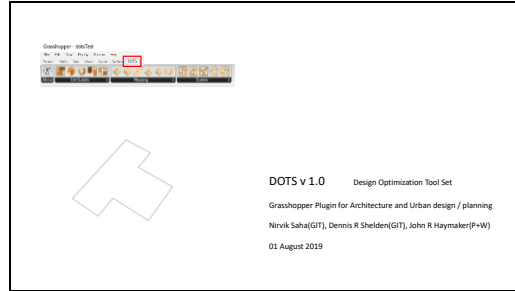


## Slide 1



This summer we have developed Design optimization tool set or dots. It is a Grasshopper plugin for architecture and urban planning. This plugin will generate appropriate configs for various types of design problems.

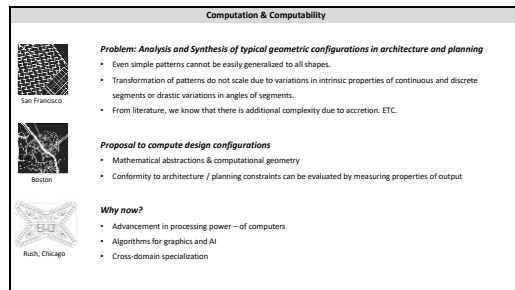
We hope to add analytical components in the near future and hopefully with your inputs, we can develop a comprehensive set of design tools.

We treat the RH-GH environment as a medium to package and distribute our algorithms.

We have taken care to organize & name the input-output fields so that knowledge of Rhino-Gh is not an impediment.

We have tried to make this plugin very easy to use.

## Slide 2



In geometry processing:

- Even simple patterns cannot be generalized to all shapes.
- Transformation of patterns do not scale well due to intrinsic properties of smooth and discrete segments or drastic variations in angles of segments.
- From literature, we know that there is additional complexity due to accretion.

It is proposed that typical design configs can be achieved by conducting a sequence of geometric transformations and evaluating logical arguments on specific parts of the region under consideration.







## Slide 3

Computation & Computability

Geometry Processing: Analysis and Synthesis of Planning Configurations

Using DOTS components

- Allows design Architecture & Urban Planning **exploration** through various algorithms.
- It will **try** to maximize the possibility of achieving input-constraints.
- Dots' components are meant to be used in **combination**.
- Appropriate optimization algorithms that respond to domain specific **constraints** are in-built or user-defined.
- Generates SMOOTH OR RECTILINEAR FORMS

Dots Workflow	Variations
	
	
	

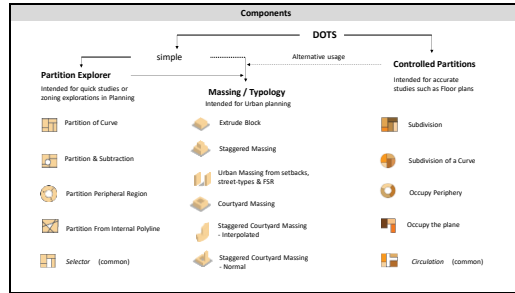
This is a step-by-step approach where partial results are preserved.

Intermediate results help in achieving a larger objective function because in case of an error, the computations do not start from the beginning but an intermediate step.

Breaking down the computational process helps in effectively exploring alternatives

because we can simply alter the parameters of an intermediate step and it will have a cascading effect.

## Slide 4



Primarily, there are two main operations - subdividing a large area into smaller pieces or parcels and then placing a spatial object in it.

Right now, we have provided four mechanisms of generating the necessary smaller regions –

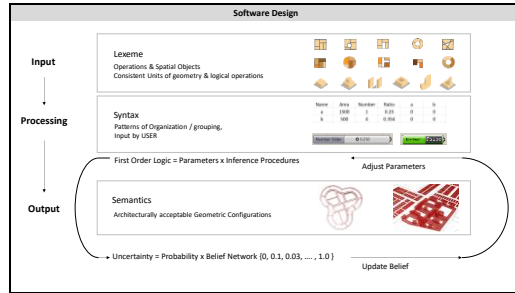
1. Recursively subdivide a region
2. By extending segments of an internal polyline
3. Place a set of regions in a plane by an exploratory process.
4. Place appropriate parcels along the periphery

For planning, we have provided components that produce well-known building typologies that can be easily placed in a given region.

These components vary in complexity and even though some of them may be developed using scripting tools, there are few considerations:

1. Can the script be generalized to all projects?
2. How many types of variations can be generated?
3. Can it handle all shapes & sizes of inputs?
4. Learnability & usage.
5. Speed is a primary concern to support large-scale design endeavors and typically Dots components are much Faster than grasshopper or python scripting -scale of 400-600 on average.
6. By classifying massing typologies we can generate various types of buildings by altering a few parameters.
7. Dots provides a comprehensive approach where the components can be connected to handle many types of problems. It is basically very easy to set up a project pipeline and add project-specific custom components. Our intention is to support the design exploration process.

## Slide 5

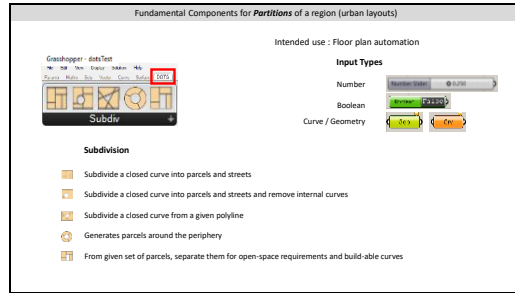


The user may think about the dots components as a way of passing inputs into a set of arguments followed by an evaluation process at the end of which, the output is exposed to the user. The arguments are a series of geometric and logical actions. The result of these operations is weighed against the input constraints. If everything is satisfied, output is generated otherwise, some parameters are changed under constraints until satisfactory results are found.

Dots implements a geometry library with in-built goals. Various permutations of the components will help in design exploration and they are meant to be used in conjunction.

As a software design, we treat the computational process as a language. Each component is a design element- for us it is a symbol or lexeme with internally consistent logic. The syntax is provided by the user determines grouping of elements. Finally, once the elements have been laid out based on the syntax, if they form a legible organization, it is accepted as a solution.

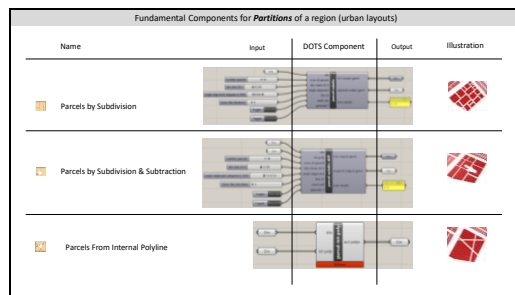
## Slide 6



With this information, we can dive into the components. This set of components, will allow the user to quickly break down a large region into smaller regions. There are 4 components that perform the subdivision operations and 1 component to select buildable parcels based on open-space requirements.

The inputs required are simple components for numbers, boolean, and external curve.

## Slide 7

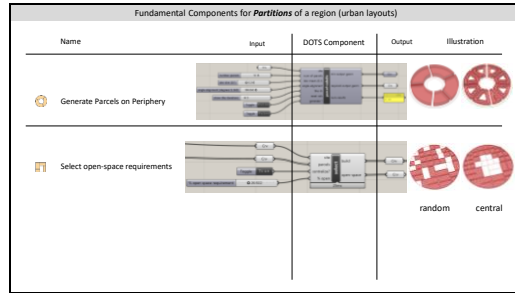


The first 2 components split up the input outer curve into orthogonal parcels internally and output a set of closed curves and street centerlines,

The second component also eliminates an input internal curve from the solution curves.

The third component takes in an internal polyline extends the lines and generates parcels for combinations of intersections.

## Slide 8

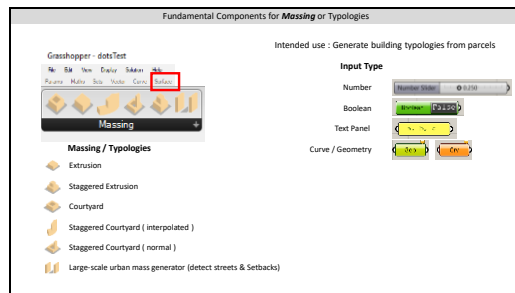


The first component generates parcels around the periphery which can be combined with massing typologies as shown.

The selection component's inputs are Output parcels, FSR & open-space requirements and this outputs parcels which should be built and those that can remain open. Either, open parcels are centrally located, or they are randomly dispersed depending on user input.

Next-step would be to use one/many attractor points to place the open-space parcels.

## Slide 9



This set of components are meant to generate massing / typology. We have attempted to:

- minimize the number of components required
- provide a way to generate the major typologies
- Speed of execution

The massing typologies are classified according to the topology of the geometry.

Their floor-floor heights, setbacks, step-backs, can be easily changed and it will update the geometry very fast.

This is intended to be connected to many parcels output by previous components.


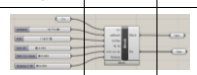




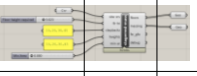








And we also intend to add analytical tools to measure what is visible to a person from a point or shading.

Then the time taken to generate becomes large.

That is why we provide separate components so that the user can judiciously use them and run many iterations in a workflow with multiple components.

It is hoped that our components will enable large-scale analysis.

## Slide 10


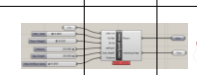













Fundamental Components for <i>Massing</i> or Typologies				
Name	Input	DOTS Component	Output	Illustration
 Extrude Block (based on FSR)				
 Staggered Massing (user defines its)				
 Urban Massing from setbacks, street-types & FSR				

The first 2 components illustrate simple extrusions.

In the second component, we propose that text panels be used where the required values are separated by a comma.

The third component is a special case where for each parcel, the closest street is found and required setback can be read and used for extrusion.

## Slide 11

Fundamental Components for <i>Massing</i> or Typologies				
Name	Input	DOTS Component	Output	Illustration
 Courtyard Massing				
 Staggered Courtyard Massing - Interpolated				
 Staggered Courtyard Massing - Normal				

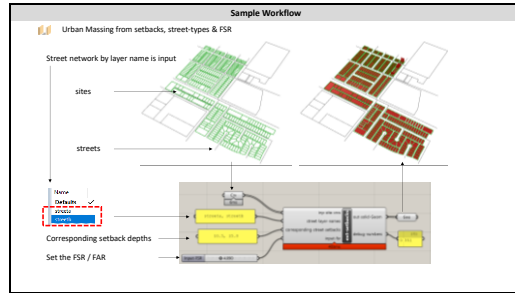
The courtyard typology is shown.

And the second and third components are shown where there is a base and towers can be generated from the upper curve.

These two are quite sensitive to inputs - generating the towers.

Explain some of the inputs.

## Slide 12



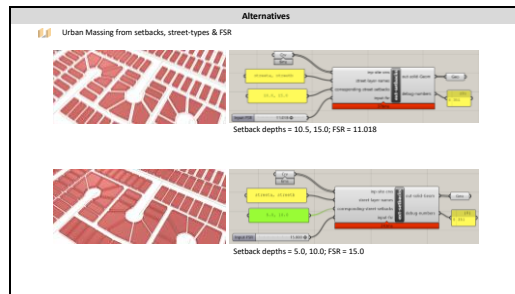
Using this component, a number of small parcels along a street network can be extruded based on setback for each street.

The street name separated by comma and setback distance separated by comma are input.

It references a Rhino3d layer for the street centerline geometry and checks which site is closest to what street. Then the corresponding setback value is used to determine the region which can be extruded.

Finally based on the FSR, the region is extruded and massing generated.

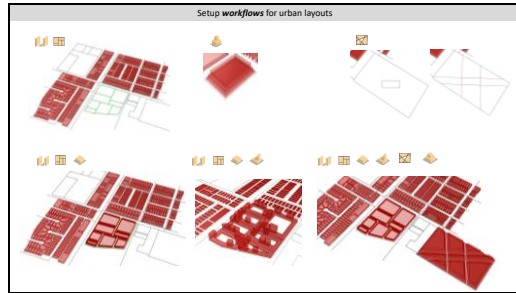
## Slide 13



Illustrations of possible variations are shown. The curve can be of any type and the numerical inputs changed at runtime to generate results, very quickly.



## Slide 14



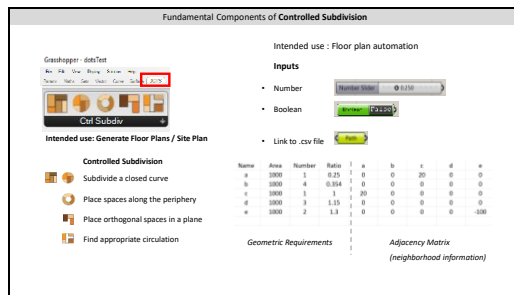
Dots components can help in setting up workflows for urban planning. Here the parcel-street-setback component is used to develop a context.

After that the large sites are broken up into smaller pieces and the massing components are used to generate a tentative solution.

The set up is trivial components can be changed to study various massing typologies.

Even the partitions generated can be changed in various ways including rotation, number of parcels required or type of partitions.

## Slide 15



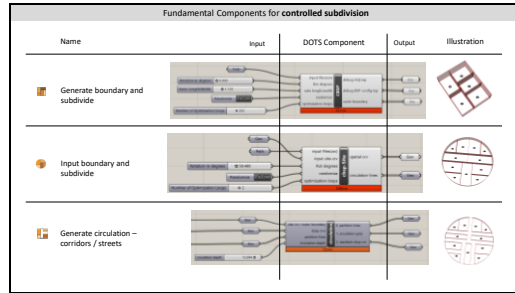
These components are similar to the previously shown components at the urban-scale but they are more sensitive to constraints where adjacencies and exact area can be plotted.

Typical floor plans may be generated from these but even planning problems can be solved.

The input .csv file is simple to understand- the first few columns provide dimensions and the last few cols are meant to provide information about neighbors.

Some components take-in an input curve but the boundary can be generated as well. The additional layer of logic forced us to separate the components.

## Slide 16

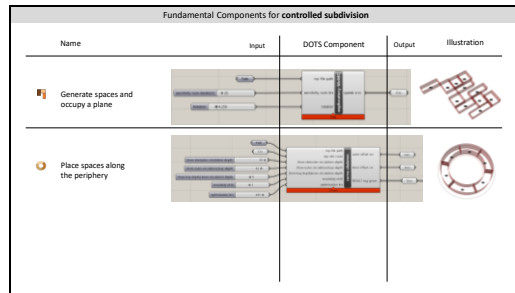


The first component generates an outer rectangle and places the spaces based on inputs provided. The aspect ratio of the outer rectangle and rotation can be input to check for various sites.

The second component inputs a site curve and places the spaces based on inputs provided.

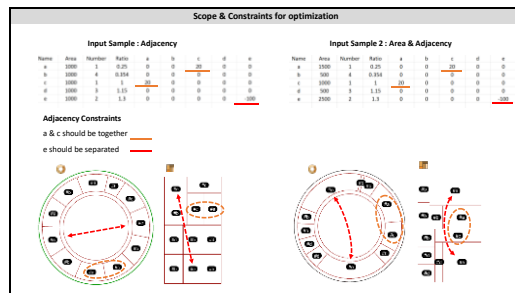
Circulation component accepts the spaces and provides circulation – corridor or streets

## Slide 17



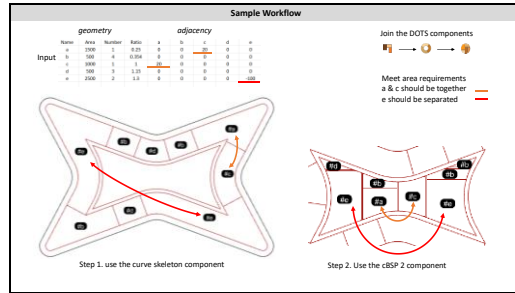
The first component here is exploratory – it places a space on the plane and tries to find the placement for the next best space. This way it proceeds until all spaces are plotted. The final component places the spaces along the periphery and shuffles them until all neighboring conditions are fulfilled.

## Slide 18



This slide shows how grouping is achieved. It considers the area required, number of each type of space, and the favorable neighbors. The excel sheet captures the adjacency values. +ve values denote attraction whereas -ve values capture the repulsive force. The numerical value is the intensity. In this case, a&c should attract each other whereas e should repel other e's.

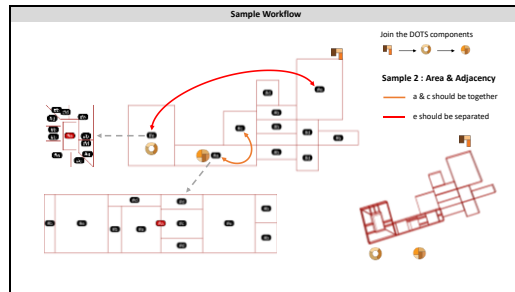
## Slide 19



To a given curve – say floor plate, apply a peripheral space generation then apply a subdivision component successively. The input to .csv file is simply a component which can be set to the required path by double-click.

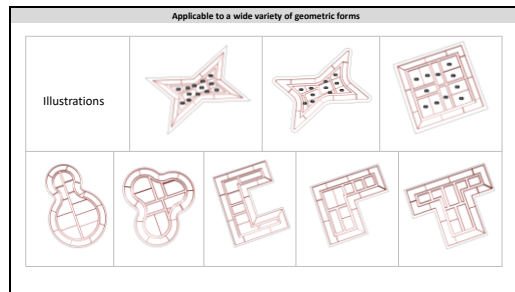
Once again, the same spreadsheet is used where a&c attract and e repels.

## Slide 20



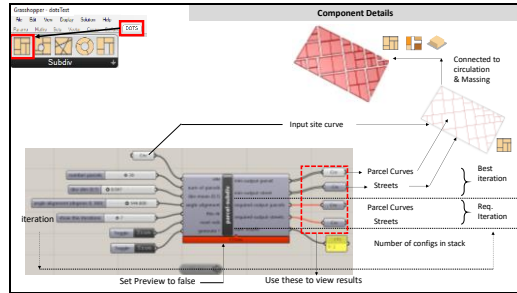
This shows three connected components with hierarchies of spatial structures. First, spaces are placed on the plane, then a particular space is occupied along the periphery and finally another space is subdivided internally. For each operation, area and neighborhood is considered.

## Slide 21

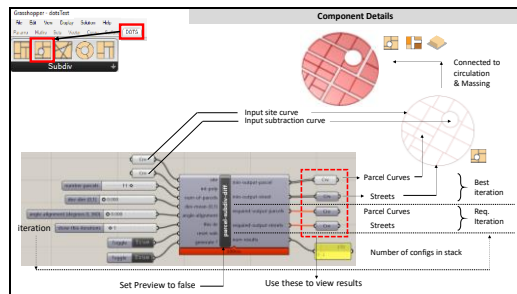


A vast range of curves were tested and it is known that sometimes the output will be very wrong But upon changing the parameters slightly, even by decimal values, appropriate results could be retrieved.

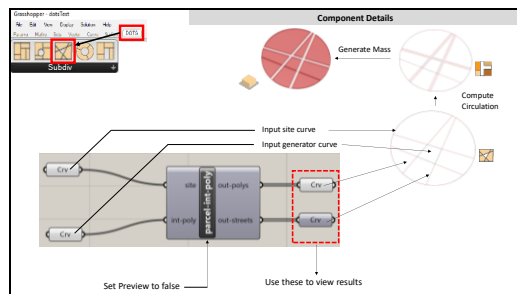
## Slide 22



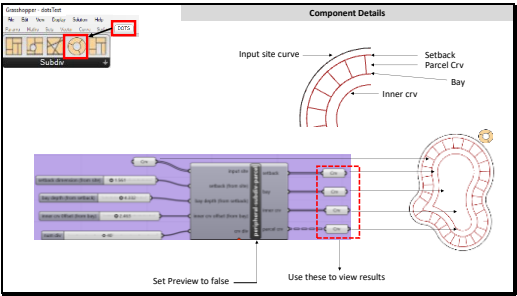
## Slide 23



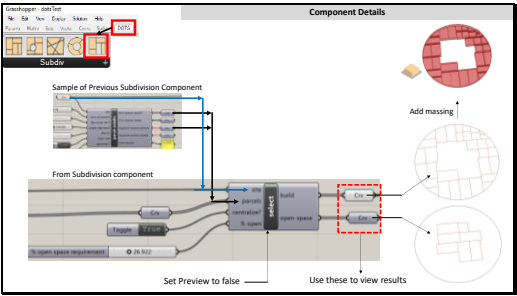
## Slide 24



Slide 25



Slide 26



Slide 27

