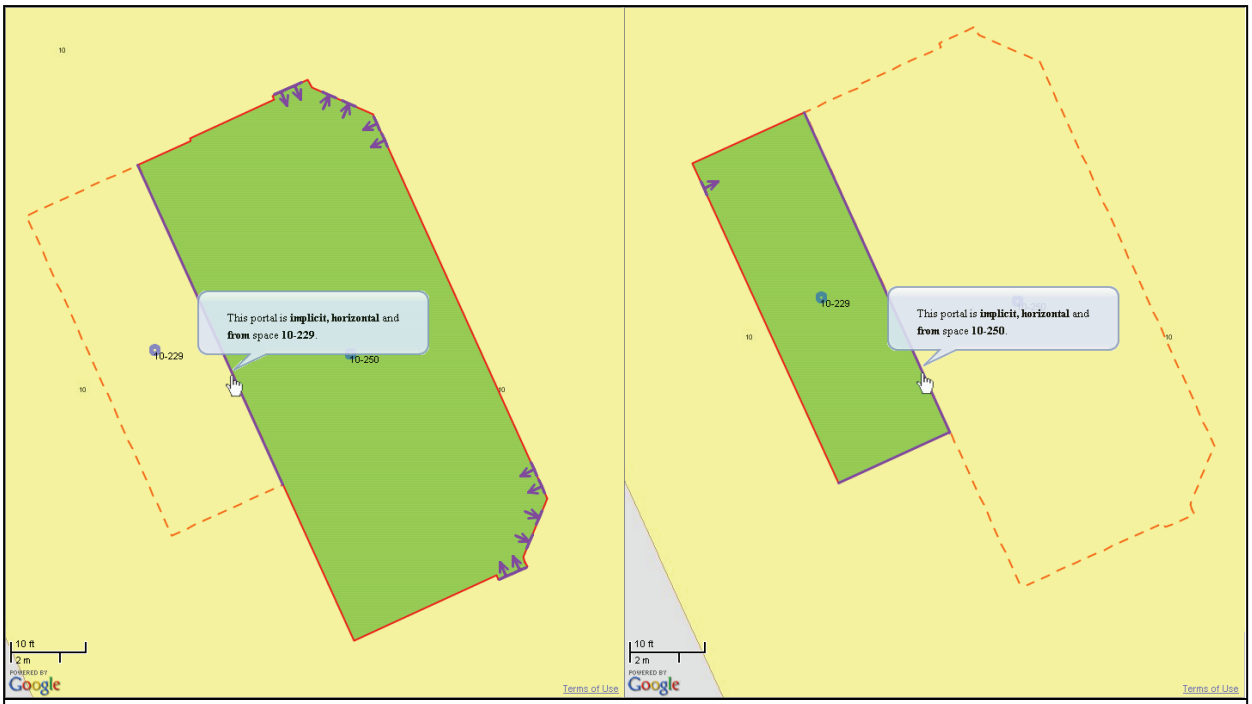

Figure 53. In addition to visualizing geospatial content on a space-by-space or floor-by-floor basis, the model can also be explored topologically. Here, a demonstration of graph-walking is presented, where the image on the left illustrates the user exploring space 10-250 and its neighbor 10-229, and the image on the right shows the result of the user clicking into 10-229 and subsequently exploring it (and its neighbor 10-250).

4.1.2 Route-Finding

In addition to fully visualizing the geospatial representation, there is another feature that can be explored using the MIT WikiMap. This is the route-finding capability mentioned throughout the paper and seen in Figure 16. The route-finder is actually composed in a separate application and is integrated into WikiMap for seamless querying and viewing, together with the other elements of the geospatial representation.

The route finder takes as input two spaces, and performs an A* search across the full campus space graph. The space graph is generated when the route-finding server is first launched. For computational speed, the XMLs of the complete geospatial database are processed and loaded into memory as a lean graph representation. Both before and during queries, optimizations, such as the low-grain graph-walking discussed in section 2.4.8, and pre-computation of short paths, are carried out. These optimizations greatly enhance the performance of the route-finding application.

Once a query is entered into the WikiMap, it is sent over to the route-finding back-end application for processing. A result is computed and subsequently returned to the WikiMap front-end application for display, where it is seamlessly integrated into the visual representation of indoor geospatial features.

4.2 Organic Indoor Location

Another, more sophisticated application that uses the indoor geospatial model described in this thesis is an indoor location discovery service known as Organic Indoor Location (OIL). OIL works by matching real-time, wireless observations of the environment to a known map of wireless

signatures [Teller2008]. When a wireless observation match is found, OIL reports the associated location on the map to the client.

4.2.1 Using the Indoor Geospatial Model for Location Discovery

OIL takes advantage of the indoor geospatial model in several ways. Firstly, as with other location discovery services, the most intuitive user interface for such an application is a familiar geometric map, much like an architectural floor plan. OIL derives such a map using the XML files generated from the pipeline, and displays the map to the user together with its best estimate of the user's location (cf. Figure 1).

Furthermore, although the location estimates are not computed geometrically (i.e. via multilateration), the location estimator does leverage the geospatial model in another way. A pre-existing map of wireless signatures is generated via user-contributions. Users can "bind" current wireless observations to certain locations by informing the system when they reside in those location. The system then associates the real-time wireless observations of the user's device to the location specified by the user, and a wireless signature map is incrementally built.

One can think of these "binds" as semantic attributes of a location. The user selects a space on the map and annotates the space with a piece of meta-information. This information is then stored in a database and semantically-linked to the space, i.e. the geospatial object. In such a way, the location estimator is able to identify a user's location by gathering real-time wireless observations and discovering the closest-matching "semantic annotation" (i.e. a signature on the map of wireless signatures).

Future work on the location estimator may further utilize the geospatial model by exploiting topological constraints in the map to enhance location estimates.

5 Future Work

Although a considerable amount of research has emerged from this thesis, much work remains in improving both the pipeline process and the resultant product. With regard to the pipeline itself, there are many sources of geospatial input, some structured and some not, which could accommodated. These range from ad-hoc, user-contributed datasets, to antiquated, poorly structured datasets. Creating a more robust, more adaptive, and more flexible input handler that can accept a full spectrum of input types, including those that lack coherence, conformity, or the use of conventions, would facilitate the formation of a much richer geospatial model.

As importantly, once inputs have been processed by the pipeline and a product becomes available, this output must undergo considerable validation to confirm that the data processing was sound and that the input was of a high fidelity and accuracy. The following sections discuss these avenues of future work.

5.1 Expanding the Range of Geospatial Input

The key to generating rich geospatial models lies with the input. The quality (accuracy), degree of detail (precision), scale (breadth), richness (depth), and freshness all play a role in how complete, relevant, and useful a geospatial model is. The more knowledge the model has access to, the more powerful it becomes. Unfortunately, the properties of fidelity enumerated above are not always available when dealing with legacy input data, and thus the burden of model enrichment lies with input processing and not the input itself. This thesis demonstrated one implementation for processing legacy geospatial data in order to populate an indoor geospatial model. The input

data was semi-structured, and in that respect was of a somewhat high-fidelity. Not always, however, will access to semi-structured input data be available.

If indoor geospatial models are to truly become ubiquitous, a processing pipeline that relies on more poorly structured data will need to be devised. Section 3.2.3.1 outlined the various levels of structure that can be present in legacy input data. Unfortunately, the lion's share of existing legacy data for indoor environments has minimal structure (i.e. it does not use a drafting specification, nor does it employ a drafting protocol) and is laden with noise (i.e. imprecise or inaccurate authorship, poor digital transcription, representational quality or resolution, etc). A future pipeline processor will need to adequately handle such inputs gracefully, and more importantly, will need to be adaptive in its ability to discern geospatial features. In other words, rather than a developer like this author manually analyzing a corpus of data and building a system of drafting invariants in order to parse legacy data, a future pipeline will need to robustly infer such invariants from a small set of supervised examples, and more importantly, learn from user-provided parsing errors.

For example, a user could provide as input a graphical indoor representation with minimal structure, highlight and annotate certain geospatial features, and allow the software to pattern-match based on the limited set of supervised examples. Alternatively, as pointed out in section 2.5, the direction of geospatial data composition seems to be moving in the direction of organic contribution. For instance, a user might draw a minimal indoor representation using only a few rules (such as contour, door arc, stairs, and so on), and have the system infer a richer representation. If the input was too sparse, other users might come along and fill in the gaps that were left by the earlier user. Similarly, subsequent users might correct errors or vandalism, add

precision to pre-existing structures, update stale data with a fresher equivalent, and so on. The dual ideas here are to make the most of existing data and to minimize (and distribute) the effort required to provide new data. A future version of the pipeline might accomplish these loftier goals.

5.2 Validating a Populated Model

Another important aspect of the geospatial model is data fidelity. As mentioned in the previous section, not all data is guaranteed to be of high-quality or structure. One the one hand, it is up to the processing stage to interpret the legacy data to the best of its ability, in the most accurate way possible. On the other hand, there must be a system of subsequent validation, whereby a pipeline product is sent through a validity test to sanity-check its coherence. This can be

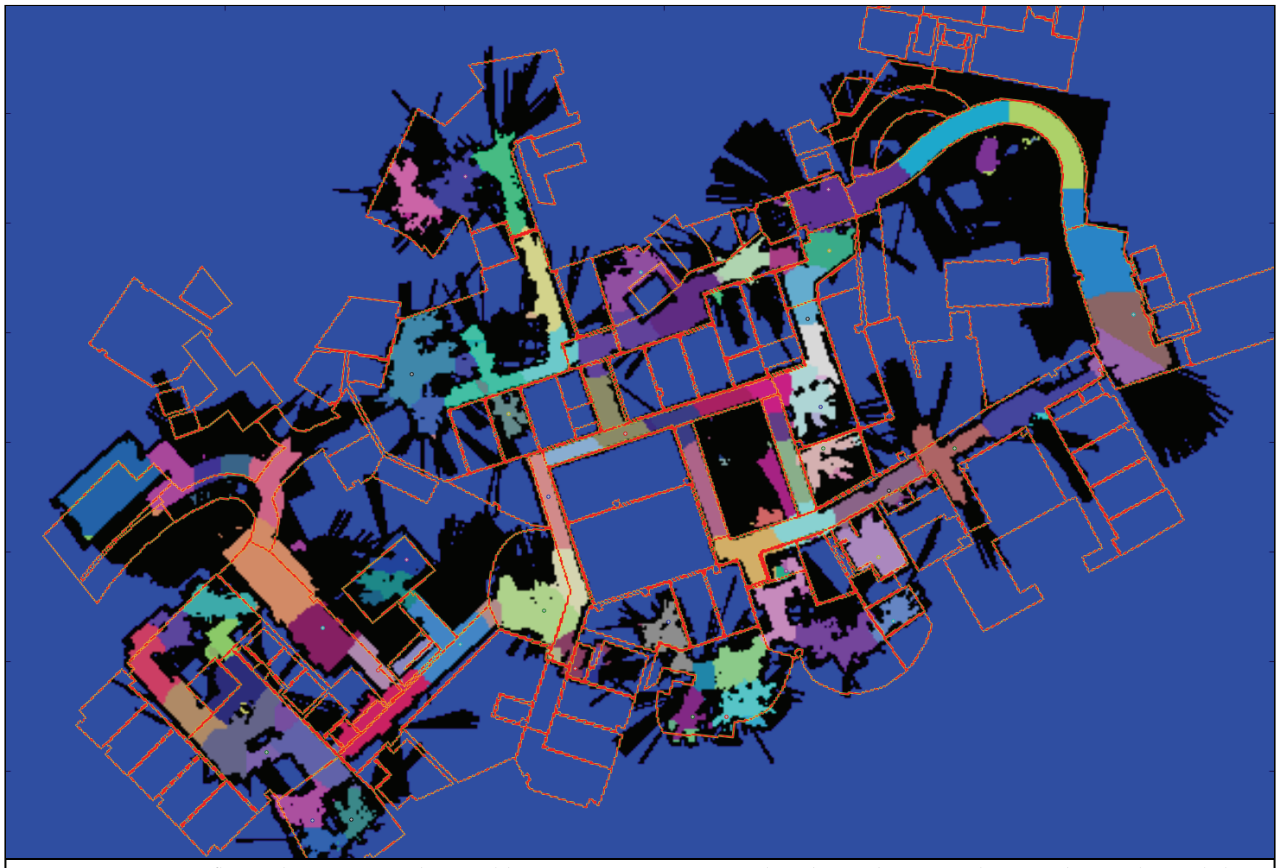

Figure 54. Legacy floor plans, as corroborated by LIDAR sensor groundtruth. Such sensor data can be used to validate existing geospatial datasets for lack of accuracy. Similarly, such sensor data can also be used to populate a geospatial model with richer, more precise, or more recent data.

as simple as making sure that all spaces are topologically-accessible (otherwise, a portal was overlooked somewhere), or as complex as confirming that the integral of all circular paths is zero.

Alternatively, legacy data and its resulting product can be cross-checked and corroborated with groundtruth sensor data, such as LIDAR (Figure 54) or other, orthogonally derived legacy data (for example, a different author or drafting organization). Ultimately, however, data fidelity must be benchmarked in some way, and future work on geospatial models should encompass this aspect of model composition and population.

6 Conclusion

This thesis described a system for extending the current paradigm in GIS to support rich indoor spatial environments. Existing GIS is designed principally for the representation of outdoor environments and required further development in order to properly capture and fully characterize indoor geospatial features. Additionally, new geospatial notions were introduced for this purpose.

In particular, this body of work presented features and properties of indoor multi-building environments that do not exist in other geographic environments or are not characterized in existing geospatial models, and put forward a comprehensive representation for fully describing such environments.

With a comprehensive indoor geospatial model in hand, the next generation of location-aware applications can emerge. Mature outdoor GIS models, coupled with GPS technology, have facilitated the mass-adoption of consumer-facing location-enabled devices and services, such GPS navigation devices and coarse-grain friend-finders. As indoor geospatial models become more prevalent, more accurate, and richer, a new paradigm in location-based services will emerge. This paradigm will shift from low-resolution, outdoor location-based services, to fine-grain, indoor "richmap" services, and will similarly facilitate the next wave of mobile, location-enabled technology adoption.

Glossary

Adjacent – a topological and geometric property that indicates immediate proximity. In the topological sense, this means connected via a single edge (i.e. "one hop"). Geometrically, this means there are no intervening areas.

Adjacency-type – this refers to several properties of the adjacency relationship, both topological and geometric. For example, it can indicate whether connectivity is horizontal or vertical, explicit or implicit, indoors or outdoors, intra-floor or inter-floor, and so on.

Area – a section of the *xy*-plane of arbitrary shape and size, often used to describe spaces.

Atomic spatial unit – the minimum quantized geospatial feature. Topologically, used to describe nodes on the space graph. Geometrically, used to describe semantically-indivisible areas. Also referred to as a space.

By-reference – to link to an object via an address (e.g. in memory or storage) rather than duplicate the contents of the object. Akin to a programming *pointer*. Useful when referring to geospatial data through alternate means, such as a nickname.

Canonical name – the primary, or authoritative way of describing a geospatial object, typically defined by the organization. Usually the target of a symbolic or by-reference link.

Canonical path – the primary, or authoritative way of hierarchically accessing a geospatial object in the model. The recommended canonical path uses a region-based hierarchy, which mirrors regional containment found in actual indoor environments.

Class (of portal) – indicates whether the portal is horizontal or vertical.

Collection – nestable location entities that may contain other collections or spaces.

Colloquial name – a common name or alias given to a geospatial object, such as a nickname. Contrast with canonical name.

Connected component – a topological graph that has no breaks. Each node is reachable by all other nodes.

Dimensional direction – indicates whether an edge moves along the *xy*-plane or the *z*-axis. See class (of portal).

Edge – a topological notion that indicates immediate connectedness between two nodes. See also adjacent.

Edge direction – indicates whether the edge points from *node A* to *node B*, vice versa, or bidirectionally.

Explicit connection (portal) – indicates that topological adjacency is defined via explicit architectural cues, such as doorways.

External-memory – an element of data storage that is not resident (i.e. immediately available), and can only be accessed in a resource-intensive way. The goal is to minimize the need to access external memory, or at least make the process asynchronous and transparent to a client.

Fully-qualified canonical name – a concatenation of the canonical path and canonical name, which indicates the authoritative and complete access handle of a geospatial object.

Geospatial model – a means for representing geospatial objects and their associated geometric, topological, hierarchical, and semantic properties.

Implicit connection (portal) – indicates that topological adjacency is defined not by architectural cues but rather through administrative or logical partitioning.

Internal location entity – see collections.

Layer – an existing notion in GIS, whereby geospatial objects can be semantically grouped in a limited fashion.

Level of detail (LOD) – the degree of detail appropriate for a certain context. For instance, at low zoom levels, a visually-low level of detail is appropriate, whereas at higher zoom levels, more detail is useful.

Location entities – geospatial objects with geometric, topological, hierarchical, and semantic properties. Location entities can be either terminal (see space) or internal (see collection).

Location-based service – an application that utilizes a client's location to provide useful and interesting geospatially-relevant information.

Minimum bounding rectangle (MBR) – in relation to a set of two-dimensional points, the MBR is composed of four values: the minimum x and y values, and the maximum x and y values. Also known as the bounding box.

Namespace – a naming abstraction that constrains all members of a group to have unique names.

Node – the basic unit of a topological graph, and often a notional representation of some object or entity that is symbolically-connected to other objects or entities.

Out-of-core – the ability to intelligently handle and manipulate large datasets through the use of external memory, introducing only minimal latency and inconvenience to the user.

Path – topologically, a set of edges that connect one node to another. Geometrically, a linkage of line-segments or two-dimensional areas that connects two landmarks.

Polylines – a linked set of line-segments. A closed polyline is a polygon.

Portal – a geospatial construct of connectedness, both topological and geometric. See edge.

Portal pair – sibling portals that define a topological and geometric linkage between three spaces.

Region – a collection where all members comprise a connected-component.

Richmap – a collection of geospatial objects that are heavily annotated with geometric, topological, hierarchical and semantic properties, both expert-driven and organically-sourced. Often used as the substrate of location-based services.

Semantic (attributes) – properties of a geospatial object that relate to its purpose, meaning, or identity.

Sibling portals – two portals that belong to the same space.

Space graph – a collection of spaces that are topologically-connected via a network of portals.

Space – see atomic spatial unit.

Spatial containment – the notion of geospatial hierarchy, whereby one geospatial entity contains another, often in a nestable fashion. For instance, buildings contain floors and floors contain rooms.

Terminal location entity – see collection.

Topology – a concept that models symbolic-connectedness. For example, through a graph of connected nodes.

Type (of portal) – indicates whether a portal is explicit or implicit, for horizontal portals, or whether it is a stair, elevator, or ramp for vertical portals.

Unique identifier – an unambiguous access handle to an object, typically unique only within a namespace.

Bibliography

Autodesk, Inc. *AutoCAD 2002 Drawing Exchange Format (DXF) Specification v.u16.1.01*, originally published 2001 [http://www.autodesk.com/techpubs/autocad/dxf/dxf2002.pdf]

Boora, *Leed Platinum Studio* demo. October 2009. [http://www.boora.com/projects/boora_leed_platinum_studio]

Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E. and Yergeau, F. *Extensible Markup Language (XML) 1.0* – 3rd edition, originally published 2004. [http://www.w3.org/TR/REC-xml/].

Coin3D, *DIME DXF Parsing Library*. August 2009. [http://www.coin3d.org/lib/dime/]

Cyros, K., Korb, R. Postsecondary *Education Facilities Inventory and Classification Manual (FICM): 2006 Edition*. NCES 2006-160

Department of Facilities, Massachusetts Institute of Technology. *Floor Plans Archive*. January 2010. [https://floorplans.mit.edu/searchPDF.asp].

Department of Facilities, Massachusetts Institute of Technology. *Space Accounting Overview*. August 2009. [https://floorplans.mit.edu/reports.asp].

Funkhouser T., Séquin C., and Teller S. *Management of Large Amounts of Data in Interactive Building Walkthroughs*. In Computer Graphics (Proc. 1992 Symposium on Interactive 3D Graphics), 25(2), pp. 11-20, March 1992.

Funkhouser T., Teller S., Khorramabadi D., and Séquin C. *The UC Berkeley System for Interactive Visualization of Large Architectural Models*. In Presence, Vol. 5, No. 1, Winter (January) 1996, pp. 13-44.

Giovine, G. *Federating Data Sets for Use In a Map Context*. M.Eng Thesis, Massachusetts Institute of Technology, June 2006.

Jiang, B., and Claramunt, C. *Topological analysis of urban street networks*. In Environment and Planning B: Planning and Design, Vol. 31:151-162, 2004.

Lewis R. *Generating Three-Dimensional Building Models from Two-Dimensional Architectural Plans*. Master's Thesis, University of California, Berkeley, May 1996.

Longley, Paul. *Geographic Information Systems and Science*. West Sussex, England: John Wiley and Sons Ltd, 2005.

Look G., Kottahachchi B., Laddaga R., and Shrobe H. *A Location Representation for Generating Descriptive Walking Directions*. In IUI '05: Proceedings of the 10th International Conference on Intelligent User Interfaces, pp. 122-129, San Diego, CA, USA, January 2005.

MassGIS, Office of Geographic and Environmental Information, Commonwealth of Massachusetts Executive Office of Environmental Affairs. *Datalayers/GIS Database Overview*. March 2006. [http://www.mass.gov/mgis/spc-pts.htm].

Mealling, M. and R. Denenberg, *Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations*, RFC 3305, August 2002.

Niederauer C., Houston M., Agrawala M., and Humphreys G. *Non-invasive Interactive Visualization of Dynamic Architectural Environments*. In Proceedings of the Symposium on Interactive 3D Graphics 2003, pages 55–58, 2003.

Nichols, P. *Location-Aware Active Signage*. M.Eng Thesis, Massachusetts Institute of Technology, January 2004.

Shekhar S. and Chawla S., editors. *Spatial Databases: A Tour*. University of Minnesota: Prentice Hall, 2003.

Star, J., Estes, J. *Geographic information systems: An introduction*. In Geocarto International, 1752-0762, Volume 6, Issue 1, 1991, Page 46.

Symbian, All About Symbian Blog. *SEE 2009: NAVTEQ*. October 2009. [http://www.allaboutsymbian.com/news/item/10610_SEE_2009_NAVTEQ.php]

Teller S., Battat J., Charrow B., Curtis D., Ryan R., Ledlie J. and Hicks J. *Organic Indoor Location Discovery*. Technical Report CSAIL TR-2008-075, Massachusetts Institute of Technology, December 2008.

Teller S. and Séquin C. *Visibility Preprocessing for Interactive Walkthroughs*. In Computer Graphics (Proc. Siggraph '91), 25:61-69.

Timpf, S. and Frank, A. *Using Hierarchical Spatial Data Structures for Hierarchical Spatial Reasoning*. In Hirtle, S. and Frank, A. (eds.), Spatial Information Theory, pp. 69-83, Springer, Berlin, 1997.

TomTom NV. *Premium Technologies ... Available on New TomTom ONE and TomTom XL*. BusinessWire, April 2009. [http://www.businesswire.com/portal/site/home/permalink/?ndmViewId=multimedia_detail&eid=5929880&newsLang=en]

Tufte, Edward. *The Visual Display of Quantitative Information*. Cheshire, CT: Graphics Press, 1983.

Van Sickle, Jan. *Basic GIS Coordinates.* Boca Raton, FL: CRC Press, 2004.

Varadhan, G. and Manocha, D. *Out-of-Core Rendering of Massive Geometric Environments*. In Proc. of IEEE Visualization, 2002.

Ware, Michael. *Cyanotype: The History, Science and Art of Photographic Printing in Prussian.* London, England: Cromwell Press, 1999.

Wehowsky A. *Procedural Generation of a 3-D Model of the MIT Campus*. Independent Project Report, Massachusetts Institute of Technology, May 2001.

White, Seth J. *Pointer swizzling techniques for object-oriented database systems*. Ph.d. Thesis. University of Wisconsin, Madison, 1994.

Whiting, E. *Geometric, Topological, & Semantic Analysis of Multi-Building Floor Plan Data*. M.Sc. Thesis, Massachusetts Institute of Technology, June 2006.

Whiting E., Battat J., and Teller S. *Generating a Topological Model of Multi-Building Environments from Floor Plans*. In Proc. of Computer-Aided Architectural Design (CAAD) Futures, pages 115–128, July 2007.

Appendix

1 Software Build and Run-time Instructions

1.1 Dependencies

As discussed in Chapter 3, the Coin3D *DIME* DXF library is used as the core DXF parsing component. It can be checked-out of SVN using instructions available at http://www.coin3d.org/lib/dime/svn.

Furthermore, the *Xerces* XML library is used for reading and writing XML files. It can be downloaded and installed using instructions available at http://xerces.apache.org/xerces-c/.

Lastly, *Python* 2.5 is the interpreter environment used to run the Floor Inventory Manager. *Python* can be downloaded at http://www.python.org/download.

Otherwise, a standard *Linux* installation build (kernel 2.6) is used to compile and run the Floor Inventory Manager and the BMG DXF parsing engine.

1.2 Checkout and Build Instructions

The *Floor Inventory Manager* and the BMG DXF parser can both be checked out via this command:

```
$ svn co file:///afs/csail/group/rvsn/Repository/walkthru/mit/src/agents
```

The above command assumes the user has access to AFS space.

Once checked out, the *Floor Inventory Manager* does not require any compilation, as it is a Python script and can be run using a Python interpreter. As discussed in Chapter 3, there are two versions of the *Floor Inventory Manager*. One method uses web "scraping" to aggregate and download *Facilities* space inventories, and the other method accesses inventory information directly from a *Facilities* database. The "scraping" method is programmed into version 3 of the *Floor Inventory Manager*, whereas the database-driven method is programmed into version 5.

The DXF parser, on the other hand, is programmed using the C++ programming language and must be compiled in order to be executed. This can be accomplished using the "`setup.sh`" file contained in the parser's directory.

Before the setup script is run, it should be modified such that all dependency references (mentioned above) are accurate. For example, header file paths and linked library paths for the *DIME* library and *Xerces* should be updated to reflect the configuration of the machine at hand.

Once this step is complete, the parsing engine can be compiled simply by running:

```
$ ./setup.sh
```

1.3 Run-time Instructions

The first step necessary for running the pipeline to process the entire MIT campus is to download the relevant DXF files. This can be accomplished using the *Floor Inventory Manager*. As mentioned previously, there are two versions of the *Floor Inventory Manager*, one that "scrapes" information from the MIT *Department of Facilities* website, and another that downloads the necessary information from the *Department of Facilities* SQL database maintained by the MIT *Data Warehouse*.

The "scraping" method can be used by invoking version 3 of the *Floor Inventory Manager*:

```
$ python floorInventoryManager-v3.py
```

The database-driven method can be run by invoking version 5 of the *Floor Inventory Manager*:

```
$ python floorInventoryManager-v5.py
```

In each case, a summary file is produced. This file documents the buildings and floors that were discovered in the inventory reports, and the associated DXFs that were successfully downloaded (or not).

Appendix 2 includes one instance of this output file.

The next step in running the pipeline is to process each DXF file using the DXF parser. The best way to do this is to follow the example documented in "`dxf-xml-test`" file located in the parser directory. A script that sequentially processes each DXF file in a way similar to the "`dxf-xml-test`" can process 1000 floor plans easily in under an hour.

Note: the translation parameters described in Chapter 3 are found in the `transforms.xml` in the parser directory. This file of legacy translation data is automatically referred to at the relevant time during DXF processing.

Output naming conventions of the processing pipeline are determined at the command-line and can be specified by the user. As mentioned previously, the output generated is in the XML file format. However, the user can specify a path and filename for the output at the command-line. The generally accepted practice is to mirror the name of the DXF file, but to append it with an "`xml`" extension. For instance, if the DXF file were for 32-3 (building 32, 3rd floor), it would be named `32_3.dxf` and the associated XML file can be named `32_3.xml`.

2 Sample Output

2.1 Floor Inventory Manager Sample Summary File

```
Summary Report
Created on 2008-05-26 18:13:45.292588
-------------------
MIT Floor Inventory and DXF Acquisition:

        Building 1:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/1_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/1_1.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/1_2.dxf
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/1_3.dxf

        Building 10:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_1.dxf
                Floor 1M:               /afs/csail.mit.edu/u/y/yonib/dxfs/10_1M.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_2.dxf
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_3.dxf
                Floor 4:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_4.dxf
                Floor 4M:               /afs/csail.mit.edu/u/y/yonib/dxfs/10_4M.dxf
                Floor 5:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_5.dxf
                Floor 6:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_6.dxf
                Floor 7:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_7.dxf
                Floor 8:                /afs/csail.mit.edu/u/y/yonib/dxfs/10_8.dxf
                Floor 8M:               /afs/csail.mit.edu/u/y/yonib/dxfs/10_8M.dxf

        Building 11:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/11_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/11_1.dxf
                Floor 1M:               /afs/csail.mit.edu/u/y/yonib/dxfs/11_1M.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/11_2.dxf
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/11_3.dxf
                Floor 4:                /afs/csail.mit.edu/u/y/yonib/dxfs/11_4.dxf

        Building 12:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/12_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/12_1.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/12_2.dxf
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/12_3.dxf

        Building 12A:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/12A_0.dxf

        Building 13:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/13_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/13_1.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/13_2.dxf
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/13_3.dxf
                Floor 4:                /afs/csail.mit.edu/u/y/yonib/dxfs/13_4.dxf
                Floor 5:                /afs/csail.mit.edu/u/y/yonib/dxfs/13_5.dxf

        Building 14:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/14_0.dxf

        Building 14E:
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/14E_1.dxf
                Floor 1M:               /afs/csail.mit.edu/u/y/yonib/dxfs/14E_1M.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/14E_2.dxf
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/14E_3.dxf
                Floor 4:                /afs/csail.mit.edu/u/y/yonib/dxfs/14E_4.dxf
                Floor 5:                /afs/csail.mit.edu/u/y/yonib/dxfs/14E_5.dxf

        Building 14N:
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/14N_1.dxf
```

```
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/14N_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/14N_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/14N_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/14N_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/14N_5.dxf

Building 14S:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/14S_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/14S_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/14S_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/14S_2M.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/14S_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/14S_4.dxf

Building 14W:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/14W_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/14W_1M.dxf

Building 16:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_7.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/16_9.dxf
        Floor 9M:           /afs/csail.mit.edu/u/y/yonib/dxfs/16_9M.dxf

Building 17:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/17_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/17_1.dxf

Building 18:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/18_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/18_6.dxf

Building 2:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/2_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/2_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/2_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/2_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/2_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/2_5.dxf

Building 24:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/24_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/24_7.dxf

Building 26:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/26_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_5.dxf
```

```
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/26_7.dxf

Building 3:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/3_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/3_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/3_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/3_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/3_3.dxf
        Floor 3M:           /afs/csail.mit.edu/u/y/yonib/dxfs/3_3M.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/3_4.dxf
        Floor 4M:           /afs/csail.mit.edu/u/y/yonib/dxfs/3_4M.dxf

Building 31:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/31_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/31_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/31_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/31_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/31_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/31_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/31_3.dxf

Building 32:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/32_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/32_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/32_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/32_3.dxf
        Floor D4:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_D4.dxf
        Floor G4:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_G4.dxf
        Floor D5:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_D5.dxf
        Floor G5:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_G5.dxf
        Floor D6:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_D6.dxf
        Floor G6:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_G6.dxf
        Floor D7:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_D7.dxf
        Floor G7:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_G7.dxf
        Floor D8:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_D8.dxf
        Floor G8:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_G8.dxf
        Floor D9:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_D9.dxf
        Floor G9:           /afs/csail.mit.edu/u/y/yonib/dxfs/32_G9.dxf
        Floor D10:          /afs/csail.mit.edu/u/y/yonib/dxfs/32_D10.dxf
        Floor G10:          /afs/csail.mit.edu/u/y/yonib/dxfs/32_G10.dxf

Building 32P:
        Floor 000:          /afs/csail.mit.edu/u/y/yonib/dxfs/32P_000.dxf
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/32P_00.dxf

Building 33:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/33_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/33_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/33_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/33_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/33_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/33_5.dxf

Building 34:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/34_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/34_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/34_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/34_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/34_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/34_5.dxf

Building 35:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/35_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/35_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/35_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/35_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/35_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/35_5.dxf
        Floor 5M:           /afs/csail.mit.edu/u/y/yonib/dxfs/35_5M.dxf
```

```
Building 36:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/36_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/36_9.dxf

Building 37:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/37_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/37_7.dxf

Building 38:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/38_7.dxf

Building 39:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/39_7.dxf

Building 4:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/4_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/4_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/4_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/4_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/4_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/4_4.dxf
        Floor 4M:           /afs/csail.mit.edu/u/y/yonib/dxfs/4_4M.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/4_5.dxf

Building 41:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/41_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/41_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/41_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/41_3.dxf

Building 43:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/43_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/43_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/43_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/43_2M.dxf

Building 44:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/44_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/44_1.dxf

Building 46:
```

```
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/46_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_6.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/46_9.dxf

Building 48:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/48_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/48_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/48_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/48_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/48_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/48_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/48_5.dxf

Building 5:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/5_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/5_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/5_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/5_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/5_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/5_4.dxf

Building 50:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/50_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/50_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/50_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/50_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/50_4.dxf

Building 51:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/51_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/51_1.dxf

Building 54:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/54_9.dxf
        Floor 10:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_10.dxf
        Floor 11:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_11.dxf
        Floor 12:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_12.dxf
        Floor 13:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_13.dxf
        Floor 14:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_14.dxf
        Floor 15:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_15.dxf
        Floor 16:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_16.dxf
        Floor 17:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_17.dxf
        Floor 18:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_18.dxf
        Floor 19:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_19.dxf
        Floor 20:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_20.dxf
        Floor 21:           /afs/csail.mit.edu/u/y/yonib/dxfs/54_21.dxf

Building 56:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/56_00.dxf
        Floor 00M:          /afs/csail.mit.edu/u/y/yonib/dxfs/56_00M.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_2.dxf
```

```
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_7.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/56_9.dxf

Building 57:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/57_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/57_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/57_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/57_3.dxf

Building 6:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/6_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/6_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/6_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/6_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/6_2M.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/6_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/6_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/6_5.dxf

Building 62H:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/62H_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/62H_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/62H_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/62H_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/62H_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/62H_5.dxf

Building 62M:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/62M_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/62M_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/62M_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/62M_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/62M_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/62M_5.dxf

Building 62W:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/62W_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/62W_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/62W_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/62W_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/62W_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/62W_5.dxf

Building 64B:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/64B_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/64B_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/64B_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/64B_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/64B_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/64B_5.dxf

Building 64G:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/64G_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/64G_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/64G_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/64G_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/64G_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/64G_5.dxf

Building 64W:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/64W_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/64W_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/64W_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/64W_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/64W_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/64W_5.dxf
```

```
Building 66:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/66_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/66_6.dxf

Building 68:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/68_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/68_7.dxf
        Floor 7M:           /afs/csail.mit.edu/u/y/yonib/dxfs/68_7M.dxf

Building 6B:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/6B_1.dxf

Building 6C:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/6C_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/6C_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/6C_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/6C_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/6C_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/6C_5.dxf

Building 7:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/7_2M.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_3.dxf
        Floor 3M:           /afs/csail.mit.edu/u/y/yonib/dxfs/7_3M.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/7_6.dxf

Building 7A:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/7A_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/7A_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/7A_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/7A_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/7A_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/7A_6.dxf

Building 8:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/8_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/8_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/8_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/8_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/8_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/8_4.dxf

Building 9:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/9_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/9_6.dxf

Building E1:
```

```
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E1_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E1_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E1_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E1_3.dxf
        Floor 3M:           /afs/csail.mit.edu/u/y/yonib/dxfs/E1_3M.dxf

Building E15:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E15_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E15_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/E15_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E15_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E15_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E15_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E15_5.dxf

Building E17:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_5.dxf
        Floor 5M:           /afs/csail.mit.edu/u/y/yonib/dxfs/E17_5M.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E17_7.dxf
        Floor 7M:           /afs/csail.mit.edu/u/y/yonib/dxfs/E17_7M.dxf

Building E18:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_5.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E18_7.dxf

Building E19:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/E19_8.dxf

Building E2:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E2_6.dxf

Building E23:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E23_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E23_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E23_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E23_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E23_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E23_5.dxf

Building E25:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_6.dxf
```

```
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E25_7.dxf

Building E28:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E28_1.dxf

Building E33:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E33_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E33_2.dxf

Building E34:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E34_6.dxf

Building E38:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/E38_8.dxf

Building E39:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E39_0.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E39_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E39_3.dxf

Building E40:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E40_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E40_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E40_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E40_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E40_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E40_5.dxf

Building E48:
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E48_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E48_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E48_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E48_5.dxf

Building E51:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E51_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E51_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E51_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E51_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E51_4.dxf

Building E52:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E52_7.dxf
        Floor 7M:           /afs/csail.mit.edu/u/y/yonib/dxfs/E52_7M.dxf

Building E53:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E53_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/E53_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E53_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E53_2.dxf
```

```
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E53_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E53_4.dxf

Building E55:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/E55_9.dxf
        Floor 10:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_10.dxf
        Floor 11:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_11.dxf
        Floor 12:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_12.dxf
        Floor 13:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_13.dxf
        Floor 14:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_14.dxf
        Floor 15:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_15.dxf
        Floor 16:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_16.dxf
        Floor 17:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_17.dxf
        Floor 18:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_18.dxf
        Floor 19:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_19.dxf
        Floor 20:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_20.dxf
        Floor 21:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_21.dxf
        Floor 22:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_22.dxf
        Floor 23:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_23.dxf
        Floor 24:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_24.dxf
        Floor 25:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_25.dxf
        Floor 26:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_26.dxf
        Floor 27:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_27.dxf
        Floor 28:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_28.dxf
        Floor 29:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_29.dxf
        Floor 30:           /afs/csail.mit.edu/u/y/yonib/dxfs/E55_30.dxf

Building E60:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/E60_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/E60_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/E60_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/E60_3.dxf

Building E70:
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/E70_6.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/E70_8.dxf

Building N10:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N10_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N10_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N10_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/N10_2M.dxf

Building N16:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/N16_2M.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16_3.dxf

Building N16A:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16A_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16A_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16A_2.dxf

Building N16T:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N16T_1.dxf

Building N4:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_2.dxf
```

```
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/N4_8.dxf

Building N42:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N42_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N42_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N42_2.dxf

Building N51:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N51_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N51_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N51_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/N51_3.dxf
        Floor 3M:           /afs/csail.mit.edu/u/y/yonib/dxfs/N51_3M.dxf

Building N52:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N52_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N52_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N52_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/N52_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/N52_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/N52_5.dxf

Building N57:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/N57_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N57_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/N57_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/N57_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/N57_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/N57_5.dxf

Building N9:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/N9_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/N9_1M.dxf

Building NE108:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE108_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE108_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE108_3.dxf

Building NE125:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE125_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE125_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE125_3.dxf

Building NE18:
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE18_4.dxf

Building NE20:
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE20_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE20_4.dxf

Building NE25:
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE25_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE25_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE25_4.dxf

Building NE30:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_0.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_1M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_7.dxf
```

```
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30_8.dxf

Building NE30R:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE30R_1.dxf

Building NE47:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE47_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE47_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE47_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE47_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE47_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE47_5.dxf

Building NE48:
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE48_3.dxf

Building NE49:
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE49_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE49_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE49_4.dxf

Building NE80:
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/NE80_6.dxf

Building NW10:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW10_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW10_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW10_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW10_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW10_4.dxf

Building NW13:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW13_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW13_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW13_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW13_3.dxf

Building NW14:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_2M.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW14_6.dxf

Building NW15:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW15_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW15_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW15_1M.dxf

Building NW16:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW16_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW16_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW16_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW16_3.dxf

Building NW17:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW17_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW17_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW17_2.dxf

Building NW20:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW20_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW20_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW20_1M.dxf

Building NW21:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW21_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW21_1.dxf
```

```
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW21_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW21_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW21_2M.dxf

Building NW22:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW22_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW22_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW22_2.dxf

Building NW30:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW30_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW30_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW30_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW30_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW30_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW30_5.dxf

Building NW61:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW61_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW61_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW61_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW61_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW61_4.dxf

Building NW62:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW62_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW62_2.dxf
        Floor 2M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW62_2M.dxf

Building NW86:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_9.dxf
        Floor 10:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW86_10.dxf

Building NW86P:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW86P_0.dxf

Building NW95:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/NW95_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/NW95_1M.dxf

Building OC1:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/OC1_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC1_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC1_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC1_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC1_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC1_4.dxf

Building OC11:
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC11_4.dxf

Building OC19:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC19_1.dxf
        Floor 1M:           /afs/csail.mit.edu/u/y/yonib/dxfs/OC19_1M.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC19_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC19_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC19_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC19_5.dxf

Building OC19A:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/OC19A_1.dxf

Building OC19B:
```

```
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19B_1.dxf

Building OC19C:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19C_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19C_2.dxf

Building OC19D:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19D_1.dxf

Building OC19E:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19E_1.dxf

Building OC19F:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19F_1.dxf
        Floor 1M:             /afs/csail.mit.edu/u/y/yonib/dxfs/OC19F_1M.dxf

Building OC19G:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC19G_1.dxf

Building OC1A:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC1A_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC1A_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC1A_2.dxf

Building OC21:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC21_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC21_2.dxf

Building OC22:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC22_1.dxf

Building OC23:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC23_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC23_1.dxf

Building OC24:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC24_1.dxf

Building OC25:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC25_1.dxf

Building OC26:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC26_1.dxf

Building OC6:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/OC6_1.dxf

Building W1:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_3.dxf
        Floor 4:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_4.dxf
        Floor 5:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_5.dxf
        Floor 6:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_6.dxf
        Floor 7:              /afs/csail.mit.edu/u/y/yonib/dxfs/W1_7.dxf

Building W11:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W11_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W11_1.dxf

Building W13:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W13_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W13_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W13_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W13_3.dxf
        Floor 4:              /afs/csail.mit.edu/u/y/yonib/dxfs/W13_4.dxf

Building W15:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W15_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W15_1.dxf
```

```
        Floor 1M:             /afs/csail.mit.edu/u/y/yonib/dxfs/W15_1M.dxf

Building W16:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W16_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W16_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W16_2.dxf
        Floor 2M:             /afs/csail.mit.edu/u/y/yonib/dxfs/W16_2M.dxf

Building W2:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W2_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W2_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W2_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W2_3.dxf

Building W20:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_0.dxf
        Floor 0M:             /afs/csail.mit.edu/u/y/yonib/dxfs/W20_0M.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_3.dxf
        Floor 4:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_4.dxf
        Floor 5:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_5.dxf
        Floor 6:              /afs/csail.mit.edu/u/y/yonib/dxfs/W20_6.dxf
        Floor 6M:             /afs/csail.mit.edu/u/y/yonib/dxfs/W20_6M.dxf

Building W31:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W31_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W31_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W31_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W31_3.dxf

Building W32:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W32_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W32_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W32_2.dxf
        Floor 2M:             /afs/csail.mit.edu/u/y/yonib/dxfs/W32_2M.dxf

Building W33:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W33_0.dxf

Building W34:
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W34_1.dxf
        Floor 1M:             /afs/csail.mit.edu/u/y/yonib/dxfs/W34_1M.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W34_2.dxf

Building W35:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W35_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W35_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W35_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W35_3.dxf
        Floor 4:              /afs/csail.mit.edu/u/y/yonib/dxfs/W35_4.dxf
        Floor 5:              /afs/csail.mit.edu/u/y/yonib/dxfs/W35_5.dxf

Building W4:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_3.dxf
        Floor 4:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_4.dxf
        Floor 5:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_5.dxf
        Floor 6:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_6.dxf
        Floor 7:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_7.dxf
        Floor 8:              /afs/csail.mit.edu/u/y/yonib/dxfs/W4_8.dxf

Building W45:
        Floor 0:              /afs/csail.mit.edu/u/y/yonib/dxfs/W45_0.dxf
        Floor 1:              /afs/csail.mit.edu/u/y/yonib/dxfs/W45_1.dxf
        Floor 2:              /afs/csail.mit.edu/u/y/yonib/dxfs/W45_2.dxf
        Floor 3:              /afs/csail.mit.edu/u/y/yonib/dxfs/W45_3.dxf
        Floor 4:              /afs/csail.mit.edu/u/y/yonib/dxfs/W45_4.dxf
```

```
Building W5:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W5_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W5_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W5_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W5_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W5_4.dxf

Building W51:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/W51_6.dxf

Building W53:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W53_1.dxf

Building W53A:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W53A_1.dxf

Building W53B:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W53B_1.dxf

Building W59:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W59_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W59_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W59_2.dxf

Building W61:
        Floor 000:          /afs/csail.mit.edu/u/y/yonib/dxfs/W61_000.dxf
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/W61_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61_4.dxf
        Floor 16:           /afs/csail.mit.edu/u/y/yonib/dxfs/W61_16.dxf
        Floor 17:           /afs/csail.mit.edu/u/y/yonib/dxfs/W61_17.dxf
        Floor 18:           /afs/csail.mit.edu/u/y/yonib/dxfs/W61_18.dxf

Building W61A:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61A_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61A_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61A_3.dxf

Building W61B:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61B_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61B_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61B_3.dxf

Building W61C:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61C_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61C_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61C_3.dxf

Building W61D:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61D_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61D_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61D_3.dxf

Building W61E:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61E_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61E_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61E_3.dxf

Building W61F:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61F_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61F_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61F_3.dxf
```

```
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61F_4.dxf

Building W61G:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61G_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61G_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61G_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61G_4.dxf

Building W61H:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61H_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61H_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61H_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61H_4.dxf

Building W61J:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61J_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61J_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61J_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61J_4.dxf

Building W61M:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61M_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W61M_2.dxf

Building W7:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/W7_7.dxf

Building W70:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70_5.dxf

Building W70A:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70A_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70A_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70A_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70A_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70A_5.dxf

Building W70B:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70B_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70B_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70B_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70B_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70B_5.dxf

Building W70C:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70C_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70C_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70C_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70C_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70C_5.dxf

Building W70D:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70D_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70D_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70D_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70D_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70D_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70D_6.dxf

Building W70E:
```

```
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70E_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70E_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70E_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70E_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70E_5.dxf

Building W70F:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70F_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70F_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70F_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70F_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70F_5.dxf

Building W70G:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W70G_1.dxf

Building W71:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W71_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W71_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W71_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W71_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W71_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W71_5.dxf

Building W79:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/W79_9.dxf
        Floor 10:           /afs/csail.mit.edu/u/y/yonib/dxfs/W79_10.dxf

Building W8:
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W8_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W8_1.dxf

Building W84:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_0.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/W84_9.dxf
        Floor 10:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_10.dxf
        Floor 11:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_11.dxf
        Floor 12:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_12.dxf
        Floor 13:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_13.dxf
        Floor 14:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_14.dxf
        Floor 15:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_15.dxf
        Floor 16:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_16.dxf
        Floor 17:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_17.dxf
        Floor 18:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_18.dxf
        Floor 19:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_19.dxf
        Floor 20:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_20.dxf
        Floor 21:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_21.dxf
        Floor 22:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_22.dxf
        Floor 23:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_23.dxf
        Floor 24:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_24.dxf
        Floor 25:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_25.dxf
        Floor 26:           /afs/csail.mit.edu/u/y/yonib/dxfs/W84_26.dxf
```

```
Building W85:
        Floor 00:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_00.dxf
        Floor 0:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_0.dxf
        Floor 0M:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_0M.dxf
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_3.dxf
        Floor 4:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_4.dxf
        Floor 5:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_5.dxf
        Floor 6:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_6.dxf
        Floor 7:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_7.dxf
        Floor 8:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_8.dxf
        Floor 9:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85_9.dxf
        Floor 10:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_10.dxf
        Floor 11:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_11.dxf
        Floor 12:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_12.dxf
        Floor 13:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_13.dxf
        Floor 14:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_14.dxf
        Floor 15:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_15.dxf
        Floor 16:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_16.dxf
        Floor 17:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_17.dxf
        Floor 18:           /afs/csail.mit.edu/u/y/yonib/dxfs/W85_18.dxf

Building W85A:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85A_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85A_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85A_3.dxf

Building W85B:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85B_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85B_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85B_3.dxf

Building W85C:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85C_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85C_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85C_3.dxf

Building W85D:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85D_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85D_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85D_3.dxf

Building W85E:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85E_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85E_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85E_3.dxf

Building W85F:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85F_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85F_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85F_3.dxf

Building W85G:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85G_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85G_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85G_3.dxf

Building W85H:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85H_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85H_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85H_3.dxf

Building W85J:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85J_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85J_2.dxf
        Floor 3:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85J_3.dxf

Building W85K:
        Floor 1:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85K_1.dxf
        Floor 2:            /afs/csail.mit.edu/u/y/yonib/dxfs/W85K_2.dxf
```

```
                Floor 3:                /afs/csail.mit.edu/u/y/yonib/dxfs/W85K_3.dxf

        Building W91:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/W91_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/W91_1.dxf
                Floor 1M:               /afs/csail.mit.edu/u/y/yonib/dxfs/W91_1M.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/W91_2.dxf

        Building W92:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/W92_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/W92_1.dxf
                Floor 2:                /afs/csail.mit.edu/u/y/yonib/dxfs/W92_2.dxf

        Building WW15:
                Floor 0:                /afs/csail.mit.edu/u/y/yonib/dxfs/WW15_0.dxf
                Floor 1:                /afs/csail.mit.edu/u/y/yonib/dxfs/WW15_1.dxf

--------------------
Buildings and floors for which DXF files could not be retrieved:

        Building 62:
                (no floors found in the building 62 report)

        Building 64:
                (no floors found in the building 64 report)

        Building 10:
                Floor 3M DXF file not found.

        Building 11:
                Floor 2M DXF file not found.
                Floor 4M DXF file not found.

        Building 12:
                Floor 0M DXF file not found.

        Building 14E:
                Floor 2M DXF file not found.

        Building 16:
                Floor 8 DXF file not found.

        Building 32:
                Floor 0M DXF file not found.

        Building 4:
                Floor 0M DXF file not found.

        Building 42 missing *completely*:
                Floor 00 DXF file not found.
                Floor 0 DXF file not found.
                Floor 1 DXF file not found.
                Floor 2 DXF file not found.
                Floor 2M DXF file not found.

        Building 46:
                Floor 4M DXF file not found.
                Floor 5M DXF file not found.
                Floor 6M DXF file not found.
                Floor 7 DXF file not found.

        Building 56:
                Floor 8 DXF file not found.

        Building 68:
                Floor 00 DXF file not found.

        Building 7:
                Floor 1M DXF file not found.

        Building E15:
```

```
        Floor 2M DXF file not found.
        Floor 3M DXF file not found.
        Floor 4M DXF file not found.

Building E17:
        Floor 6 DXF file not found.

Building E18:
        Floor 6 DXF file not found.
        Floor 6M DXF file not found.

Building E19:
        Floor 2M DXF file not found.
        Floor 3M DXF file not found.
        Floor 4M DXF file not found.

Building E23:
        Floor 5M DXF file not found.

Building E25:
        Floor 2M DXF file not found.

Building E28:
        Floor 1M DXF file not found.

Building E38:
        Floor 1M DXF file not found.

Building E40:
        Floor 1M DXF file not found.

Building E48:
        Floor 3M DXF file not found.

Building E52:
        Floor 1M DXF file not found.

Building E70:
        Floor 8M DXF file not found.

Building NE25:
        Floor 2M DXF file not found.

Building NE47:
        Floor 3M DXF file not found.

Building NE49:
        Floor 2M DXF file not found.
        Floor 3M DXF file not found.
        Floor 4M DXF file not found.

Building NW12 missing *completely*:
        Floor 0 DXF file not found.
        Floor 1 DXF file not found.
        Floor 1M DXF file not found.
        Floor 2 DXF file not found.
        Floor 3 DXF file not found.
        Floor 4 DXF file not found.

Building OC25:
        Floor 1M DXF file not found.

Building W31:
        Floor 3M DXF file not found.

Building W35:
        Floor 2M DXF file not found.

Building W51:
        Floor 1M DXF file not found.
```

```
        Building W79:
                Floor 7M DXF file not found.

        Building W89 missing *completely*:
                Floor 1 DXF file not found.
                Floor 2 DXF file not found.
                Floor 2M DXF file not found.

        Building W92:
                Floor 1M DXF file not found.

        Building WW15:
                Floor 1M DXF file not found.
```

2.2 DXF Parser Sample XML Output – (`34-1.xml`)

```
<?xml version="1.0" ?>
<!DOCTYPE MITquest [

<!ATTLIST space name ID #IMPLIED>

]>
<MITquest>
    <floor name="FLOORCONTOUR">
          <contour>
              <centroid x="710149.49" y="496226.39"/>
              <extent maxx="710202.91" maxy="496280.30" minx="710108.64" miny="496177.19"/>
              <point x="710191.46" y="496211.77"/>
              <point x="710195.20" y="496210.37"/>
              <point x="710191.06" y="496199.27"/>
              <point x="710171.12" y="496190.18"/>
              <point x="710171.12" y="496190.18"/>
              <point x="710170.21" y="496189.77"/>
              <point x="710170.21" y="496189.77"/>
              <point x="710163.76" y="496186.83"/>
              <point x="710163.76" y="496186.83"/>
              <point x="710162.85" y="496186.42"/>
              <point x="710162.85" y="496186.42"/>
              <point x="710142.61" y="496177.19"/>
              <point x="710131.02" y="496181.52"/>
              <point x="710132.32" y="496184.73"/>
              <point x="710132.02" y="496184.84"/>
              <point x="710123.56" y="496187.85"/>
              <point x="710122.42" y="496184.80"/>
              <point x="710110.61" y="496189.22"/>
              <point x="710114.65" y="496199.32"/>
              <point x="710119.80" y="496197.27"/>
              <point x="710124.89" y="496210.01"/>
              <point x="710119.65" y="496212.28"/>
              <point x="710122.70" y="496220.25"/>
              <point x="710126.84" y="496231.17"/>
              <point x="710118.93" y="496234.08"/>
              <point x="710113.28" y="496231.50"/>
              <point x="710108.64" y="496241.66"/>
              <point x="710113.35" y="496243.80"/>
              <point x="710112.45" y="496245.78"/>
              <point x="710110.24" y="496250.63"/>
              <point x="710123.25" y="496276.52"/>
              <point x="710125.10" y="496280.30"/>
              <point x="710129.24" y="496279.25"/>
              <point x="710157.15" y="496272.47"/>
              <point x="710159.54" y="496267.24"/>
              <point x="710160.44" y="496265.26"/>
              <point x="710164.53" y="496267.13"/>
              <point x="710169.23" y="496256.82"/>
              <point x="710165.38" y="496255.06"/>
              <point x="710161.47" y="496246.41"/>
              <point x="710172.29" y="496242.48"/>
```

```
            <point x="710180.62" y="496239.37"/>
            <point x="710178.79" y="496234.32"/>
            <point x="710191.54" y="496229.56"/>
            <point x="710193.55" y="496234.59"/>
            <point x="710202.91" y="496231.18"/>
            <point x="710198.97" y="496220.41"/>
            <point x="710195.26" y="496221.80"/>
            <point x="710191.46" y="496211.77"/>
        </contour>
        <portal class="horizontal" source="true" target="34-100LA" type="explicit">
            <edge index="32" param="0.38"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="36" param="0.35"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="36" param="0.61"/></portal>
        <portal class="horizontal" source="true" target="34-100LA" type="explicit">
            <edge index="38" param="0.44"/></portal>
        <portal class="horizontal" source="true" target="34-100LA" type="explicit">
            <edge index="38" param="0.75"/></portal>
        <portal class="horizontal" source="true" target="34-100LA" type="explicit">
            <edge index="23" param="0.26"/></portal>
        <portal class="horizontal" source="true" target="34-100LA" type="explicit">
            <edge index="23" param="0.62"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="25" param="0.39"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="25" param="0.66"/></portal>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="42" param="0.14"/></portal>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="19" param="0.86"/></portal>
        <portal class="horizontal" source="true" target="34-100LA" type="explicit">
            <edge index="29" param="0.59"/></portal>
        <portal class="horizontal" target="34-100SC" type="implicit">
            <edge index="6" maxparam="-0.0" minparam="1.0"/></portal>
        <portal class="horizontal" target="34-100LA" type="implicit">
            <edge index="34" maxparam="0.0" minparam="1.23"/></portal>
</floor>
<space name="34-100LA" type="LOBBY">
      <contour>
            <centroid x="710136.38" y="496255.78"/>
            <extent maxx="710168.57" maxy="496280.30" minx="710109.31" miny="496231.63"/>
            <point x="710149.44" y="496242.58"/>
            <point x="710151.90" y="496241.59"/>
            <point x="710154.86" y="496248.87"/>
            <point x="710160.99" y="496246.58"/>
            <point x="710165.01" y="496255.44"/>
            <point x="710168.57" y="496257.06"/>
            <point x="710164.28" y="496266.47"/>
            <point x="710160.64" y="496264.81"/>
            <point x="710159.54" y="496267.24"/>
            <point x="710159.08" y="496267.03"/>
            <point x="710156.80" y="496272.04"/>
            <point x="710129.12" y="496278.77"/>
            <point x="710129.24" y="496279.25"/>
            <point x="710125.10" y="496280.30"/>
            <point x="710123.25" y="496276.52"/>
            <point x="710123.70" y="496276.30"/>
            <point x="710110.79" y="496250.62"/>
            <point x="710112.90" y="496245.98"/>
            <point x="710112.45" y="496245.78"/>
            <point x="710113.55" y="496243.35"/>
            <point x="710109.31" y="496241.41"/>
            <point x="710113.52" y="496232.16"/>
            <point x="710118.91" y="496234.62"/>
            <point x="710127.02" y="496231.63"/>
            <point x="710129.25" y="496237.60"/>
            <point x="710137.00" y="496234.71"/>
            <point x="710137.99" y="496237.36"/>
            <point x="710149.44" y="496242.58"/>
      </contour>
```

```
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="1" param="0.53"/></portal>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="1" param="0.21"/></portal>
        <portal class="horizontal" source="false" target="34-1FLOORCONTOUR" type="explicit">
            <edge index="10" param="0.61"/></portal>
        <portal class="horizontal" source="true" target="BMAP-4221" type="TRANSITION">
            <edge index="5" param="0.66"/></portal>
        <portal class="horizontal" source="true" target="BMAP-4221" type="TRANSITION">
            <edge index="5" param="0.37"/></portal>
        <portal class="horizontal" source="false" target="BMAP-4400" type="TRANSITION">
            <edge index="3" param="0.55"/></portal>
        <portal class="horizontal" source="false" target="BMAP-4400" type="TRANSITION">
            <edge index="3" param="0.24"/></portal>
        <portal class="horizontal" source="false" target="BMAP-4490" type="TRANSITION">
            <edge index="22" param="0.73"/></portal>
        <portal class="horizontal" source="false" target="BMAP-4490" type="TRANSITION">
            <edge index="22" param="0.39"/></portal>
        <portal class="horizontal" source="true" target="BMAP-4490" type="TRANSITION">
            <edge index="20" param="0.61"/></portal>
        <portal class="horizontal" source="true" target="BMAP-4490" type="TRANSITION">
            <edge index="20" param="0.32"/></portal>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="24" param="0.73"/></portal>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="24" param="0.43"/></portal>
        <portal class="horizontal" source="false" target="34-1FLOORCONTOUR" type="explicit">
            <edge index="15" param="0.40"/></portal>
        <portal class="horizontal" target="FLOORCONTOUR" type="implicit">
            <edge index="7" maxparam="0.18" minparam="1.0"/></portal>
  <triangle v0="0" v1="1" v2="2"/>
  <triangle v0="0" v1="2" v2="11"/>
  <triangle v0="0" v1="11" v2="26"/>
  <triangle v0="26" v1="11" v2="15"/>
  <triangle v0="26" v1="15" v2="24"/>
  <triangle v0="26" v1="24" v2="25"/>
  <triangle v0="24" v1="15" v2="16"/>
  <triangle v0="24" v1="16" v2="17"/>
  <triangle v0="24" v1="17" v2="19"/>
  <triangle v0="24" v1="19" v2="22"/>
  <triangle v0="24" v1="22" v2="23"/>
  <triangle v0="22" v1="19" v2="21"/>
  <triangle v0="21" v1="19" v2="20"/>
  <triangle v0="19" v1="17" v2="18"/>
  <triangle v0="15" v1="11" v2="13"/>
  <triangle v0="15" v1="13" v2="14"/>
  <triangle v0="13" v1="11" v2="12"/>
  <triangle v0="11" v1="2" v2="10"/>
  <triangle v0="10" v1="2" v2="9"/>
  <triangle v0="9" v1="2" v2="7"/>
  <triangle v0="9" v1="7" v2="8"/>
  <triangle v0="7" v1="2" v2="4"/>
  <triangle v0="7" v1="4" v2="6"/>
  <triangle v0="6" v1="4" v2="5"/>
  <triangle v0="4" v1="2" v2="3"/>
</space>
<space name="34-100SA" type="STAIR">
    <contour>
            <centroid x="710125.55" y="496215.03"/>
            <extent maxx="710129.56" maxy="496219.66" minx="710121.54" miny="496210.28"/>
            <point x="710121.54" y="496212.56"/>
            <point x="710126.79" y="496210.28"/>
            <point x="710129.56" y="496217.67"/>
            <point x="710124.26" y="496219.66"/>
            <point x="710121.54" y="496212.56"/>
    </contour>
    <portal class="horizontal" target="34-101" type="implicit">
        <edge index="1" maxparam="0.0" minparam="1.0"/></portal>
    <portal class="VERTICAL" direction="DOWN" target="34-000SB" type="STAIR"/>
  <triangle v0="0" v1="1" v2="2"/>
  <triangle v0="0" v1="2" v2="3"/>
```

```
    </space>
    <space name="34-100SB" type="STAIR">
      <contour>
              <centroid x="710174.45" y="496237.01"/>
              <extent maxx="710179.11" maxy="496240.92" minx="710169.71" miny="496233.11"/>
              <point x="710179.11" y="496238.15"/>
              <point x="710171.72" y="496240.92"/>
              <point x="710169.71" y="496235.88"/>
              <point x="710177.29" y="496233.11"/>
              <point x="710179.11" y="496238.15"/>
      </contour>
      <portal class="horizontal" target="34-101" type="implicit">
          <edge index="1" maxparam="-0.0" minparam="1.0"/></portal>
      <portal class="VERTICAL" direction="DOWN" target="34-000SA" type="STAIR"/>
    <triangle v0="0" v1="1" v2="3"/>
    <triangle v0="3" v1="1" v2="2"/>
</space>
<space name="34-100SC" type="STAIR">
      <contour>
              <centroid x="710165.64" y="496191.26"/>
              <extent maxx="710170.21" maxy="496195.68" minx="710161.07" miny="496186.83"/>
              <point x="710163.07" y="496188.35"/>
              <point x="710163.76" y="496186.83"/>
              <point x="710170.21" y="496189.77"/>
              <point x="710169.52" y="496191.29"/>
              <point x="710167.51" y="496195.68"/>
              <point x="710161.07" y="496192.75"/>
              <point x="710163.07" y="496188.35"/>
      </contour>
      <portal class="horizontal" target="FLOORCONTOUR" type="implicit">
          <edge index="1" maxparam="0.0" minparam="1.0"/></portal>
      <portal class="VERTICAL" direction="DOWN" target="34-000SC" type="STAIR"/>
    <triangle v0="0" v1="1" v2="3"/>
    <triangle v0="0" v1="3" v2="5"/>
    <triangle v0="5" v1="3" v2="4"/>
    <triangle v0="3" v1="1" v2="2"/>
</space>
<space name="34-101" type="LECT H">
          <contour>
              <centroid x="710155.33" y="496213.19"/>
              <extent maxx="710200.46" maxy="496247.15" minx="710112.92" miny="496183.56"/>
              <point x="710112.92" y="496190.49"/>
              <point x="710122.19" y="496187.02"/>
              <point x="710122.97" y="496189.12"/>
              <point x="710133.69" y="496185.30"/>
              <point x="710133.68" y="496185.30"/>
              <point x="710138.50" y="496183.56"/>
              <point x="710159.74" y="496193.24"/>
              <point x="710160.65" y="496193.66"/>
              <point x="710167.10" y="496196.59"/>
              <point x="710168.01" y="496197.01"/>
              <point x="710188.34" y="496206.28"/>
              <point x="710190.20" y="496211.23"/>
              <point x="710190.19" y="496211.24"/>
              <point x="710194.68" y="496223.08"/>
              <point x="710197.44" y="496222.05"/>
              <point x="710200.46" y="496230.30"/>
              <point x="710193.87" y="496232.70"/>
              <point x="710191.86" y="496227.66"/>
              <point x="710177.29" y="496233.11"/>
              <point x="710169.71" y="496235.88"/>
              <point x="710171.72" y="496240.92"/>
              <point x="710160.30" y="496245.06"/>
              <point x="710154.70" y="496247.15"/>
              <point x="710152.37" y="496241.41"/>
              <point x="710159.26" y="496238.72"/>
              <point x="710162.64" y="496237.51"/>
              <point x="710162.11" y="496236.08"/>
              <point x="710159.35" y="496236.29"/>
              <point x="710136.61" y="496225.92"/>
              <point x="710134.28" y="496223.40"/>
```

```
            <point x="710132.66" y="496223.94"/>
            <point x="710134.09" y="496227.43"/>
            <point x="710136.81" y="496234.25"/>
            <point x="710130.64" y="496236.55"/>
            <point x="710128.59" y="496231.06"/>
            <point x="710124.26" y="496219.66"/>
            <point x="710129.56" y="496217.67"/>
            <point x="710126.79" y="496210.28"/>
            <point x="710120.97" y="496195.72"/>
            <point x="710115.83" y="496197.77"/>
            <point x="710112.92" y="496190.49"/>
        </contour>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="22" param="0.32"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="22" param="0.73"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="32" param="0.32"/></portal>
        <portal class="horizontal" source="false" target="34-100LA" type="explicit">
            <edge index="32" param="0.70"/></portal>
        <portal class="horizontal" source="false" target="BMAP-4400" type="TRANSITION">
            <edge index="17" param="0.80"/></portal>
        <portal class="horizontal" source="false" target="BMAP-4378" type="TRANSITION">
            <edge index="37" param="0.18"/></portal>
        <portal class="horizontal" source="false" target="34-101E" type="explicit">
            <edge index="31" param="0.25"/></portal>
        <portal class="horizontal" source="false" target="34-101A" type="explicit">
            <edge index="23" param="0.77"/></portal>
        <portal class="horizontal" target="34-100SB" type="implicit">
            <edge index="19" maxparam="0.0" minparam="1.0"/></portal>
        <portal class="horizontal" target="34-100SA" type="implicit">
            <edge index="36" maxparam="-0.0" minparam="1.0"/></portal>
    <triangle v0="0" v1="1" v2="2"/>
    <triangle v0="0" v1="2" v2="38"/>
    <triangle v0="0" v1="38" v2="39"/>
    <triangle v0="38" v1="2" v2="3"/>
    <triangle v0="38" v1="3" v2="37"/>
    <triangle v0="37" v1="3" v2="5"/>
    <triangle v0="37" v1="5" v2="6"/>
    <triangle v0="37" v1="6" v2="36"/>
    <triangle v0="36" v1="6" v2="29"/>
    <triangle v0="36" v1="29" v2="30"/>
    <triangle v0="36" v1="30" v2="35"/>
    <triangle v0="35" v1="30" v2="34"/>
    <triangle v0="34" v1="30" v2="31"/>
    <triangle v0="34" v1="31" v2="32"/>
    <triangle v0="34" v1="32" v2="33"/>
    <triangle v0="29" v1="6" v2="28"/>
    <triangle v0="28" v1="6" v2="7"/>
    <triangle v0="28" v1="7" v2="8"/>
    <triangle v0="28" v1="8" v2="27"/>
    <triangle v0="27" v1="8" v2="9"/>
    <triangle v0="27" v1="9" v2="26"/>
    <triangle v0="26" v1="9" v2="19"/>
    <triangle v0="26" v1="19" v2="25"/>
    <triangle v0="25" v1="19" v2="20"/>
    <triangle v0="25" v1="20" v2="21"/>
    <triangle v0="25" v1="21" v2="24"/>
    <triangle v0="24" v1="21" v2="23"/>
    <triangle v0="23" v1="21" v2="22"/>
    <triangle v0="19" v1="9" v2="18"/>
    <triangle v0="18" v1="9" v2="12"/>
    <triangle v0="18" v1="12" v2="17"/>
    <triangle v0="17" v1="12" v2="13"/>
    <triangle v0="17" v1="13" v2="15"/>
    <triangle v0="17" v1="15" v2="16"/>
    <triangle v0="15" v1="13" v2="14"/>
    <triangle v0="12" v1="9" v2="10"/>
    <triangle v0="12" v1="10" v2="11"/>
    <triangle v0="5" v1="3" v2="4"/>
</space>
```

```
  <space name="34-101A" type="CLA SV">
        <contour>
              <centroid x="710149.10" y="496236.58"/>
              <extent maxx="710159.10" maxy="496242.03" minx="710140.74" miny="496230.02"/>
              <point x="710140.74" y="496238.06"/>
              <point x="710144.40" y="496230.02"/>
              <point x="710158.51" y="496236.45"/>
              <point x="710159.10" y="496238.24"/>
              <point x="710151.72" y="496241.13"/>
              <point x="710149.45" y="496242.03"/>
              <point x="710140.74" y="496238.06"/>
        </contour>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="3" param="0.21"/></portal>
    <triangle v0="0" v1="1" v2="5"/>
    <triangle v0="5" v1="1" v2="4"/>
    <triangle v0="4" v1="1" v2="2"/>
    <triangle v0="4" v1="2" v2="3"/>
  </space>
  <space name="34-101E" type="ELEC">
        <contour>
              <centroid x="710139.24" y="496231.54"/>
              <extent maxx="710144.02" maxy="496237.89" minx="710134.54" miny="496226.37"/>
              <point x="710138.38" y="496236.99"/>
              <point x="710137.46" y="496234.53"/>
              <point x="710134.54" y="496227.22"/>
              <point x="710136.39" y="496226.37"/>
              <point x="710144.02" y="496229.85"/>
              <point x="710140.36" y="496237.89"/>
              <point x="710138.38" y="496236.99"/>
        </contour>
        <portal class="horizontal" source="true" target="34-101" type="explicit">
            <edge index="1" param="0.76"/></portal>
    <triangle v0="0" v1="1" v2="5"/>
    <triangle v0="5" v1="1" v2="4"/>
    <triangle v0="4" v1="1" v2="3"/>
    <triangle v0="3" v1="1" v2="2"/>
  </space>

</MITquest>
```