ABSTRACT

In-Between Architecture Computation describes the evolution of the Computational Design approach at the AedasR&D Computational Design and Research group founded in 2004 at Aedas architects in London. The approach has transformed itself from an academic inspired thinking about computing media to a more flexible model of design heuristics and search algorithms that finally start to produce new hybrid design workflows in the industry while also swimming against the industry trend of super-integration software. Only if computing is not exclusively defined through architectural design intent or purely computing logic, does computational design explore new design thinking.
1. Introduction

The title of this paper describes an approach that has crystallized over many years of implementing Computational Design and its theories in a professional context at Aedas architects and other international clients of the Computational Design and Research group of Aedas|R&D (CDR). It tries to explore the reasons and describe the decisions that led the group to this approach within an industry context.

In-Between Architecture Computation intends to encapsulate the problem that Computational Design often tends to be either Architecture or Computation but seldom leads to a feasible synthesis between the two disciplines. ‘True’ Computational Design must sit in-between the two fields and therefore demands new standards for design thinking, its professional workflows and the use of algorithms. Computational Design if conducted properly will have to sit at highly integrated nodes in the network of disciplines partaking in the complex task of designing buildings. As such, Computational Design should be analogous to a high ‘betweenness’ value in graph centrality terms and well linked to all stages of the design process.

To capture the essence of the CDR group’s approach in the industry context briefly, we intend a lose network of small applications with precise scopes that can be but don’t have to be woven into the design workflow at discrete stages and scales. The approach is a kind of ‘dis-integration’ or differentiation of computing over the span of a design process rather than ‘total integration’, the currently more dominant current paradigm such as Building Information Modelling (BIM).

2. Computational design research in the architectural industry

The scale and stages of application of Computational Design in the industry often depends on the size of the practise and if the practise perceives itself to produce ‘signature’ architecture.

Small to medium sized studios – especially those set up by graduates with experience in computing – have to participate in most stages of the design workflow and thus are able to integrate computing better early into the briefing and design concept stages. However, their very size or experience doesn’t allow them to build complex buildings or masterplans where computing is more adequate. Thus, they usually apply computing to sculptural aspects and installations. The other domain for smaller and young studios is to consult medium to large offices on specialist aspects such as cladding packages or structural / construction solutions.

On the other hand, large global firms can afford to sponsor internal or external design research and technological innovation. But they have also developed well established workflows with larger design teams, standards and distinct stages. When large firms like Skidmore Owen and Merrill, Forster’s or Hadid started to employ digital media – CAD, parametrics and
computing – it was and mostly still serves to enhance established workflows and design stages. Only within established stages of a design innovation is taking place that result in new standards of production. Gehry architects went as far as outsourcing their new standard in surface, assembly and information production into Gehry Technologies.

To change existing workflows carries risks when dealing with large commissions\(^1\). Computational Design therefore is often restricted to the scope of a work-stage or a singular aspect of a design in order to limit potential fall-out from an incomplete or buggy code development. Otherwise, Computational Design is used to develop quantitative evaluation and analysis methods that form part of a design stage iteration as performance check rather than solution search. Such checks commonly are restricted to explicitly quantifiable parameters such as structure, climatic performance or schedules. Rarely does it engage with spatial, occupational or organizational design and evaluation.

Gehry Technology’s Digital Project, Autodesk’s Revit and Bentley’s Building softwares are trying to minimize the risk of development by integrating as much information and parameters (not process) as possible into one platform that can span the entire workflow. Semi-open development platform attached to CAD packages such as Generative Components and Grasshopper are already designed with a traditional workflow alignment in mind but allow for more process thinking.

Apart from workflow synchronization constraints, Computational Design faces other hurdles in larger practice. For some ‘Starchitect’ studios – be they large or small – the very presence of a signature style limits the introduction of computational generation and search of the design space. Technological innovation in the form of computing exists of course but is forced into rationalizing a known approach and not undermining the lead architect’s intentions.

For large commercial offices, clients’ expectations can limit deeper engagement with computation and innovation in general. Or they can channel technological innovation into monotonous implementations like repeated designs of ‘landmarks’ or ‘icons’ with well established procedures like tangent arcs surface construction. Therefore, educating the client by convincing them of the value of computational design can increase the pressure on large firms to innovate, such as the request for Space Syntax like graph analysis for spatial integration [1].

3. Dis-integrate

To arrive at our current in-between system of light applications we had to first of all ‘unlearn’ some traits that we had learned either at university or when working in isolation. From 2004 onwards we attempted to implement...

\(^1\)Hugh Whitehead of Foster’s expressed his concern about our computation approach during a panel discussion at the CICA in 2005 saying that Foster would not take the risk of allowing ‘generative [computational] design’ as it might not produce what is required as input/output within their workflow.
popular academic notions of systems theory, complexity and emergence by building larger simulations in SDKs like Visual Basic for Applications for AutoCAD or Microstation or C# for Rhino – chosen for their easy I/O with the design team and built-in geometry objects – trying to span more than one design aspect. Such applications included a Tower generation and optimization program [2] as seen in “Figure 1” that would

1. generate 3 dimensional layouts for office buildings via a user weighted Cellular Automaton representing the adjacency matrix of imported accommodation schedules
2. generate floor-plates and cladding from an implicit surface algorithm (marching cubes)
3. analyse the envelope for solar exposure
4. calculate area and cladding schedules
5. evolve the resulting tower via a multi-criteria Genetic Algorithm and hybrid selection process (natural + artificial) to optimize for floor-to-wall ratio and user aesthetics

Or a masterplanning project of even greater complexity was produced the same year in 2005 with a student from the MSc Computing & Design at University of East London [3] as shown in “Figure 2” that

1. produce the plot ratio and circulation via an ant-colony algorithm
2. generate envelopes via L-system
3. evolve massing via Pareto optimization
4. units allocation
5. density & block accessibility
6. solar exposure

Projects of such complexity were valuable to communicate the potential of different algorithms to design teams and clients, research the scopes and implementations of algorithms and build code libraries for later recycling. However, large frameworks of this kind prove overly ‘closed’ or static for architectural briefs that require fast adaptation to respond to changing conditions in design iterations. Equally, the above incorporated scales and
aspects of design do not align with traditional work stages that British designers are used to (Royal Institute of British Architects work stages or the Council for Architecture and Build Environment). Design decisions are differently associated for architecture within the RIBA workflow stages or even across different scales for the CABE planning stages than were our hard-coded assumptions in those projects [4].

Almost by necessity, we shifted our research and development focus to either more generic aspects of design or bespoke developments to clear specifications of the design problem at hand. While bespoke development by specification produced the solutions intended by the design team, it also hardly produced surprise and did not allow the search of a brief’s design space – as described above for large architecture firms.

The development of generic design methodologies required an application to coincide with a singular decision within the design workflow. As such we first implemented either tested computational methods like Isovists for the measurement of visual integration of spatial layouts (early applications were called ‘Passive Supervision’) [5], agent-based models for egress simulations (‘People Movement’), or applications that simulate distinct design stages like adjacency diagrams (‘Adjacency Diagrams’), see “Figure 3”.

As communication of processes, parameters and outputs for generic design applications is more transparent, the dialog with design teams increases and feedback can be integrated more quickly, which leads to reduced development cycle times. Algorithms become more sophisticated with each project and interactivity becomes feasible as parameters are clearly defined. Since development iterates over several projects, findings and outputs reduce the risk of unknown alignment and results.
3.1. Agility and lightness

With the commission in 2007 to analyze the geometry of the World Trade Centre Memorial Museum New York for user experience based on movement and visual conditions, the CDR group had to extend its existing Visual Graph Analysis [6] applications – see “Figure 4”. A new strategy was introduced to simultaneously develop different types of algorithms for the same design aspects. Short cycles and frequent exchange led to faster progress and more rigorous coding. In the software industry this type of open and connected development is known as Agile Development and aims
to increase the usability of well structured lightweight software widgets rather than large monolithic single code [7].

In order to guarantee a fast exchange of widgets of code and interactivity for the design teams with real-time feedback, we have since resorted to build all code with Java and OpenGL. Only for fast prototyping, SDKs are occasionally still used.

The group continues on most projects with this agile approach and has since produced a large library of functionally autonomous but compatible lightweight applications with clearly defined input and output specifications. The scale of application of computation therefore becomes dependent on the combination with other widgets and architects can assemble them according to their specific workflow stages. For some typologies, such as transport, architects have been able to prompt clients to commission new work stages not existing in the traditional transport hierarchy, such as public realm impact studies.

Thus, a mesh between architect and code widgets has been created where computation sits in-between design decisions also exploring new design stages where computing affords novel insights. Additionally, through simultaneous and continuous development, healthy redundancy is built into the framework of applications that allows to identify new fields of architectural implementation and serves as a safety net.

The representation of this network of applications is affine to theoretical discourses about complexity in academia [8]. An omission or failure of a single application cannot disrupt the overall workflow. This condition could be recognized as a ‘distributed system’, both on the scale of the code description – the nature of heuristic search algorithms – as well as the meshing in with the design teams and work-stages.

The flexibility and openness that this disintegration affords, has created trust among architects and an opening towards more advanced design computation. The group has since been able to attempt a more directed re-introduction of powerful search and combinatorial algorithms for generative design that could be regarded as the true strength of computational design. Architects and design teams are starting to relax their design intentions and expectation if generative search applications are integrated that produce valid but undirected outcomes.

Shortly after the completion of the World Trade Centre Memorial Museum development, a government funded research project for urban design represented the perfect chance to combine the Agile development method with search (meta-heuristic) algorithms to establish the prototype approach the group has since consolidated.

4. Digital masterplanning

Digital Masterplanning started out as a knowledge-transfer project called Smart Solutions for Spatial Planning [9] in collaboration with the University of East London’s (UEL) Centre for Evolutionary Computing in
Architecture (CECA). The theme of the project’s funding body’s call was ‘Building Sustainable Communities’. The overall intention of the CDR group was to create a digital workflow from a matrix of applications as seen in “Figure 5” that loosely align with the planning stages. The digital workflow should generate urban structures that synthesize an urban designers/ planners heuristics and help gauge generally implicit assumptions about urban qualities such as ‘connectivity’, ‘accessibility’ or ‘mix & scale’. Two London Borough Council’s Regeneration and Urban Design teams – Newham and Tower Hamlets - as well as two Urban Design specialists – Urban Initiatives and Christoph Hadrys - collaborated to specify urban planning stages, input quantities, key-performance indicators and most importantly create explicit statements of their heuristics [9].

4.1. Strategy & scales

It was the explicit aim of the CDR group to avoid algorithmic representations, imitating existing manual approaches or building quantitative visualizations. Currently, algorithmic representations or quantitative visualizations are the standard in urban design simulations imposing algorithmic patterns instead of questioning the implications of computing for urban design, i.e. could there be novel synthetic heuristics. Instead of producing patterns, an alignment with criteria and processes that support decisions in urban design thinking should determine the structure of the digital workflow.
The ‘sustainability’ component of the funding project was aimed at urban morphological or structural sustainability, which we identified as the key stages of the urban planning procedures where ‘spatial design’ occurs but no computational design applications exist. Within the UK urban planning procedures this means that the applications support the ‘Spatial Strategy’ and ‘Masterplanning’ phases [10] [11].

From analysis with our partners, we drew up the following scales and stages within the Spatial Strategy and Masterplanning to be most relevant for urban structure design as seen in “Figure 6”.

Of all criteria for spatial planning, accessibility appeared the most important as it applies to all scales and phases of design. Access levels indicate development potentials, development schedules, density levels and land-uses as well as crime rates. Accessibility is so important that a masterplan requires to be accompanied by a separate ‘Design & Access Statement’ [10]. Therefore, the first set of applications dealt with issues of accessibility.

### 4.2. Accessibility

There are two existing accessibility methods for pedestrian or sustainable accessibility an urban designer can refer to: Public Transport Access Levels (PTAL) and Space Syntax evaluations [6]. Either one however is commission-based and therefore an expert service where the designer has
no interaction during its production or doesn’t apply them himself. This ‘design distance’ renders those access evaluations more opaque and places them firmly in the ‘analysis’ rather than design stages.

If applied during design stages, a designer has no explicit formal intentions towards accessibility other than the implementation of well accessible locations or relations between typologies. As such heuristic search algorithms suit the problem well.

Walking distance & connectivity

To calculate the walking distances (which on the legends are displayed as minutes), the site needed to be represented through a network. This representation consists of all the existing walking paths of the site, represented through line segments in a DXF file. It also includes all relevant accessibility locations set by the user, such as public transport nodes.

The application uses the Dijkstra algorithm to measure walking distances throughout the network to the given access locations [12]. The metric distances are multiplied with a ratio of 83 m/min average to estimate walking times. It does not take specific conditions like landscape or crowding into account.

If access times are not satisfactory, additional links within the paths network can be proposed by the program that would reduce the walking times. Links such as other paths, bridges or tunnels are calculated by performing a search that takes two parameters: maximum radius of link and minimum walking time gain.

► Figure 7. Exaggerated implementation of new links to illustrate network interventions at given parameters.
First the algorithm searches all possible connections in the network that are within the maximum radius, in order to find the link that reduces the walking distance of a specific area as much as possible. The process is iterative, suggesting a series of connections in order to improve the accessibility of the site. For each intervention the network accessibility is re-calculated by the Dijkstra’s algorithm.

Secondly, the minimum gain in walking time (minutes) can be set, which the proposed links must provide, for example a gain of 3 minutes over 50 meters – see “Figure 7”. If the constraints set cannot produce a solution, no link is shown. The user can continue to change the parameters for the search interactively. As the proposed links for the improvement for the path network happens in real-time, the user will gauge quickly where new links are necessary as well as understand general minimum walking time thresholds of the site without creating large new infrastructures.

Routes, depth & catchment

As a second set of accessibility applications, we build three routes and catchment simulations where all access points – departure or arrival – can be set and changed interactively and thus give the designer immediate feedback about their interventions – see “Figure 8”.

Apart from indicating whether access locations are placed correctly, simulating direct routes also indicate ‘desire lines’ – most travelled routes – that inform land-use allocation. Retail elevations and entrances will obviously want to sit along a highly frequented route unless otherwise intended.

All applications are based on the field of ‘motion planning’, which is primarily used in robotics to calculate routes for vehicles [12]. The input representation for the applications requires only closed polylines for building or obstacle outlines in a DXF format. Access points can optionally be set a priori or interactively.

The Visibility Graph is calculated from the polygons’ vertices that produces all possible visible connections. On that graph, the Dijkstra algorithm is applied again to generate direct metric routes from all-to-all or all-to-one access locations. A grid or perimeter of points can be set to add noise to the input data if departure quantities or locations are not exactly known.

In additional current developments we have added footfall calculations along aggregate shortest routes. While all routes are calculated on the basis of ‘minimum’, the metric can change between

- Distance
- Depth
- Angle

Each metric can be interpreted differently for pedestrian routes. Shortest distances are taken if the pedestrian knows the site, i.e. a local. Depth is measured by least-turns and as such indicates the simplest route
between locations. If a pedestrian is not familiar with a site (or gets instructions), the most likely route will be the one with least turns (also called Manhattan Distance).

Angular distances indicate the accumulated amount in degrees by which one has to turn before arriving at a location. While the depth analysis is improved by setting interstitial grid locations off the Visibility Graph, angular analysis actually reflects ‘true depth’ as perceived on the ground.

To calculate the catchment areas, a hybrid between nested Isovists and motion planning was applied. A geometric method of circle segment cropping was added to calculate the thresholds of access times and to visualize the resulting contours.

The multitude of access simulation applications is a direct result from using the ‘agile’ development method mentioned earlier as small development steps could be exchanged and enhanced to produce new applications.

4.3. Urban structure

*Circulation [primary | secondary]*

As the test site in East London was an inner urban site, circulation principally is designed on two levels: primary and secondary. Primary circulation identifies the axes connecting major activity locations or places of interest such as landmarks, markets, stations etc. while linking up open routes from the context. The secondary circulation most likely derives from...
a set urban grid, block character and development densities. We devised two applications to generate a circulation for either stage.

The primary circulation searches the set of routes that connect all activity nodes and access locations with the minimum circulation length. That means that the resulting circulation affords the shortest possible travel time between any location. This promotes short walking or driving times between all parts of the site and its context.

The generation of the primary circulation is based on the K-minimum spanning tree. The tree is based on a specified graph that represents the sub-set of a number of nodes – here the locations of interest. The input to the application consist of all possible main route edges through or within the site in the DXF format.

The secondary circulation takes the primary circulation, a set urban grid size and secondary access points or locations of interest as input. The grid nodes are connected by edges and form a large network graph that ties into the primary circulation. The grid size itself is dependent on the character of the place the designer wants to create.

Also the secondary circulation should produce a network of paths that represent the shortest possible length of all routes to connect all locations – this time the primary and secondary access points. But while the K-minimum spanning tree is limited to a given set of nodes, the secondary

Figure 9. A still from the Ant-Colony-Optimization running within the primary circulation and a grid.
circulation relies on interpolating new nodes to the graph and finding new routes using the grid edges.

As a consequence, a custom code was created based on the meta-heuristic algorithm described by Blum and Blesa [13], albeit approximating a Steiner tree rather than a K-minimum spanning tree. As the search represents a NP-Complete class of problems where the possible solutions increase exponentially if the input increases linearly and solutions cannot be solved in finite time, we applied an Ant Colony Optimization algorithm to accelerate the search for 'good solutions'- see “Figure 9”.

The implementation consists on generating initially a random tree that connects the primary and secondary access points, and then marking with a pheromone value all edges included in the tree. This pheromone value depends on the length of the tree: the shorter the tree, the higher the pheromone deposited. The trees generated after this first random one, will be biased to include edges with high pheromone value, and thus iteratively converging towards a good solution.

The result from the secondary circulation represents not just the full circulation network but also the development plot outlines. Both can be fed-forward to the next application to generate scenarios for land-use mix and density levels.

Relational land use mix & development density

The land use mix and density application exposes the advantages of computational design as it can generate distributions on a finer scale than manually or CAD produced zoning diagrams. In combination with previous circulation and plot outlines simulations, the land use mix simulation unhinges traditional planning stages especially the mentioned coarse grained zoning for Area Action Plans and Masterplans [11]. Spatial strategies can be formed more detailed at an earlier stage and assumptions about distribution, density and development schedules can be evaluated through simulation early on.

The application generates possible and valid scenarios for land-use allocations based on adjacency preferences to other land-uses and conditions on site while also fulfilling accessibility conditions - see “Figure 10”. Just as land-use locations are relational their density given in height for each plot is also calculated in relation to its surrounding land-uses, their heights and site conditions.

The input consists of a DXF file with site features, such as rivers, motorways, parks or green areas. Either previously generated circulation or manually drawn circulation serves as input and outline development plots.

Alongside the routes a generic mesh covering the site can either be generated or manually put into the program through the input DXF file. The computational process works by modifying this mesh, and its properties are therefore important for the output of the program. Each of the mesh edges are considered by the program to either act as a traversable path, or it can contain one of the land use components, in which case it may only be
traversable for some uses (for example if it is a park one should most likely consider it to be possible to walk through).

Apart from the geometric features, quantities of land-use units of the development schedule as well as desired adjacencies amongst them and site conditions are read in from XML. Maximum development heights have to be specified for each land-use.

The application uses a technique called Quantum Annealing (QA) for finding solutions that satisfy the adjacencies sufficiently well, without the strict requirement of finding the global optimum solution [14].

Formally, the algorithm tries to figure out the combinatorial problem of which edges in the mesh should be kept open (as roads or undefined area) and which edges should be occupied by different land-uses specified. The algorithm operates by switching the edges between the different types of land-uses (each defined as a set of edges). An initial distribution of all the land uses is generated randomly by placing the appropriate amount of edges in each of the sets according to the specified land-use areas, and subsequently new solutions are generated by exchanging edges between the different sets.

The QA is similar to a hill climber in that it tests out solutions that are close to it and moves always to new solutions if they have a better score than the current one. The uniqueness of the QA algorithm is that with time the magnitude of its sampling area is changed, meaning that it selects solutions that may be further away in the beginning of the process, and continuously narrows this area down with time. In this application it starts with sampling the entire search space and decreases continuously over time, until finally converging towards two edges [14].

Block definition

The output from the land-use mix and density application proposes good-enough development scenarios from development quantums and indicates heights for each land-use location. It doesn’t however generate block outlines. This is left to the designer as a translational step before entering the outlines as footprints into the last application in the series produced during the Smart Solutions for Spatial Planning project. This manual transition fits into our semi-automatic open approach to minimize risks and integrate the designer into a digital workflow.
The block definition program is based on the collaboration between UEL CECA and AedasR&D in 2005 and is still using VBA for AutoCAD [3]. The program was developed by the project partner 4M under the supervision of the CDR group.

The aim of the block definition program is to generate a mix of units into a massing envelop within a block outline. The block outlines and target criteria for the generative process are input via a DXF and a text file for quantities.

As the generative process tries to solve for 9 criteria, a Pareto Optimization algorithm from the field of Evolutionary Computing has been implemented that produces best-fit solutions for each block within the site. The target criteria include

- Unit depths
- Unit ratios
- Solar exposure for each unit type
- Plot density ratio
- Perimeter continuity

To decode each phenotype an L-system was used that stacked the units into the block. The binary strings for each expression were evolved through complex archiving and selection procedures using the Pareto fronts. 'A Pareto front are archives of solutions with varying degrees of equality amongst them. The best front contains solutions where parameters are not improving at each others expense and thus produce a well-balanced compromise between apparently contradicting performance criteria. After several hundred generations of evolution employing cross-over, mutation and selection, the Pareto fronts converge with the error for each target criteria reduced to a minimum, see “Figure 11”.

Figure 11. A massing scenario on site from the Pareto Optimization. The massing results from the application of the complete SSSP matrix.
4.4. Dis-integration and compatibility

While scales and stages have been carefully represented by differentiated algorithmic processes and correlating design criteria, it is an essential aspect of Digital Masterplanning that inputs and outputs between applications are the compatibility. For this compatibility many different hybrid workflows can be assembled from the set of applications, depending on brief and time-frames. Additionally, risks are minimized as each application can be replaced by a designer's manual I/O. The essence of dis-integration becomes flexibility and a high potential for designer-application based integrated workflows.

4.5. Professional implementation

Before this open platform for Digital Masterplanning was finished, commissions for its applications started. Its versatility, designer-friendliness and unique approach to a spatial planning have made it a favourite among our developments. The ‘agile’ principle appears to be successful and further applications have emerged from this first set, expanding the platform continuously.

By now Digital Masterplanning is applied to strategic planning, investment appraisals, cost consultancy, urban design, transport planning, landscaping, masterplanning, design & access statements etc. After supporting masterplans in Belarus, UK, India and Kosovo, we are currently working on the development of Masdar City’s largest neighbourhood in Abu Dhabi.

5. Conclusions

5.1. Brief and abstraction

The original funded project Smart Solutions for Spatial Planning was seized as a chance to develop the agile approach and attempt to synthesize design heuristics with algorithmic meta-heuristics (heuristic search). While the language of the development brief between the planners, urban designers and the CDR group appeared to contain no ambiguity, the initial results were not instantly accepted by our partners. For the urban structure aspects of circulation network and land-use mix, several applications were built that didn’t differentiate sufficiently between scales and processes,
apparently mixing too many design intentions. For instance, the Relational
Land-use Mix & Density application first incorporated geometric block
ratios and visual representations of such blocks. Hence, the distribution of
land-use units was tied to geometric descriptions rather than topological
and geographical specifications. After divorcing the geometric ratios and
massing from the topologic arrangements and giving each aspect a set of
meta-heuristic processes with distinct input and performance criteria, the
quality of computing was revealed to the partners. The higher the level of
disambiguation between representation and process, the more likely design
teams and partners will be able to identify their design heuristics and its
drivers with the computational processes.

It is no coincidence that the last stage of the Digital Masterplanning
process – Block Massing – is so far the least implemented as it contains the
highest amount of ambiguity in representation, i.e. doesn’t hit the right level
of differentiation into process and abstraction.

5.2. Driving communication between silos

The role of computation as a diagrammatic design simulation that
encapsulates process and heuristics rather than visuals (non-abstractions) of
specific final solutions, provides a strong relay for silos to communicate each
others drivers. Drivers as process and design intentions rather than just
performance criteria and parameters. Recent digital urban design software
appears at aiming to imitate the role that CAD plays for architects where
no process simulation takes place but data-manipulation in the form of
regulations represents the only way of driving the design. While large
amounts of data seems integrated and concatenated, the designer as user
finds himself external to the computation and only communicates about
input/ output quantities.

Where process and design heuristic simulation takes place interactively
as in Digital Masterplanning, knowledge about ways-of-doing, i.e. design
drivers can be identified and shared. Process simulation and interaction
allows diverse silos to tag their drivers into/onto the process and render
them explicit in a common language, that of simulation.

Consequently, it becomes the role of the computational designer to ask
the right questions about design heuristics and find the appropriate meta-
heuristic algorithms to synthesize and translate knowledge into processes.
Computational Design assumes the central role of in-between communication
via simulation.

5.3. Computational design research consultancy

The liberty of direction of research and long-term thinking enjoyed by
the Aedas|R&D and the CDR group, often absent in large commercial
firms, is partly due to the visionary management of some directors in the
firm and an absence of imposed design style.
As noticed by a variety of companies, the group has evolved into providing computational design research as a service where the knowledge built within the group over years is implemented also Aedas externally and supporting other companies to develop new heuristics via computation. It takes a number of years to consolidate a flexible and efficiently working computational design department (or any effective research department) with knowledge of a large number of algorithmic implementation from different fields.

Acknowledgements

The Aedas|R&D CDR team comprises Christian Derix, Pablo Miranda, Asmund Gamlesaeter and Lucy Helme. Our directors Peter Oborn and Brian Johnson as well as my R&D partner Judit Kimpian have continued to support the CDR projects, much as the whole of Aedas UK.

A lot of our inspiration is drawn from Paul Coates with whom we have studied and been working for many years.

References

9. Coates, P. and Derix, C., Smart Solutions for Spatial Planning: a design support system for urban generative planning, in proceedings eCAADe,Antwerpen, 2008