# CASEBOOK

*An information retrieval system for housing floor plans*

B. SINAN INANC
*Design Knowledge Systems Research Center*
*TU Delft, Faculty of Architecture*
*Berlageweg 1, Delft, The Netherlands*
*s.inanc@bk.tudelft.nl*

**Abstract.** Floor plans are representations of choice for spatial information in architectural practice. They are expressive, readable, and familiar. My research examines possible uses of floor plan layouts in architectural information systems. Classification problems that arise are addressed by lazy computation. A prototype in the domain of residential units, CaseBook, has been developed and implemented. CaseBook uses graphical floor plans as core representations for storage, classification and retrieval. To reflect the plasticity of interpretations inherent to the complex and ill-defined architectural domain, the focus is on the flexibility of classification schemes. Flexibility is achieved through the application of adaptable automatic feature extraction and classification-on-demand by user-selected criteria. Queries can be graphically expressed in example layouts. The system ranks layouts according to their similarity to a query based on weighted nearest neighbor algorithm.

Graphical floor plans are expressive, easy-to-read and familiar representations for spatial information. This is one of the main reasons to use floor plan layouts to express the spatial characteristics of architectural artifacts in an information retrieval system. This research addresses problems related to the classification of spatial characteristics of architectural artifacts.

The complexity of architectural artifacts and the context dependency of their interpretation make it very unlikely that a generally agreed body of explicit domain knowledge will be developed. Spatial information objects are therefor subject to various interpretations. A lazy, post-coordinated classification approach is chosen to deal with this plasticity of interpretation.

CaseBook is a prototype for the storage and retrieval of floor plans of residential units. Current implementation contains a graphical case-editor for structured input of layouts. An automatic feature extraction (AFE) module extracts various topological and quantitative features from the layouts. Classification depends on similarity measurements according to user-defined criteria. Stored layouts can be retrieved through queries by examples (QBE).

Separation of the storage of raw information objects from their interpretation facilitates flexible classification. The interpretation of stored raw layouts can be adjusted at various levels of interactivity. End-users can influence AFE by adjusting topological transformation parameters and are able to classify cases by selecting and weighting the extracted features. AFE capabilities can be expanded or altered by plug-ins or by reprogramming.

The first CaseBook prototype is implemented in Object Pascal and tested on a hundred cases. A new prototype is now being developed to provide a better user-interface for the input of floor plans. It will also have extended and more robust AFE capabilities and additional modules for similarity visualization.

## 1. Context

Classification related problems need to be addressed by designing any information retrieval system. It is particularly difficult to classify spatial characteristics because of the lack of consensus on their interpretation. Domain complexity combined with context and perspective dependency of interpretations complicates the development of a generally accepted body of knowledge.

In order to develop an IR system in absence of sufficient canonical domain knowledge one can decide 1) to invest further in in-depth study into architectural thought, 2) to use a classification scheme based on personal insights, or 3) to implement a lazy approach that lets end-users carry out the classification tasks. The latter is preferred on the following considerations: 1) it is unlikely that a viable, generally accepted classification scheme is developed in the foreseeable future and 2) usability of fixed personal classification schemes are too restricted.

The following sections deal with the history and state-of-the-art of architectural archives, and the application of the proposed method.

### 1.1. ARCHITECTURAL CLASSIFICATION

As long as architectural catalogs will be compiled and archives built up, the problem of classification will remain a major issue. Vitruvius' architectural handbook Architectura, dating from the first century BC, introduced the concept of four orders to characterize styles of classical architecture by the type of column used: Doric, Ionic, Corinthian and Etruscan. Serlio (~1550) based the classification in his housing catalogue on the notions of location (city or countryside), the socio-economic status of inhabitants (from peasant to king) and character (French or Italian). The housing handbooks of Le Muet (1647) and Briseux (1743) preferred an order based on widths and depths of plots. Durand (1819) and de Quincy (1825) developed typologies of buildings, building elements and spaces. Downing (1850) defined American housing styles for catalogues. (Bakker and Rapp, 1998)

Research in the 20th century revealed a strong interest in using type as the main element for categorization. Classical categorization schemes for architectural spatial information have brought forth various architectural type systems (Klein, 1934; Sherwood, 1978; Neufert, 1980). Although their limited usability for IR-systems due to the ambiguity of interpretation, typological research has produced a welcome side effect in that it developed a rich vocabulary to describe spatial issues. From the 70s onwards, more unambiguous representations and type systems for small rectangular housing plans, offices and high-rise buildings have been developed (March, 1976; Habraken, 1976; Duffy et al. 1976; Mitchell, 1976; Bloch, 1979; Steadman, 1983; van Leusen, 1994). The relevance of certain spatial characteristics has been the subject of recent empirical studies (Hillier, 1987; Brown, 1987; Hanson, 1998).

Described conventional categorization schemes for spatial characteristics have been successfully applied to small collections for research or education. Retrieval was not the main issue due to the limited size of those collections. Categorization schemes became significant part of the exposed information as a device to reflect the collectors' viewpoints.

### 1.2. KEYWORD BARRIER

Retrieval capabilities are an important aspect in the development of large information systems. The setting up and maintaining of such systems is extremely resource-intensive. The operational value depends on a system's ability to make the available information accessible to as many users as possible. Access to information stored in large collections depends on the applied classification schemes. Like in many other real-world domains, this is the Achilles heel in architectural archives. Most of the present archives, including architectural ones, are founded on categorization schemes from conventional libraries or administrative systems. Their classification is based on attached abstract meta-data like keywords or quantitative data. Hidden assumption hereby is the ability to describe information objects sufficiently through a small set of highly abstracted meta-data.

Keywords from typological research are promoted as a dedicated vocabulary for architectural IR-systems (Leusen, 1994). The abstract iconic diagrams to visualize the type keywords are particularly appealing to the architectural world because of their descriptive, aesthetic and familiar characteristics. A type-based architectural IR-system will show the same characteristics as any other keyword-based system.

The consistency of keyword assignment by indexers cannot be guaranteed because of the complexity and vagueness of the information contents and the ambiguity of the representations used. The resulting classification will reflect the indexers' personal perspectives and vocabulary. Research from document retrieval systems shows that correspondence of judgement between various indexers, and between indexers and inquirers is rather an exception than rule (Swanson, 1988). Therefor, the retrieval effectiveness will suffer from problems related to the consistency of keyword assignment. This keyword problem is primarily a result of a linguistically atypical use of keywords as representations. New classification vocabularies or the training of indexers and inquirers will not solve this keyword barrier problem (Blair, 1990).

Different disciplines repeatedly put forward proposals in which they suggest to use post-coordinated, content-based retrieval strategies to tackle the problems (Akin, 1986; Blair, 1990; Fugmann, 1993). Post-coordinated retrieval strategies relies on the information content that is stored in a system. Successful examples of this approach are WWW search engines, (bio-) chemical archives, biometrical recognition, etc.

### 1.3. METHOD

Implementation of manually pre-coordinated classification schemes can be massively inefficient. It means that complete classifications have to be produced before any questions have been asked. A classification is useless when an inquirer poses questions that cannot be answered. On the other hand, if a classification scheme is worked out in detail it will probably contain many features nobody actually needs. In both cases, this is an impressive waste of work. How could classification systems be improved? By using a general and very powerful principle: Be lazy! - But lazy in a particular way. The lazy approach I use in this research can be characterized as follows;

1. try to store information objects 'as-is' without pre-processing,
2. defer classification tasks until asked and,
3. let the inquirer make the difficult classification decisions.

Expensive interpretation and classification of information by specialists can be limited to a minimum. Improvement of usability and reduction of lifetime costs becomes feasible because of the separation of stored information and its deferred processing.

Various problems have to be dealt with before implementing a lazy approach. Computational cost and scalability problems as result of on-demand processing must be addressed. Although classification-related decisions are left to the end-user, explicit domain knowledge is still required to develop AFE functionality. Representations suitable for raw storage have to be selected and accompanying feature extraction and comparison algorithms have to be developed. User-interfaces for classification and retrieval require special attention because of the unconventional approach and complexity of the domain.

## 2. Lazy Classification

Being lazy is usually not seen as a trait to be proud of. Anyone trying to explain that laziness is not only a very desirable virtue but also an essential aspect in the long-term success of classification systems is bound to meet with some resistance.

Lazy classification approach is based on demand-driven computation. Evading the responsibility for processing and classification decisions is an attractive alternative for painstaking but ungrateful pre-coordination efforts. Lazy classification systems imply less unnecessary work. Aha defines lazy systems as follows:

"We define purely lazy problem solvers to display the following three behaviors: (all beginning with "D")

1. Defer: They do not process their inputs (i.e. data) until given information requests.
2. Data-driven: They respond to requests by combining information from the stored data.
3. Discard: They dismiss any temporary intermediate results created during problem solving.

In contrast, eager algorithms compile their inputs into an intentional data structure (i.e., discarding their inputs), reply to information requests using this a priori compiled abstraction, and retain it for future requests." (Aha, 1997)

Similar approaches are known under synonyms like: analogy-, case-, precedent- or similarity-based reasoning. A common characteristic of these approaches is that it focuses on representation rather than on a combination of representation and processing. Lazy approaches do not only derive their intelligent behavior from their representation, but also from the flexible manner in which specific information storage is combined with answer queries. The terminology "lazy" serves to clarify this distinction by borrowing intuition from both common vernacular and, more importantly, previous scientific usage such as lazy evaluation in functional programming languages (e.g., deferred evaluation is used to process infinite data structures) (Aha, 1998).

Lazy approaches are often more appropriate for tasks that have a complex and/or dynamic solution space. For this, they need readily available cases rather than rules that are difficult to extract. This can significantly refocus knowledge acquisition efforts on how to structure cases.

Keywords, rules and other high-level abstractions can generally be used only for the purposes that guided their compilation. In contrast, the lazy approach under consideration here works with raw cases that can be used for several problem-solving purposes. By virtue of storing rather than discarding raw data, lazy approaches are suitable to generate characteristic precedent explanations in a demand-driven manner.

Unlike approaches that compile concise data abstractions, lazy approaches typically have low initial input and low classification cost. A disadvantage of lazy approaches is their need for more storage and processing resources to answer queries.
Examples of a successful application of lazy computation approaches are demonstrated by document retrieval, pattern recognition, and machine learning systems.

## 3. Implementation

Representation and processing related questions have to be addressed in order to perform lazy classifications. Representation for stored raw data must be 1) as complete as possible to reflect the complexity of real-world objects, 2) compact enough for efficient storage, and 3) suitable for AFE. Classification by lazy computation is not a radical solution for problems related to the availability of canonical domain knowledge. There is no context-free information, which means that a lazy approach too must be based on relevant architectural knowledge for every step in the development cycle. Nevertheless, a significant gain of implemented approach is the more atomistic and unambiguous character of the knowledge that is needed to extract single features and the flexibility to choose a classification viewpoint.

A prototype, CaseBook, is developed as a lazy classification based IR system. Major issues related to the implementation of CaseBook will be discussed in the following sections.

### 3.1. FLOOR PLAN LAYOUT

As has already been stated, CaseBook uses graphical layouts as core representations. Floor plan layouts are available from a variety of sources (e.g. books, magazines and company records), but unfortunately, they are not always suitable for direct processing in an automated system. CaseBook is based on a structured graphical format for the storage and processing of floor plans. A case editor has been developed for the graphical input of metric floor plan layouts.

It is also possible to attach non-graphical structured information (e.g. occurrence, site information and architects' names) and additional unstructured multi-media data. Only the structured information can be used for classification, but unstructured data can provide valuable contextual information to end-users.

Since the focus of this research is on spatial/functional characteristics, I have ignored features such as precise measurements, building materials and finishes. Their exclusion from raw case descriptions restricts classification flexibility, but inclusion means a disproportionate increase in input cost. Current level of detail is chosen for inputting a floor plan in two to five minutes from regular sources such as magazines and books.

A different level of detail may be a more attractive option in another context, e.g. the development of a corporate archive for which completeness of a few case descriptions is more important than input cost and availability of information.

### 3.2. AUTOMATIC FEATURE EXTRACTION

The present feature set has been created for pragmatic reasons, i.e. to demonstrate the feasibility of the proposed IR-system. Many qualitative features found in actual practice are difficult to extract and to compare. E.g., an algorithm for the extraction and comparison of floor shape characteristics is still under development. More research is necessary to identify distinctive features and develop related extraction algorithms.

Creating a flexible IR-system that meets end-users' unpredictable needs means that the classification scheme has to contain as many features as possible. In practice, the number of features will be restricted by finite computational resources, inadequacy of representations and the extent to which explicit domain knowledge is available. The current implementation of CaseBook lets the inquirer build own classification scheme by choosing from 280 features for each residential unit.

TABLE 1. Summary of available features.

| Features | | Source | Availability as criterion | |
|---|---|---|---|---|
| Topology | Adjacency relations | Extracted | - | Not implemented yet |
| | Access relations | Extracted | - | Used for circulation cost |
| | Vertical organization | Extracted | + | 12 features |
| Sight | Sight relations | Extracted | - | Not implemented yet |
| | Sight to outside | Extracted | + | 18 features |
| | Orientations | Extracted | + | 3 features |
| Circulation | Circulation costs | Calculated | + | 132 features |
| Structure | Bearing distances | Extracted | + | 6 features |
| | Principles | Attached | + | 4 features |
| Quantities | Dimensions | Extracted | + | 16 features |
| | Areas | Extracted | + | 42 features |
| | Area ratios | Calculated | + | 4 features |
| | Quantities | Extracted | + | 18 features |
| Context | Designer, date… | Attached | - | 6 features |
| | Access | Attached | + | 5 features |
| | Repetition | Attached | + | 12 features |
| | Occurrence | Attached | + | 8 features |

### 3.3. CLASSIFICATION

Classification depends on similarity. To compute similarities, the features to be taken into account have to be specified first. In contrast to classification tasks from other research domains (e.g. object recognition, pattern matching, machine learning) a fixed or automated feature selection is an undesirable option for an architectural IR-system. Classification must reflect an individual user's perspectives; this means that the ungrateful task of feature selection can be left to the end-users by putting further emphasis on the laziness of the approach.

Floor plans can be classified by making every possible combination available features. An inquirer may select the relevant features as criteria and adjust their relative importance with the criteria editor.

Similarity assessment in the present implementation relies on a geometric model based on nearest neighborhood (NN-)algorithm. Geometric model is preferred because of its high-resolution and low computational cost.

### 3.4. QUERY BY EXAMPLE AND RETRIEVAL

Large spatial datasets, especially geographic ones, are being put on the market. Due to the impracticability of interaction through current commercial available query languages (SQL, Quel…), various experimental spatial query languages are developed: Atlas (Tsurutani, et al. 1980), PSQL (Roussopoulos, et al. 1988), SQL-Extensions (Herring, et al. 1988)… Lexical interaction methods of these query languages for spatial datasets are primarily a-spatial. It will be a major step towards the successful utilization of spatial information systems if users are allowed to draw a picture of the image they have in their mind in order to retrieve the spatial data of interest rather than forcing the inquirer to express a spatial configuration in some formal or natural language, (Egenhover, 1996). Some examples of experimental drawing- or sketch-based spatial information interfaces are: QBIC for image retrieval [wwwqbic.almaden.ibm.com], Cigales for GIS (Calcinelli, 1994), SQbS for GIS (Egenhover, 1996), Sketcho! for GIS (Blaser, 1999) and Electronic Napkin for diagrams (Gross, 1996).

As usual in architectural discipline, the inquirer interacts with CaseBook through drawings. Using example drawings instead of a lexical query language will greatly improve user interaction. It is easy for an inquirer to use the case-editor to draw example layouts to express the spatial configuration of interest. Built-in AFE capabilities make it possible to extract features from an inquirer's drawing in just the same way features are extracted from stored layouts. Queries may rely on complete layouts, incomplete sketches or simply a layout taken from an archive.

The retrieval of floor plans is based on similarity indexes. End-user can point a prior layout or a query drawing to examine its level of similarity to other floor plans according to used criteria set.

The application of multiple criteria will seldom produce exact matches, but will show subtle gradations of similarity. This is why retrieval results are presented as similarity rankings instead of a selection of best-matched cases. Similarity rankings are presented as sorted lists or as non-linear mapping of similarity clusters. Similarity assessments can be examined by similarity explanation reports.

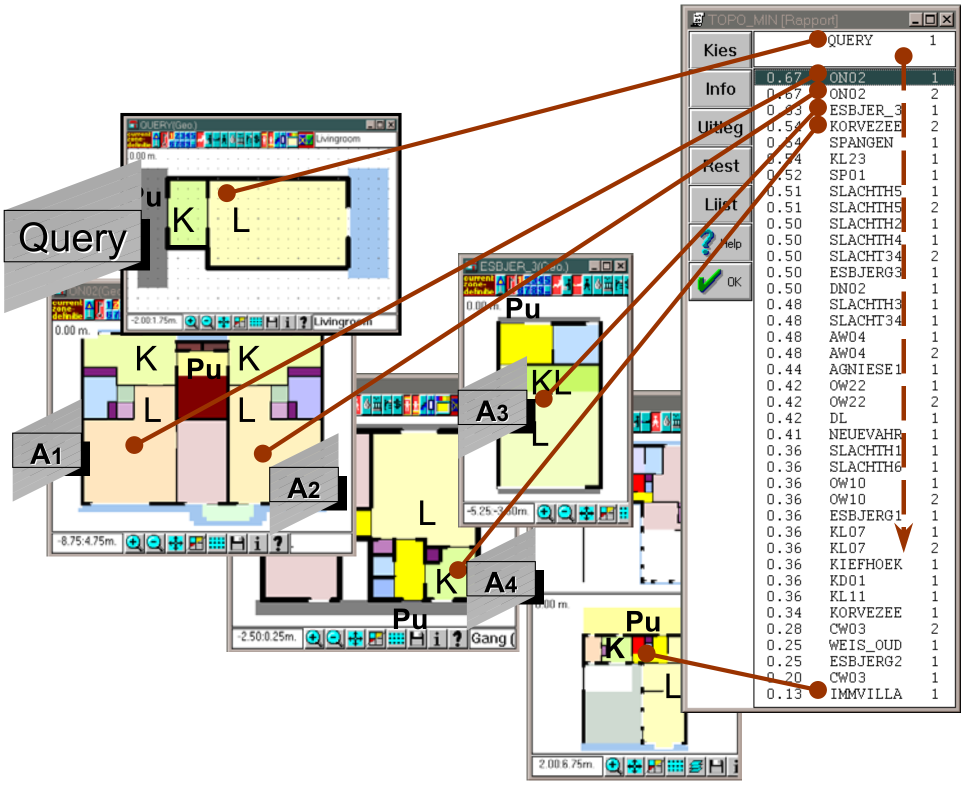


*Figure 1*. Query by example drawing (left top) and similarity ranking (right).

## 4. Discussion

The research towards an information system that can deal with spatial characteristics of architectural artifacts has led to a number of major issues that need further attention.

The cost of laziness could be high. On the one hand, deferring difficult classification decisions could mean saving on specialist manual input cost. But, on the other hand raw information instead of dense meta-data implies a higher impact on storage resources, and classification on demand requires additional processing power during user interaction. Despite of ever-growing computational resources, this persistent problem needs to be addressed.

Compared with conventional IR-systems CaseBook improves information classification:

- AFE reduce data input cost and, produce rich and unambiguous feature sets.
- Storage of raw floor plans allows perceptual restructuring.
- Similarity assessment by NN-algorithms improves classification accuracy.
- Complex spatial requests can be interactively expressed through QBE.

Laziness is not a discrete state. The implementation of a pure lazy system will not be achievable, but the performance of conventional ‘eager’ systems can be improved through additional lazy tactics.

## References

Aamodt, A. and Plaza, E.: 1994, *Case-based reasoning: Foundational issues, methodological variations, and system approaches,* AI Communications, 7, 39-59.

Aha, D. W. (ed.): 1997, *Lazy learning,* Kluwer, Norwell, MA.

Aha, D. W.: 1998, *The omnipresence of case-based reasoning in science and application, Report,* http://www.aic.nrl.navy.mil/~aha.

Akin, Ö.: 1986, *Psychology of architectural Design,* Pion, London.

Bakker, D. and Rapp, C.: 1998, *Het kant en klaar huis,* NAI Publishers, Rotterdam.

Blair, D. C.: 1990, *Language and representation in information retrieval,* Elsevier, Amsterdam.

Blaser, A.: 1999, *Prototype application Sketcho!,* Technical Report TR99-1, University of Maine, NCGIA, Maine.

Bloch, C. J.: 1979, *Catalogue of small rectangular plans,* Environment and planning B: Planning and Design 6, 155-190.

Calcinelli, D. and Mainguenaud, M.: 1994, *Cigales, a visual query language for a geographical information system: The user-interface,* Journal of Visual Languages and Computing, 5, 113-132.

Duffy, F., Cave, C. and Worthington, J.: 1976, *Planning office space,* Architectural Press, London.

Egenhofer, M. J.: 1996, Spatial-Query-by-Sketch, *in* M. Burnett and W. Citrin (eds.), *VL’96: IEEE Symposium on Visual Languages*, Boulder, CO, IEEE Press, pp. 60-77.

Fugmann, R.: 1993, *Subject analysis and indexing. Theoretical foundation and practical advise,* Index Verlag, Frankfurt/Main.

Habraken, N. J., Boekholt, J. T. and Dinjens, P. J. M.: 1976, *Variations; the systematic design of supports,* MIT Press, Cambridge, Mass.

Hanson, J.: 1998, *Decoding homes and houses,* Cambridge University Press, Cambridge, UK.

Herskovits, A.: 1998, Schematization, *in,* P. Olivier and, K. Gapp (eds.), *Representation and Processing of Spatial Expressions,* Lawrence Erlbaum Associates, New Jersey.

Hillier, B., Hanson, J. and Graham, H.: 1987, *The ideas are in things: an application of the space syntax method to discovering house genotypes,* Environment and planning B: Planning and Design, 14, 363-385.

Kak, A.: 1988, *Spatial reasoning,* AI Magazine, 9(2).

Klein, A.: 1934, *Das einfamilienhaus sudtyp,* Hoffmann, Stuttgart.

Kolodner, J. L.: 1993, *Case-based reasoning,* Morgan Kaufmann, San Mateo.

Leusen, M. van: 1994, *A system of types in the domain of residential buildings,* Publicatieburo Bouwkunde, Delft.

March, L.: 1976, A Boolean description of a class of built forms, *in,* L. March (ed.), *The architecture of form,* Cambridge University Press, Cambridge, UK, pp. 41-73.

Neufert, E.: 1980, *Architect’s data,* Granada Publishing, London.

Roussopoulos, N., Faloutsos, C. and Sellis, T.: 1988, *An efficient pictorial database system for PSQL,* IEEE Transactions on Software Engineering 14(5), 630-638.

Sherwood, R.: 1978, *Modern housing prototypes,* Harvard University Press, Cambridge Mass.

Swanson, D. R.: 1988, *Historical note: Information retrieval and the future of an illusion,* Journal of the American Society for Information Science, 39.

Tsurutani, T., Kasakawa, Y. and Naniwada, N.: 1980, *Atlas: a geographic database system—data structure and language design for geographic information,* ACM Computer Graphics, 14 (3), 71-77.