Industrialized Building Renovation: Manufacturing Through a Constraint-Based On-line Support System

Andres Felipe Barco Santa, Élise Vareilles, Paul Gaborit, Jean-Guillaume Fages, Michel Aldanondo

Industrialized Building Renovation: Manufaturing Through a Constraint-based On-line Support System

A. F. Barco[1], E. Vareilles[1], P. Gaborit[1], J. G. Fages[2], M. Aldanondo[1]
[1]Université de Toulouse-Mines d'Albi, Albi, France
[2]COSLING S.A.S., Nantes, France
(corresponding author email: abarcosa@mines-albi.fr)

Abstract - **We present the coupling of two constraint-based environments into an on-line support system for facade-layout configuration in the context of building renovation. The configuration consist on the definition and allocation of a set of rectangular parameterizable panels over a facade surface. The coupling allows to solve two configuration tasks: To configure a set of questions relating the renovation model needed to determine limits for panels' size and panels' weight and to configure a constraint satisfaction model for each of the facades to renovate. Two constraint-based systems handle the filtering of incompatible values and the generation of layout plans in a web-service setup. The first service performs initial filtering to set panels' limits, based on the questionnaire, using a constraint filtering engine called Cofiade. The second service uses several facade-layout configuration algorithms, using as underlying engine the constraint solver Choco, to generate compliant layout-plan solutions. We show that by dividing filtering and search, and by coupling the two constraint-based systems, we gain modularity and efficiently as each service focuses on their own strengths.**

Keywords – **Building renovation, manufacturing, layout plan, support system, constraint satisfaction problems**

I. INTRODUCTION

Product configuration refers to the task of build a target product using predefined components, respecting requirements from customers and following some rules that shape a correct configuration [1]. This task have been increasingly supported by intelligent systems given the complexity and size of relations within a single product.

A particular scenario on product configuration arises from the context of building thermal renovation as an effort to reduce current energetic consumption levels. Here, the problem lies on the configuration of rectangular parameterizable panels, and their attaching devices called fasteners, that must be allocated over the facade surface in order to provide an insulation envelope [2]. A configuration solution is a plan which satisfies optimization criteria and a set of constraints (such as geometrical, weight or resources constraints) provided by users and the facade itself. As part of the product configuration family problems, an instance of facade-layout configuration problem has a huge search space that depends on the size of the panels and the elements on the facade, such as windows, doors and supporting areas. In consequence, to solve this configuration problem, we use a technique from artificial intelligence and operation research called constraint satisfaction problems [3].

Constraint satisfaction problems (CSPs) are conceived to allow the end-user to state the logic of the computation rather than its flow. For example, in the context of scheduling, instead stating a set of steps to avoid tasks overlapping, the user declares *"for any pair of tasks they must not overlap"*. The user may do so by stating a) variables representing elements of the problem, b) a set of potential values associated to each variable and c) relations over the stated variables also known as constraints [3,4]. Solving a CSP means finding an assignment of values for each variable in such a way that all constraints are satisfied [3].

Regardless the considerable number of literature on layout configuration and CSP [5,6,7,8], our problem include three characteristics never considered simultaneously: Its deals with the allocation of an *unfixed number of rectangular* panels that must not overlap, frames (existing windows and doors) must be *overlapped* by one and only one panel, and facades have specific areas providing certain *load-bearing capabilities* that allow to attach panels. Thus, as far as we know, no support system nor design system is well-suited for addressing such particularities. Also, most systems are desktop-oriented and not web-oriented, making difficult to adapt new requirements and functionalities as they need new versions to be released.

The goal of this paper is two-fold. First, we propose an architecture that divides initial filtering and consequent search for constraint-based product configuration. The architecture allow us to solve two configuration tasks; configure a set of questions relating the renovation model needed in the renovation process and needed to determine limits for panels' size and panels' weight and; configure a constraint satisfaction problem for each of the facades to renovate. In a second time, we present an on-line support system, and formalize its behavior, for the problem of facade-layout configuration.

The paper is divided as follows. A brief description of the industrial process and the configuration sequence is presented in Section II. In section III we introduce details of the two configuration tasks performed by the support system. The service oriented architecture, along with details of the constraint-services' behavior, is presented in Section IV. In Section V we discuss the benefits of the tasks division and coupling of the constraint systems. Some conclusions are drawn in Section VI.

II. RENOVATION PROCESS

In order to start the configuration process, specific information have to be extracted from a set of spatial entities. The renovation is carried out on *facades* that are part of a given *building* part of a given *block* part of a given *working site*. Each of these spatial entities have geometrical and structural properties and may have different environmental conditions that must be taken into account for the layout-plan definition.

Information about spatial entities is acquired by the support system by means of an input file describing all geometrical and structural properties, and by means of a web-based questionnaire for each spatial entity in the input file. After questionnaire completion, the lower bound and upper bound for panels' size and panel's weight have been deduced. Also, given that several instances for facades are need to be solve, at the end of the questioning stage the systems creates a constraint satisfaction model for each facade using the inputed information and the deduced limits for panels' size and weight. Here, each constraint satisfaction model instance is parameterized according with the facade information (e.g. environmental conditions) and the particular deduced panels' limits.

Lets consider the information flow from the user perspective. We highlight the fact that to the end-user should be transparent all the configuration process. The complete sequence of the configuration goes as follows.

(1) The user uploads a file containing the geometry and structural specification of spatial entities. The support system stores information in a data base.
(2) The filtering service presents a questionnaire for each of the spatial entities in the input file.
(3) The user answers the questions (leaving in blank the questions he does not know the answer).
(4) Using the information about spatial entities (database) and their environmental/user conditions (user answers) the system deduce lower and upper bounds for panels' size and panels' weight by using the filtering service.
(5) If a manual configuration is desired, the user draws each panel on the clients GUI. Each panel is assured to be consistent with the problem requirements by sending its information to validate into a solving web-service.
(6) If a semi-automatic configuration is desired, the user draws some panels and then asks the solving service to finish the configuration.
(7) If an automatic configuration is desired, the user asks the solving service to provide layout-plan solutions.

III. CONFIGURATION TASKS

We describe the two configuration tasks within the support system: The configuration of a questionnaire to be filled by the end-user and, the configuration of a constraint satisfaction model for each facade to renovate used as input for layout-plans generation.

A. Questionarie

The renovation includes four spatial entities, namely, working site, block, building and facade, and some configurable components, namely, panels and fasteners (fasteners are devices to attach panels onto the facades). Once the input file has been read by the support system, it can proceed by configuring a set of questions for each spatial entity in the file. Then, after the user answer the questionnaires, the system configures, i.e., deduces, the limits for panels' size and panels' weight for *each facade*. The questionnaires ask the following information.

- *Working site*. This is the bigger spatial division in the renovation. It is commonly referred by a name and is well-know by the community. Values provided by the user are:
  - Number of blocks in the working site?
  - Working site is in a windy region?
  - Season when renovating?
  - Target for cost?
  - Target for performance?
  - Obstacles presence?
  - Accessibility to the working site?
  - Panel's width and height lower bound?
  - Panel's width and height upper bound?
  - Panel's maximum weight?
- *Block*. Is a set of buildings which are usually attached by a common wall. Values provided by the user are:
  - Number of buildings in the block?
  - Obstacles presence?
  - Accessibility to the block?
  - Panel's width and height lower bound?
  - Panel's width and height upper bound?
  - Panel's maximum weight?
- *Building*. Is the actual place where apartment are arranged and is the host of several facades. Values provided by the user are:
  - Number of facades in the building?
  - Obstacles presence?
  - Accessibility to the block?
  - Panel's width and height lower bound?
  - Panel's width and height upper bound?
  - Panel's maximum weight?
- *Facade*. Maybe seen as a big wall, but is in fact a composition of apartment along with its doors, windows and so on. Values provided by the user are:
  - Obstacles presence?
  - Accessibility to the block?
  - Type of attaching device {bottom, top, lateral}?
  - Panel's width and height lower bound?
  - Panel's width and height upper bound?
  - Panel's maximum weight?

B. An independent csp for each facade

One critical aspect of the support system is the ability to configure a constraint satisfaction model for each facade to renovate. This is important because, first, each facade has (potentially) different size, number of windows, supporting areas etc. Possible positions for panels, for instance, must lie between zero and the facade width and height. Simply put, each facade has its own configuration parameters used in the constrain satisfaction model and in the layout generation process. And second, each facade may have different accessibility conditions, obstacles or even user preferences. Thus, panels' size limits, as well as their weight, are constrained by the specific conditions of the facade and not only by the conditions of the working site, block or building.

When configuring these CSP instances it is important to conserve downwards consistency. Downwards consistency refers to the fact that information on higher level of the renovation are is propagated to the inferior levels but it can not propagate upwards. As an example consider only accessibility conditions, obstacles presence and panels' size limits, for the specification in Fig. 1.

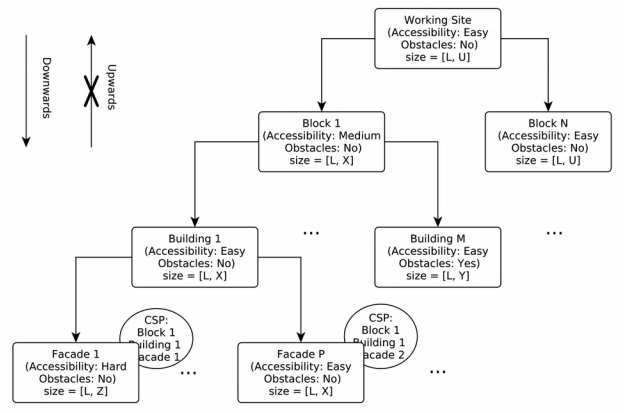

Fig. 1. One CSP for each facade.

Note that inferior entities on the hierarchy inherit values from superior levels. But, it is not the case that information on superior levels should be consistent with information on inferior levels. In Fig. 2, for instance, facade 1 has a hard accessibility condition and thus the upper bound for panels' size is reduced to a given Z. This upper bound is not propagated upwards to the building 1; it conserves its inherited value X. Consequently, facade 2 will inherit the value of X because no further reduction is needed for their panels configuration. Naturally, it is the case that Z is less than or equal than X, and X is less than or equal than of U. Using this information a CSP is configured for each facade to renovate.

IV. CONSTRAINT SERVICES

In order to divide configuration tasks we divide the support system in two services that may be implemented in different servers. In essence, information about the renovation, entered by means of the input file and the questionnaire, is filtered by means of the first service called *Filtering Service*. Then, the second service called *Solving Service*, upon user request, uses its configuration algorithms to provide complaint layout-plans solutions. Fig. 3 presents the architecture of the support system.

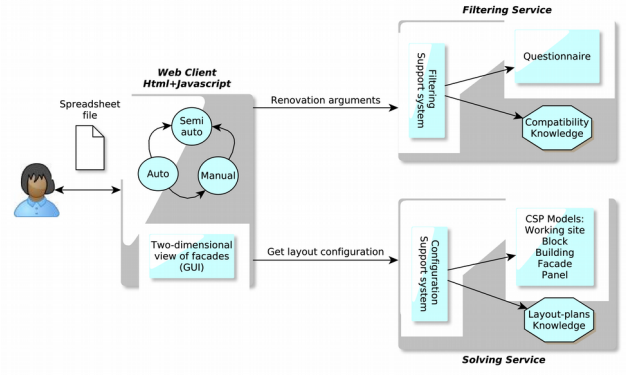

Fig. 2. On-line support system architecture.

In order to give a clear understanding on how the services works, let us describe the input in a formal way. For each of the services the input is a tuple of the form $\langle SPEC, V, D(V), C(V) \rangle$ with

1. $SPEC = \langle WS, BK, BG, FAC \rangle$; WS variables describing the working site, BK variables describing blocks, BG variables describing buildings and FAC variables describing facades.
2. $V = \langle P, FA \rangle$; P variables describing a single panel and FA variables describing a single fastener.
3. $D(V) = \langle D(P), D(FA) \rangle$; domain for each one of the variables in V.
4. $C(V)$ a set of constraints over variables in V.

Information in $SPEC$ describes only properties of the spatial entities such as the number, sizes, positions, etc. Variables in V and $D(V)$, on the other hand, are manufacturer dependent and include size and position of panels and fasteners, and their initial size and weight domains which depends on the manufacturing process. Constraint in $C(V)$ are extracted from the problem domain by an expert and are different in each service.

A. Filtering service

Mapping. The filtering service is in charge of removing domain values from elements in $D(V)$ that are not allowed by the established constraints. Here, constraints $C(V)$ describe valid combination among different parameters in $SPEC$ and variables in $D(V)$. We denote this set of constraints $C_f(V)$ to distinguish them from the ones used on the solving service. These constraints are formalized as compatibility tables. Formally, the filtering is a mapping M from variables and domains to domains

$$M\left(SPEC,V,D(V),C_f(V)\right)\rightarrow D'(V)$$

The result *D'(V)* contains the new domain for panels and fasteners, where $D'(V)\subseteq D(V)$. As stated previously, the initial filtering has as goal setting domains for configurable components and takes spatial entities information and constraints to do so. In our on-line support system we use the Cofiade [9] system to perform this filtering. Several reasons support our choice. First, the system is already on-line, making it usable in no time. Second, it is well conceived for supporting decision-making processes. And third, it uses efficient compatibility tables for domain pruning; applying a given compatibility table is made in constant time *O(1)*.

Compatibility Knowledge. Configurable components of the renovation are panels and fasteners to attach panels. Panels are configurable by fixing their width, height, weight and position over the facade. Fasteners are configurable by fixing its length and setting its type {bottom, top, lateral}. The following compatibility tables, presented from Table 1 to Table 6, show the allowed combination between user's input values and configurable components values.

TABLE 1

Relation C1 between obstacles in spatial entities and panel's size, where ϕ and σ are upper-bounds for panel's width and height, respectively.

Obstacles	Panel's size
Yes	$(w_p \leq \phi) \wedge (h_p \leq \sigma)$
No	-

TABLE 2

Relation C2 between accessibility and panel's size, where λ and π are upper-bounds for panel's width and height, respectively.

Accessibility	Panel's size
Easy	-
Hard	$(w_p \leq \lambda) \wedge (h_p \leq \pi)$

TABLE 3

Relation C3 between renovation cost and panel's insulation: It illustrates the fact that the quality of the insulation depends on the user budget. Cost in Euros.

Cost	Panel's insulation
< 50000	low
[50000,100000]	medium
> 100000	high

TABLE 4

Relation C4 between desired performance and panel's insulation: It illustrates the fact that the quality of the insulation depends on the desired final energetic performance.

Performance	Panel's insulation
< 25	low
[25,50]	medium
> 50	high

TABLE 5

Relation C5 between panel's weight and fasteners' positions.

Weight	Position Fasteners
< 500	-
[500,1000]	Top, bottom
> 1000	bottom

TABLE 6

Relation C6 between fasteners' position and number of fasteners.

Position Fasteners	Panel's size
Top, bottom	{2,4,6}
Laterals	{4,6}

B. Solving service

Search. The second service in the support system is in charge of layout-plans generation. The system uses several algorithms to generate layout plans but, although their behavior are quite different, their semantic remains the same. It uses Choco [10] as underlying solver.

Now, while information in *SPEC* and *V* are the same as the filtering services, it is not the case for domains and constraints. To differentiate them lest call the input domains $D_s(V)$ and the constraints $C_s(V)$. Intuitively, variable domains $D_s(V)$ are provided by the mapping of the filtering service, i.e.,

$$M\left(SPEC,V,D(V),C_f(V)\right)=D'(V)=D_s(V)$$

where *D(V)* is the initial variable domain of the problem. Constraints in $C_s(V)$ are stated as first order formulas and express, not compatibility among elements but, requirements for valid layout plans (see next section for a description of these constraints). The output of the server's process is a set of layout-plan solutions. Formally, the server's process is a function of the form

$$F\left(SPEC,V,D_s(V),C_s(V),H\right)=\langle X,Y,DX,DY,W\rangle$$

where *X* and *Y* represent the origin of coordinates, *DX* and *DY* the width and height, respectively, and *W* represent the weight for each panel in the solution. Additionally, the function is parameterized by an heuristic *H* stating the way the solution space is explored. Available strategies are greedy and depth-first search.

Layout knowledge. Let *F* denote the set of frames and *S* the set of supporting areas. Let $o_{e.d}$ and $l_{e.d}$ denote the origin and length, respectively, of a given entity *e* in the dimension *d*, with $d\in[1,2]$. Additionally, lb_d and ub_d denote the length lower bound and length upper bound, respectively, in dimension d for all panels. Each panel is described by its origin point w.r.t. the facade origin and its size. For convenience, lets assume that *P* is the set of panels composing the layout-plan solution. Then, each $p\in P$ is defined by $\langle o,l\rangle$ where $o_{p.d}\in[0,o_{fac.d}]$ is the origin of panel *p* in dimension *d*. and $l_{p.d}\in[lb_{p.d},ub_{p.d}]$ is the length of panel *p* in dimension *d*. The following six constraints express the relations among panels, and panels and facade that must respect a layout solution.

1. Manufacturing and transportation limitations constrain panel's size with a give upper bound
$$\forall\, p \in P, d \in [1,2], l_{p.d} \leqslant ub_d$$
2. For two given panels p and q there is at least one dimension where their projections do not overlap
$$\forall\, p,q \in P, p \neq q, \exists d \in [1,2] \mid o_{p.d} \geq o_{q.d} + l_{p.d} \vee o_{q.d} \geq o_{p.d} + l_{p.d}$$
3. A given panel p must either be at the facade edge or ensure that enough space is left to fix another panel
$$\forall\, p \in P, d \in [1,2], o_{p.d} + l_{p.d} \leq l_{fac.d} - lb_d \vee p_{p.d} + l_{p.d} = l_{fac.d}$$
4. Frames must be completely overlapped by one and only one panel respecting minimum distance Δ
$$\forall\, f \in F, \exists p \in P \mid o_{p.d} + \Delta \leq o_{f.d} \wedge o_{f.d} + l_{f.d} \leq o_{p.d} + l_{p.d} + \Delta$$
5. The entire facade surface must be covered with panels $\sum_{p \in P} \prod_{d \in 1,2} \left(o_{p.d} + l_{p.d}\right) = \prod_{d \in 1,2} l_{fac.d}$.
6. Panels' corners must be matched with supporting areas in order to be properly attached onto the facade
$$\forall\, p \in P, \exists s \in S \mid o_{s.d} \leq o_{p.d} \wedge o_{p.d} + l_{p.d} \leq o_{s.d} + l_{s.d}$$

V. BENEFITS

Benefits for configuration tasks division are rather simple. On the one hand we apply the well-known principle d*ivide and conquer*. In our on-line system this principle allow us to add or remove variables, domains and questions in the filtering service, i.e., by means of adding or removing compatibility tables. In addition, as we use Cofiade [9], we may mix different variable representation as integer domains, continuous domains and symbolic domains whereas in most constraint systems mixing variable domains is not allowed or is not efficient.

On the other hand, as a benefit of tasks division, we improve performance by avoiding the use of binary equalities and binary inequalities constraints whose computational time is O(n*m), where n and m are the number of values in the domain of the two variables involved in the constraint. Thus, at the moment of finding solutions, the underlying constraint solver, in our case Choco [10], propagates and applies search using only those constraints defining a layout plan.

The decoupling of configuration tasks is supported by the underlying declarative model. Indeed, the monotonic properties of constraint satisfaction make it possible to add knowledge (constraints) on one system without loosing any solution on the other system. Thus, the declarative view of constraint satisfaction make it possible to handle services as independent communicating agents.

VI. CONCLUSION

The aim of this paper has been to introduce an architecture for constraint-based product configuration coupling two constraint-based systems. We have presented an architecture that divided initial variable domain filtering and space exploration. The method divide and conquer allow us to make straightforward adaptations to each service separately. Our approach have been applied to facade-layout configuration and implemented in an on-line support system. For this particular scenario we have a) formalize each service behavior and the relation among them, b) presented the constraints, stated in compatibility tables and carried out by the Cofiade system, for initial filtering, c) presented the constraints, stated as first order formulae and carried out by Choco constraint solver, that are used to generate compliant layout solutions, d) show how to solve the configuration tasks by coupling the two constraint-based systems and e) show that consistency and integrity of solutions are straightforward modeled and implemented thanks to the monotonic view of constraint satisfaction.

REFERENCES

[1] Dong Yang, Ming Dong, and Rui Miao, 'Development of a product configuration system with an ontology-based approach', Computer-Aided Design, 40(8), 863 – 878, (2008).

[2] E. Vareilles, A. F. Barco, M. Falcon, M. Aldanondo, and P. Gaborit, 'Configuration of high performance apartment buildings renovation: a constraint based approach', in Conference of Industrial Engineering and Engineering Management (IEEM). IEEE., (2013).

[3] S. C. Brailsford, C. N. Potts, and B. M. Smith, 'Constraint satisfaction problems: Algorithms and applications', European Journal of Operational Research, 119(3), pp. 557 – 581, (1999).

[4] U. Junker, Configuration., Chapter 24 of Handbook of Constraint Programming (Foundations of Artificial Intelligence). Elsevier Science Inc., New York, NY, USA, 2006.

[5] M. Zawidzki, K. Tateyama, and I. Nishikawa, 'The constraints satisfaction problem approach in the design of an architectural functional layout', Engineering Optimization, 43(9), pp. 943–966, (2011).

[6] S Shikder, A Price, and M Mourshed, 'Interactive constraint-based space layout planning', W070-Special Track 18th CIB World Building Congress May 2010 Salford, United Kingdom, 112, (2010).

[7] R. S. Liggett. Automated facilities layout: past, present and future. Automation in Construction, 9(2):pp. 197 – 215, 2000.

[8] Amine Drira, Henri Pierreval, Sonia Hajri-Gabouj, 'Facility layout problems: A survey', Annual Reviews in Control, 31(2), pp 255-267, 2007.

[9] Elise Vareilles. Paul Gaborit. Michel Aldanondo. Sabinne Carbonnel. Laurent Steffan, 'Cofiade constraints filtering for aiding design', in Actes des neuviemes Journ´ees Francophones de Programmation par Contraintes, Toulouse France, (2012).

[10] C. Prud'homme and JG. Fages, 'An introduction to choco 3.0 an open source java constraint programming library', in CP Solvers: Modeling, Applications, Integration, and Standardization. International workshop., Uppsala Sweden, (2013).