




Homemaker: software for adaptive domestic design

Bruno Postle ¹

¹ Independent Researcher, UK

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Arfon Smith](#) 

Reviewers:

- [@JJ](#)
- [@abhishektiwari](#)
- [@Jesusbill](#)

Submitted: 28 March 2024

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

The Homemaker stack of software evolves domestic buildings using a modified version of Christopher Alexander's Pattern Language as fitness criteria. Multiple buildings can be evolved in parallel, buildings with multiple-storeys, non-orthogonal plots, and courtyard layouts are supported. The software shows how evolutionary computation provides a design method complementary to the theory of Pattern Languages, thereby addressing a major criticism of Pattern Languages that they do not offer a practical design method.

Statement of need

Christopher Alexander's work, in particular 'A Pattern Language' ([Alexander et al., 1977](#)), is a description of a saner, safer, more humane built environment, that has had profound influence outside the world of architecture in the form of Software Design Patterns ([Postle, 2019](#)). 'A Pattern Language' is a best-seller in architecture, with many advocates, but very few people appear to actually be using it to design buildings. Pattern Languages for buildings are a failure, even Alexander said it was "my biggest failure" ([Hopkins, 2010](#)). The hypothesis behind this software is that the theory behind Pattern Languages is sound, but that there is something missing in terms of a design method, which Alexander identified as iteration, adaptation through lots of small steps, or unfolding, but didn't offer any practical advice beyond that, see 'The Timeless Way of Building' ([Alexander, 1979](#)). Pattern Languages are a useful technology, and the success of Design Patterns in the field of software is evidence of this, but a practical means of implementing A Pattern Language for buildings is elusive. Homemaker generates 3D BIM (Building Information Models) in validated IFC format, ie. it *can* be used in a workflow to design actual buildings, however the research purpose is to provide a framework where the Pattern Language can be judged as a design theory on its own merits - on the basis that a theory of architecture should be sufficient to actually design buildings that people would want to live in, and that meet human needs. There are apparently no other tools with a similar aim or scope - 'AI' architectural design is a growing field ([Caetano et al., 2020](#)), but the implicit aim of the discipline is improved efficiency of commercial property development.

Homemaker was designed for researchers to use to produce building designs, but with the ability to tweak individual patterns and observe the effects.

Brief description

The Homemaker system ([Postle, 2013](#)) consists of a collection of software components intended to be run unattended. The system evolves multi-storey building designs using a Pattern Language ([Alexander et al., 1977](#)) as fitness criteria for selection in a Genetic Algorithm.

The presumed construction process of a historic vernacular building is that it starts with a single room and grows by accretion and subdivision, with periodic removal of failed and unwanted parts of the structure. To simulate this, a 'load-bearing wall' form-language is used: buildings

are nested agglomerations of rooms; each room is a four-sided quadrilateral (not necessarily orthogonal); walls are vertical; the geometry of storeys below informs the geometry of storeys above. A novel binary tree representation, with each division holding orientation and ratio numbers indicating a geometric partition of a 'room' into two smaller 'rooms', can be used to describe such a building. The graph/tree is recursive, so any subdivision of rooms and storeys can be represented. Each leaf-node in the tree is a room with an interchangeable 'usage' (kitchen, bedroom etc..) that can be explored by optimisation. Typical historic vernacular buildings closely fit this model and can usually be represented with such a binary tree.

A specific advantage of a binary tree data structure is that branches can be 'grafted' from one model to another, an analogue of crossover or recombination in sexual reproduction of biological systems - this tree grafting is a standard technique in Genetic Algorithms. The fitness function consists of selected patterns from the Pattern Language assessing the model in the context of its local environment (the daylight field formed by surrounding buildings). The fitness calculation ultimately maximises a single value derived from this 'quality', multiplied by floor area, and divided by a simplified cost function representing a physical quantity of building materials. A population of models, each a variation of the same building, is produced through mutation, crossover and culling. When multiple buildings are evolved in parallel, the daylight field environment is periodically updated based on the geometry of the buildings as they change.

The software stack consists of: [Homemaker](#), a queue manager for evolving multiple buildings in parallel; [Urb](#), the data model for a single building; and [Molior](#), [File::IFC](#) and [File::DXF](#), which are used to generate a 3D BIM model on completion of the evolution.

Updates

Recent updates to the software include:

- 'Sahn' Space Type, a courtyard circulation space prevalent in traditional Arab houses ([Hakim, 2013](#))
- A configuration system allows users to tailor pattern fitness parameters and building costs for individual and groups of buildings.
- The Algorithm::Evolutionary ([Merelo Guervós et al., 2010](#)) Perl module replaces the previous Makefile-based system for driving evolution in populations.
- The Homemaker queue manager allocates resources for the parallel evolution of multiple buildings.
- Space centrality is calculated using the average path length over the circulation graph to all other spaces.

Challenges

This software is fundamentally non-interactive. User control is limited to the setting initial parameters, though the resulting IFC BIM model is readily editable in Native IFC software such as BlenderBIM ([Moult, 2025](#)). Homemaker uses a binary tree to describe the building, but this is conceptually hard for a human to manipulate directly as it has no visual relation to a building design. The data structure is instructions to build (a genotype), rather than the finished geometry (a phenotype). A separate but related project not described here is the 'Homemaker add-on' ([Postle, 2023](#)) that provides an entirely interactive experience based on the same principles, using non-manifold spatial geometry and the Topologic library ([Jabi & Chatzivasileiadi, 2021](#)).

Evolutionary design in this manner seems to be unsuitable for placing smaller buildings on larger plots, or larger ‘towers in a park’ designs, the constraint of occupying the entire space appears to be necessary. For a more suburban typology, pre-prepared standard designs distributed in a ‘cookie-cutter’ manner would likely achieve the desired result more efficiently. Homemaker is also relentlessly domestic, producing good ordinary buildings, these buildings don’t show the hand of an architect, and as such may not be perceived as Architecture.

Comparison with Other Generative Design Approaches

Commercial generative design tools (e.g., Archistar (Archistar, 2025), Spacemaker AI (Autodesk Spacemaker, 2025), TestFit (TestFit Inc., 2025)) primarily optimise for financial metrics like return on investment and regulatory compliance. Based on publicly available documentation, these tools evaluate design success through economic feasibility rather than optimising for human-centred design criteria. Homemaker also differs from notable academic approaches:

Shape Grammar systems (Duarte, 2005; Stiny & Gips, 1972) focus on formal composition rules rather than the human experiential qualities central to Pattern Language. Space Syntax methodologies (Hillier & Hanson, 1984) operate predominantly at the urban scale, analysing city-wide movement patterns and visibility networks. While Homemaker does analyse spatial connectivity, it operates at the building scale and is underpinned by an evolvable binary tree model structure particularly suited to evolutionary processes. Urban procedural tools like CityEngine (CGA Shape Grammar) (Müller et al., 2006) and DeCodingSpaces (Koenig et al., 2018) focus primarily on street networks and block configurations rather than building-level design patterns.

Homemaker’s distinctive contribution is its use of evolutionary computation with Pattern Language as fitness criteria. This computational approach makes Alexander’s influential but unwieldy theory implementable for actual building design, addressing qualitative aspects of human experience. The binary tree representation of building layouts enables evolutionary processes while preserving the fundamental structure of load-bearing wall architecture. This represents an approach distinct from approaches that prioritise financial optimisation or formal composition.

Acknowledgements

Christopher Alexander (1936-2022) was the inspiration for this work.

References

- Alexander, C. (1979). *The timeless way of building*. Oxford University Press.
- Alexander, C., Ishikawa, S., & Silverstein, M. (1977). *A pattern language: Towns, buildings, construction*. Oxford University Press. ISBN: 9780195019193
- Archistar. (2025). *Archistar platform*. <https://www.archistar.ai>. <https://www.archistar.ai>
- Autodesk Spacemaker. (2025). *Spacemaker: Site planning and analysis*. <https://www.autodesk.com/content/autodesk/global/en/products/spacemaker/overview.html>. <https://www.autodesk.com/content/autodesk/global/en/products/spacemaker/overview.html>
- Caetano, I., Santos, L., & Leitão, A. (2020). Computational design in architecture: Defining parametric, generative, and algorithmic design. *Frontiers of Architectural Research*, 9(2), 287–300. <https://doi.org/10.1016/j.foar.2019.12.008>
- Duarte, J. P. (2005). Towards the mass customization of housing: The grammar of siza’s houses at malagueira. *Environment and Planning B: Planning and Design*, 32(3), 347–380.

- 127 Hakim, B. S. (2013). *Arabic-islamic cities building and planning principles*. London Routledge
128 2013. ISBN: 9780710300942
- 129 Hillier, B., & Hanson, J. (1984). *The social logic of space*. Cambridge University Press.
- 130 Hopkins, R. (2010, December 23). *An interview with christopher alexander*. *Transition culture*.
131 <https://www.transitionculture.org/2010/12/23/exclusive-to-transition-culture-an-interview-with-chri>
- 132 Jabi, W., & Chatzivasileiadi, A. (2021). Topologic: Exploring spatial reasoning through
133 geometry, topology, and semantics. In S. Eloy, D. Leite Viana, F. Morais, & J. Vieira Vaz
134 (Eds.), *Formal methods in architecture* (pp. 277–285). Springer International Publishing.
135 https://doi.org/10.1007/978-3-030-57509-0_25
- 136 Koenig, R., Beilik, M., Knecht, K., Abdulmawla, A., & Fuchkina, E. (2018). *New methods for*
137 *urban analysis and simulation with grasshopper - using DeCodingSpaces-toolbox*. 65–68.
138 <https://doi.org/10.52842/conf.ecaade.2018.1.065>
- 139 Merelo Guervós, J. J., Castillo, P. A., & Alba, E. (2010). Algorithm::evolutionary, a flexible
140 perl module for evolutionary computation. *Soft Computing*, 14(10), 1091–1109. <https://doi.org/10.1007/s00500-009-0504-3>
- 141
- 142 Moul, D. (2025). *Bonsai BIM add-on* (Version v0.8.2). <https://bonsaibim.org/>
- 143 Müller, P., Wonka, P., Haegler, S., Ulmer, A., & Van Gool, L. (2006). Procedural modeling of
144 buildings. *ACM SIGGRAPH 2006 Papers*, 614–623.
- 145 Postle, B. (2013). An adaptive approach to domestic design. *Journal of Biourbanism*, 11
146 (1&2/2013), 31. <https://doi.org/10.6084/M9.FIGSHARE.6267401.V2>
- 147 Postle, B. (2019). On pattern languages, design patterns and evolution. *New Design Ideas*, 3(1),
148 44–52. <http://jomardpublishing.com/UploadFiles/Files/journals/NDI/V3N1/PostleB.pdf>
- 149 Postle, B. (2023). *Homemaker add-on* (Version 2023-12-16). <https://github.com/brunopostle/homemaker-addon>
- 150
- 151 Stiny, G., & Gips, J. (1972). Shape grammars and the generative specification of painting and
152 sculpture. *Information Processing*, 71, 1460–1465.
- 153 TestFit Inc. (2025). *TestFit: Site feasibility and building design software*. <https://testfit.io>.
154 <https://testfit.io>