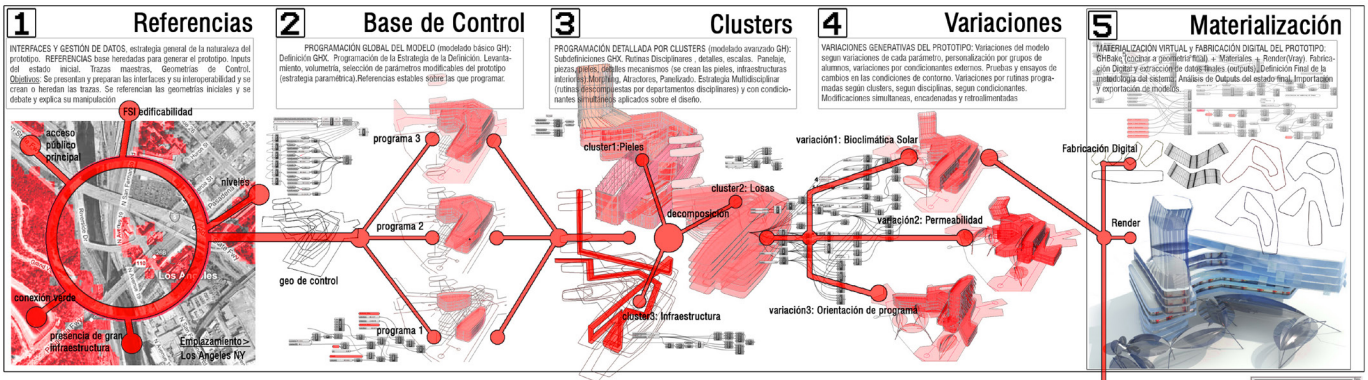
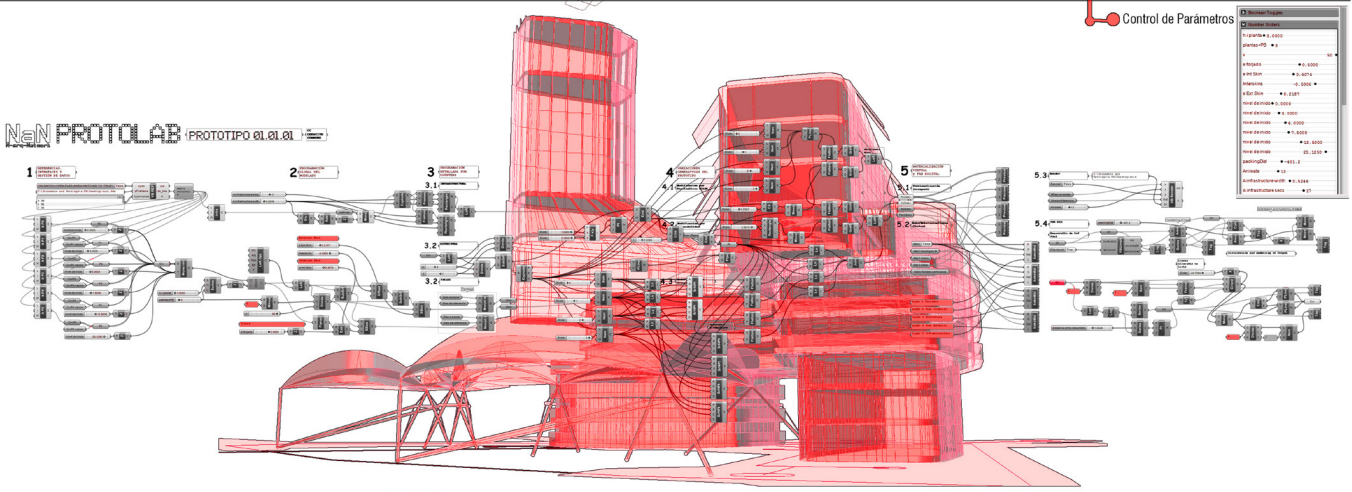
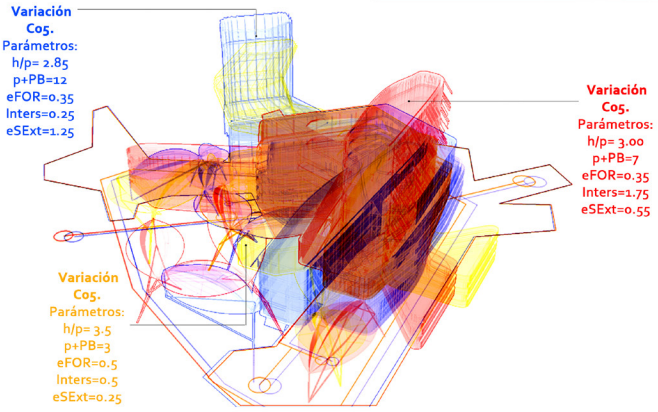
Curso

PVA

Programación Visual Aplicada a Arquitectura







Instructor:
Sergio del Castillo Tello.
Duración total: 20-40h.
Dependiendo de la extensión de cada bloque, el curso puede conformarse en tres seminarios

Imagen:
Elaboración de un esquema para programar un modelo tridimensional paramétrico.

1.-Referencias: Gestión de datos heredados como inputs del etsado inicial para las trazas maestras .

2.-Base de control: levantamiento, volumetría, selección de parámetros modificables.

3.-Clusters: aplicaciones y acciones como panelaje, morphing, atractores

4.-Variaciones: posibilidades que ofrece el modelo según cambio de las variables, personalizaciones, condicionates externos. Pruebas y ensayos de cambios en las condiciones de contorno (p.ej., dentro del subistema bioclimático, variaciones en las horas y mes de calculo de estudio)

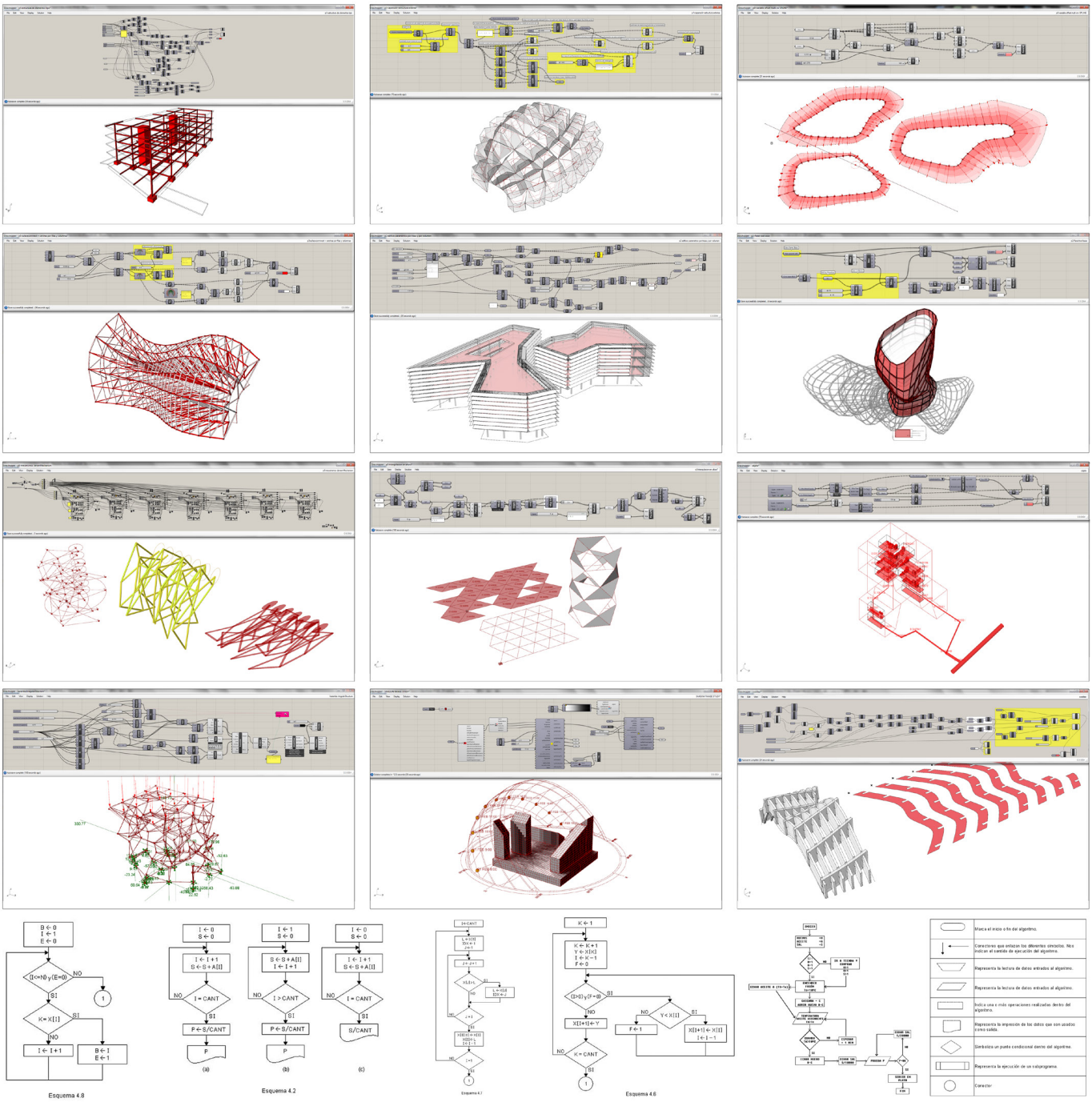
5.-Materialización: fijado del modelo con unos parámetros determiandos (preset). Cocinado (bake) de geometrias y datos resultantes por escenario

PVA / PROGRAMACIÓN VISUAL APLICADA A PROYECTOS

Objetivos docentes:

-Aprendizaje del ***entorno de programación visual*** para la generación de prototipos dinámicos de proyectos completos. Plataformas de programación basadas en nodos (*node-based*) para su gestión.
-***Diseño de algoritmos interrelacionados***. Planificar y explicitar procesos. Traducción de procesos a lenguajes de programación. Sintaxis básicas comunes entre todos los lenguajes de programación.
-Explorar tanto derivas como objetivos concretos. Programar ***herramientas de proyecto como parte del proyecto*** mismo. Explorar el proceso como esencia del proyecto.
-Incorporar al diseño datos externos al mismo. Aprender a programar, automatizar y después matizar decisiones. Generar ***proyectos adaptativos y reactivos*** en continua *reinformación*.
-Explorar los límites de lo codificado: producción de ***codigos como asistentes*** y no como imposiciones.
-Interrelacionar decisiones de equipos. Generar marcos y rutinas para el diseño colaborativo.
-Explorar ***topologías y prototipos***, entornos de incertidumbre y posibilidades. Manejo de bases de datos y flujos de herencia y transporte de datos.
- ***Generación dinámica, evolutiva y modificable***. Producción de herramientas de codigo abierto.

Herramientas y requisitos: Plataformas CG: Parametric SW (Grasshopper), CAD, BIM, SIG .
Grasshopper™ es un lenguaje de programación visual que corre dentro de la aplicación CAD Rhinoceros 3D. Los programas son creados arrastrando componentes en el área de trabajo. Los componentes tienen entradas y salidas (inputs y outputs), las salidas se conectan a las entradas de los componentes subsecuentes. Es utilizado principalmente para programar algoritmos generativos. Las programaciones pueden también contener otro tipo de algoritmos, tales como numéricos y textuales, audiovisuales, y aplicaciones hápticas.



Esquema 4.8 Esquema 4.2 Esquema 4.7 Esquema 4.6

Prácticas:
*Serie de representación mediante **diagramas de flujos** de algunos algoritmos básicos:*

a-Algoritmo básico de búsqueda

c-Algoritmo básico de cálculo de media o promedio

d-Algoritmo básico de ordenamiento de selección

e-Algoritmo básico de ordenamiento por inserción simple

f-Algoritmo básico del huevo frito (cómo hacer un huevo frito)

(Simbología de los nodos en los diagramas lógicos)

Matriz de definiciones .ghx y sus respectivos modelos dinámicos.

BLOQUE I. PRINCIPIOS DE APLICACIÓN PRÁCTICA DE LA PROGRAMACIÓN VISUAL

Objetivos:
Dominio del entorno de programación visual. Exploración de los principios esenciales de la codificación de algoritmos.

Procesos:
PV.1 IINTERFACES BASADAS EN NODOS Y USOS DE BASES DE DATOS MEDIANTE ALGORITMOS GENERATIVOS DE LAS CONDICIONES DEL PROYECTO. ASISTENTES Y ANALIZADORES.

Programación de estrategias de modelado. Introducción al algoritmo: recetas y rutinas. Traducción de algoritmos a lenguajes de programación visual. Rutinas y funciones. Elección de variables y objetivos a programar. Diseño mediante Tipos y Prototipos: diferencias entre modelos estáticos y dinámicos. Esquemas en diagramas de flujos. Código visual, selección de inputs y de outputs de la programación. Decisión de fases de diseño y de fases de acción y reacción del modelo dinámico.
Fases que dependen de otras fases: vinculación de datos entre fases del diseño.
Objetos de programación visual: Parámetros (gestores y almacenes de datos) , Componentes (generadores de datos) y cables (transportadores de datos); receptores y emisores.
Comandos en la plataforma 3D (Rhinoceros) y Componentes en Grasshopper, identificación.
Relaciones entre datos, objetos y geometría: datos en los que consta cada objeto.
Clases de objetos de programación, naturalezas y comportamiento. Reconversión o casting entre variables y entre clases de objetos de programación. Lectura de datos de archivos y lectura de datos entre plataformas: parsing o análisis pormenorizado de caracteres

Comportamiento de los Objetos de programacion visual: comportamiento de las cápsulas de parámetros y cápsulas de componentes (introducción directa, sobreescritura, herencia).
Almacenar, fabricar, grabar, limpiar, emitir, recibir e internalizar datos
Instancias de objetos de programación vs objetos *físicos*: modos de selección de objetos de programación en contraposición con los modos de selección de objetos *físicos.*
Gestión de datos persistentes, formulación y expresiones, datos por defecto
Herencia de datos: volátiles y persistentes. Sobreescritura y lectura.
Gestión de multiples cables y listas de datos. Almacenaje y apilamiento (orden) en listas.
Coincidencia y cruce de flujos de datos. Casting múltiple o conversión de naturalezas de dato.
Esquema jerárquico entre todas las clases posibles de datos en la interfaz: specials (timer,etc), brep, surface, mesh, curve, point, string(text), number(floating point), integer,colour,shader...
Lógica de confrontación de listas y árboles de datos según tipos de componente

PV.2 GESTIÓN DE DATOS MEDIANTE PROTOCOLOS DE SELECCIÓN DE PARÁMETROS DE CONTROL, FLUJOS DE TRABAJO Y OPERADORES A LO LARGO DE LOS FLUJOS

Listas y gestión de listas mediante componentes y mediante matices de parámetro.
Tipos de componentes escalares. Operaciones matemáticas básicas .Constantes, intervalos, polinomios, trigonometría, utilidades de evaluación. Operadores y funciones algebraicas.
Sentencias condicionales, booleanos. Disparadores de procesos, sentencias "if...Then"
Condiciones que disparan argumentos y condiciones que disparan otras condiciones.
Flujos de datos lineales o cíclicos. Bucles(loops): Recursividad (llamada a si mismo) e Iteración (repetición de instrucciones,condiciones de parada, *While*)

PV.3 GESTIÓN DE DATOS AVANZADA MEDIANTE CRUCES DE ÁRBOLES Y BASES DE DATOS (COMPLEX DATATREE MATCHING)

Listas de listas: matrices de datos almacenados. Rutas de rama y nombres de ruta .
Navegación entre tree-branch-item. Acceso a items a través de rutas, construcciónd e rutas
Uso mixto de listas y árboles de datos. Importar y exportar de bases de datos.
Datos entrelazados: mezclar datos y sobreescritura de datos según clases.
Parámetros y Variables almacenadas como árboles de datos. Expresiones que condicionan variables.Exportación e importación de datos. Flujos continuos entre plataformas. Ejemplos de usos concretos de almacenaje de la información del proyecto en bases de datos gestionadas mediante estructuras complejas de rutas (paths).

PV.4 MULTI-TRANSFORMACIONES E INTERACTUACION MASIVA ENTRE OBJETOS Y DATOS

Fuerzas de transformacion y deformación: vectores básicos, matrices, nubes de puntos
Manipuladores (handlers): manipulación punto/vector, utilidades de vectores
Objetos manipuladores de otros objetos mediante leyes programadas
Modificaciones adaptativas mediante atractores: campos de interactuación
Tipos de transformaciones y matrices: deformación del espacio

PV.5 ESPACIO PARAMÉTRICO INICIAL Y LA GESTIÓN MATEMÁTICA DE OBJETOS Y SUBOBJETOS

Espacio paramétrico unidimensional y bidimensional. Intervalos y parámetros *t* y *uv* .
Análisis de objetos complejos y gestión de los subelementos manteniendo estructura de datos. Reconstrucción, subdivisión y conexión de objetos de los espacios.
Operaciones sobre los objetos mediante gestión de sus parámetros internos: dominios, subdivisiones, segmentaciones. Módulos, medidas normalizadas y equilateralización.

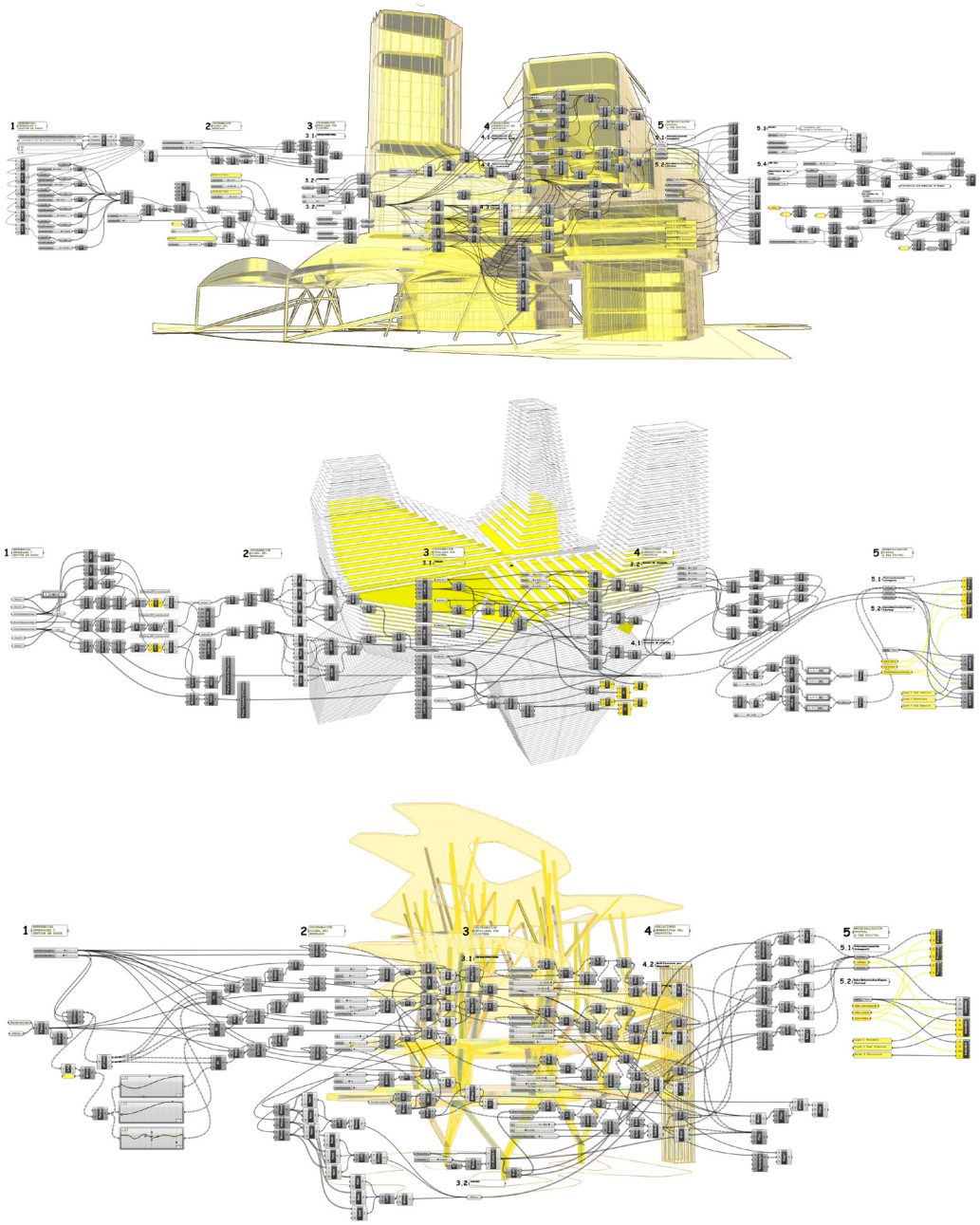
PV.6 ESPACIO PARAMÉTRICO AVANZADO PARA EL PANELIZADO COMPLEJO Y MATRICES DE OPERACIONES SOBRE GEOMETRÍAS

Herramientas de panelizado: panelizado uniforme vs no uniforme. Deformación por morfeado (morphing) entre cajas, superficies. Remapeo de objetos sobre otros objetos, repartir a lo largo de una superficie o polisuperficie. Proyecciones y presionado de objetos.
Panelizado deformado o normalizado. Operaciones de intersección booleanas complejas.

PV.7 ORGANIZACIÓN GLOBAL (HOLÍSTICA Y ESPECIALIZADA) DE LA PROGRAMACIÓN: ESTRUCTURA, ORDEN, OBJETIVO. MODELOS DINÁMICOS RESULTANTES.

Introducción a *clusters* o conjuntos acoplados de componentes. Interconexiones y ediciones.
Componentes de gestión de la programación: interruptores, disparadores, pausadores, temporizadores y ciclos . Flujos mixtos continuos y lineales en el entorno de programación.
Referenciación de clústeres. Reprogramación y lógica directa y difusa. Streaming de datos.
Introducción a componentes de scripting o programado textual. Multithreading.

Herramientas: Rhinoceros, Grasshopper



Prácticas:
Serie de prácticas de programación de prototipos (modelos dinámicos) de proyectos de arquitectura

a-Algoritmo de proyecto híbrido de usos en densidad

b-Algoritmo de proyecto de auditorios maclados conformados en un cuerpo único mediante barrido

c-Algoritmo de proyecto de plataformas perforadas soportadas por estructura ramificada arborescente

BLOQUE II. PROGRAMACIÓN VISUAL APLICADA SOBRE EL PROYECTO

Objetivos:
Aplicación de múltiples rutinas específicas sobre el proyecto concreto simultáneamente y en tiempo real. Proyectos "inteligentes": reactivos o adaptativos , sensibles a los cambios de las codiciones.

Procesos:

PV.8 ESTUDIOS DE CASOS DE PROYECTO SEGÚN MODOS DE GENERACIÓN DEL PROYECTO

Geometrías complejas adaptativas mediante deformaciones, morphing y form finding
Selección de tipos de proyectos según modo de generación. Naturalezas Nurbs / Mesh.
Explosión y reconstrucción de objetos para su deformación por datos externos.
Conversiones entre tipos de objeto y tipos de datos. Diagramas como input y output.
Topología y geometria: conexiones, voronoi, proximidad. Estructuras de interconexión de puntos y redes topológicas: deformación y reconstitución de nubes de puntos. Transferencias.

PV.9 PROGRAMACIÓN DE ELEMENTOS ARQUITECTÓNICOS. PERSONALIZACIÓN DE FAMILIAS BÁSICAS DE OBJETOS DINÁMICOS. REPROGRAMACIÓN DE PLANTILLAS DE PROTOTIPOS BASE.

Generación de terrenos según tipos, casos y necesidades: triangulaciones, contornos, alturas.
Muros. Capas, comportamiento, creación y despiece. Particiones de extrusión y proyección.
Forjados, soleras y cubierta. Capas, comportamiento y patrones de despiece.
Cerramientos. Dobles pieles,pieles reactivas y adaptativas. Atractores, permeabilidad.
Despieces. Sudivisiones y practicabilidad, cinemática, mecánica.
Fachadas tipo: Capas (hojas) y espesores. Offset variable y fachadas reactivas a orientación.

Comportamiento según rutinas específicas: soleamiento, distancias de seguridad, uso...
Estructuras espaciales. Triangulación, rutinas de interconexión entre puntos. *Tensegrity*.
Interconexión espacias entre diferentes ramas de datos y trenzado (weave).
Estructuras de entramado reticular y poligonales *n-gon (poligonos de n-lados)*
Programación de urbanismo paramétrico. Normativas e indicadores: FSI, OSR, GSI.
Extracción y cribado de series de variaciones sobre un mismo proyecto según rutina.
Programación de análisis de presupuestos y consumos de energía. Redes de transferencia.
Almacenaje de bibliotecas de datos en componentes su exportación.

PV.10 DOCUMENTACIÓN DEL PROCESO Y DE SUS RESULTADOS.

Datos de salida, display, outputs. Materializacion. Auto-referenciado. Renderizado. Animación de secuencias y procesos. Fórmula de la programación final: exportar trazados, datos y esquema.

PV.11 DISEÑO MULTIDISCIPLINAR MULTI-RUTINA

Estrategias colaborativas: clusters inter-referenciados. Sistemas de diseño multidisciplinar.
Aplicación de varias rutinas dentro de la misma programación: edición, injerto, reconexión.
Introducción de add-ons: secuencias de aplicación de componentes especializados.
Conexión en red simultánea sobre el proyecto. Interconexión con resto de plataformas.
Estudio de las **Aplicaciones (add-ons):**
Exportación, importación y referenciación entre diferentes plataformas (Horster,LocalCode)
Análisis bioclimático y energético (Ladybug y Geco Ecotect)
-LB Es un recurso libre y abierto (open source) de cálculo ambiental bioclimático ; Ejecuta la importación y analiza datos estándar meteorológicos; Dibuja diagramas como el recorrido solar, la rosa de viento, la rosa de radiación, etc.; Personaliza estos diagramas de varios modos: genera análisis controlado de radiación, estudios de la sombra, y análisis de visuales, simultaneamente al proyecto.
Diseño de estructuras y cálculo estructural (Ggym/Karamba)
-Es un programa interactivo, paramétrico de calculo estructural mediante elementos finitos. Esto permite analizar la respuesta de perfiles tridimensionales y estructuras de placa bajo cargas arbitrarias. Producción y dimensionado final del modelo con especificaciones de material y perfiles.
Modelado orgánico (T-Splines) Modelado conformativo y de *forma libre*
Propiedades físicas y mecanización (Kangaroo). Fuerzas y tensiones, gravedad y material.
Diseño evolutivo y resolucion de problemas de n-variables (Galapagos). Objetivo y genomas.
Realidad aumentada, ***conexiones remotas***, web, hojas de cálculo (Ghowl, Firefly)
Diseño de redes de instalaciones urbanas y del edificio (Worm):
-La aplicación consta de tantas herramientas como redes del edificio a programar: Instalaciones de Agua (Agua Caliente Sanitaria ACS, Agua Fría AF, Desagüe DS, Ventilación V, Energía E, Climatización CL, Protección contra incendios PCI, Energías Renovables ER).
Conectar con bases de datos específicas (Slingshot). Databases relacionales SQL.
Funciones y expresiones matemáticas científicas especificas (Mathlab-Mantis).
Flujo de trabajo y buenas prácticas en la programación de componentes personalizados u Objetos de Usuario (User Objects). Creación de herramientas personalizadas por proyecto.

PV.12 SALIDA DE LOS DATOS Y DOCUMENTOS DEL EDIFICIO (BIM) Y PREPARACIÓN PARA LA PRODUCCIÓN FÍSICA DE MODELOS (PRE-FABRICACIÓN DIGITAL)

Tipos de fabricación digital : aditiva , sustractiva, conformativa. Moldes y objetos finales.
Tipos de máquinas CAM (computer aided manufacturing). Impresión 3d. Aditiva por capas, aglomerado mediante laser y FDM (fused deposition modeling).
Cortadoras CNC (computer numerical control): cortadoras laser, fresadoras, por chorro de agua, por plasma y knife cutter.Preparación del modelo: análisis, transformación, generación de mallas, suavizado de mallas y aproximación de NURBS a mallas, exportar a formatos propios para la salida a fabricación digital, nubes de puntos ".Stl".Estrategias de fabricación (optimización y ahorro de material) Rutinas específicas por naturaleza de geometría y naturaleza de proceso de fabricado: Matriz de variables materiales y geometricas. Mecanismos: mecánica en diseño y mecánica en fabricación. Tipos de rutinas prácticas: costillas, nodos, barras, machiembrado, solapas, despliegue, origami, desarrollo, planarización, aproximación de curvaturas desarrollables, etiquetado.
Escaner 3D: registro de la forma tridimensional. Escaneo por contacto directo. Escaneos activos: escaneo por láser, por luz estructurada. Escaneos pasivos: a partir de fotos.
Mecanización de piezas finales: engranajes, pliegues, bisagras, rótulas.
BIM Building Information Modeling, principios y conversión a familias mediante add-ons
Producción automatizada de los documentos del edificio. Extensiones IFC.

Herramientas: RH, GH, Ecotect. GH add-ons:LocalCode, Ladybug, Geco, Karamba, GGym, T-splines, Kangaroo, Galapagos, Ghowl, Firefly, Worm, Slingshot. BIM: Visualarq, Grevit, Chameleon. Maquinaria CNC Fablab UPM Montegancedo.**Duración aproximada** 10-25h.

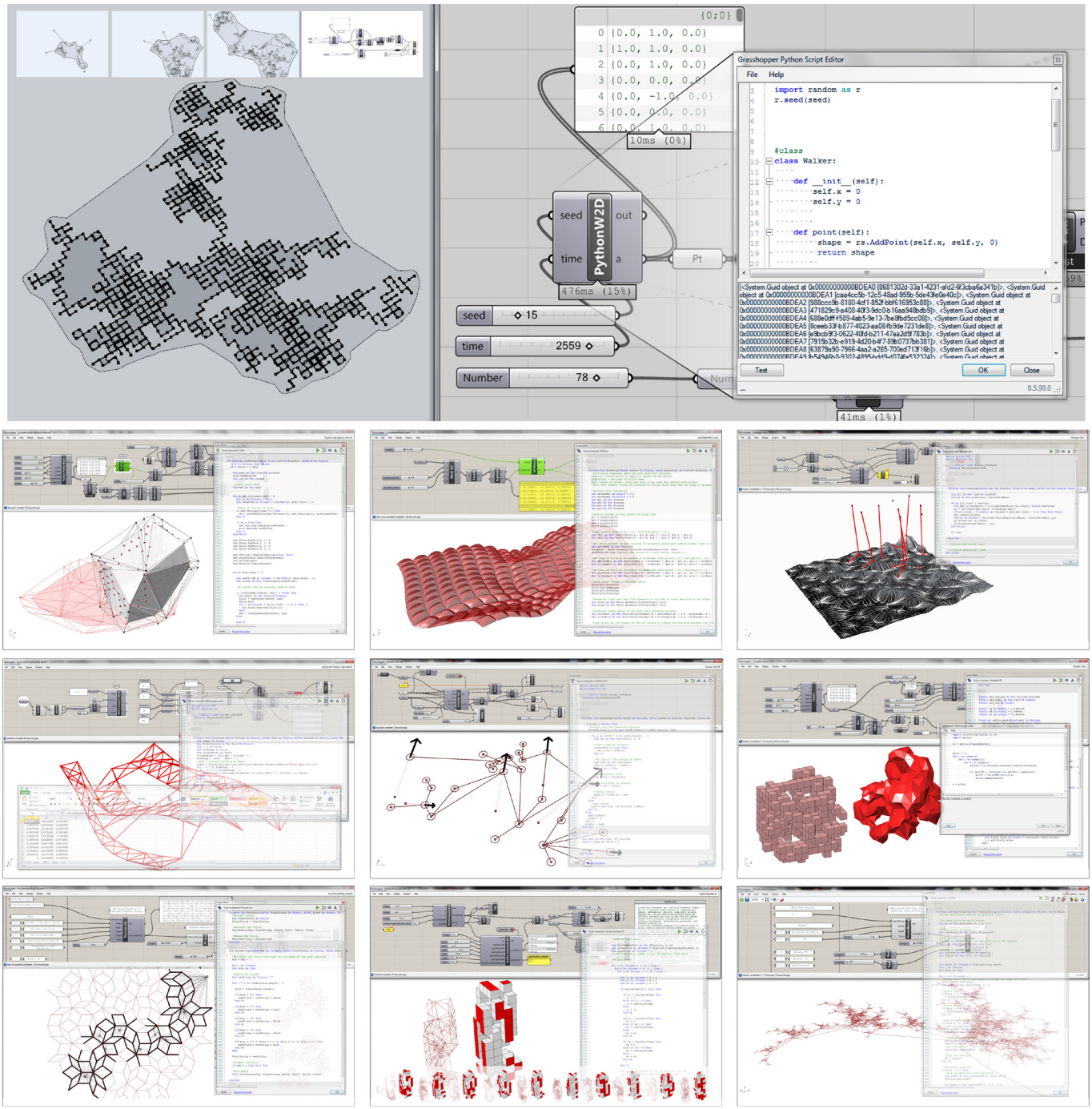


Imagen:
Ejemplo de programación de componente Python Script dentro de la interfaz de Grasshopper en lenguaje Python:
Componente "Walker" para generar caminos sucesivos mediante selección (modo random) de puntos en secuencia respecto al punto anterior(verNatureOfCode)

Matriz de prácticas de programacion con VB.net, C#, Python:

a1_Delaunay and 3dHull
a2_Surf to Mesh Pillows
a3_Drainage on Mesh

b1_Excel Read and Write
b2_Dynamic Point Cloud
b3_PerlinNoise3Dmetaballs

c1_ PenroseTiling
c2_CA CelularAutomata3D
c3_Branched L-systems

BLOQUE III. PRODUCCIÓN DE HERRAMIENTAS DE PROYECTO Y LENGUAJES DE PROGRAMACIÓN. *SCRIPTING* MEDIANTE PYTHON.

Objetivos:
Generación de componentes mediante codificado escrito. Aprendizaje de programación escrita, apertura del entorno de programación visual a los diferentes lenguajes; el caso de componentes Python.

Procesos:
PV.12 PROGRAMACIÓN DE COMPONENTES Introducción a la programación en lenguajes comunes: VB.Net, C#, Python. Sintaxis ,editores y uso. Funciones que se repiten en ciclo: recursividad y loops programados dentro de cada componente de script según lenguaje para establecer una comparativa. Jerarquía de objetos en Opennurbs y Rhinocommon. Programar en C#, Java y Processing. Programar en VB.Net: Utility Script Definitions, Excel Script Definitions, Generative Algorithms (Lindenmayer-System (L-System), Cellular Automata (CA)). Reglas de evolución y crecimiento.
PV.13 ESTRUCTURA DE CLASES DEL ENTORNO DE PROGRAMACIÓN,
Funciones, metodos. Constructores. Clases, objetos , instancias. Instancias constantes vs no-constantes . Representación de límites. Transformaciones de geometría .Funciones de utilidad global.
PV.14 PYTHON: interprete, tipos de datos, control de flujo y estructuras de datos.Control de errores y excepciones. Módulos, paquetes y objetos. Sintaxis y métodos comunes. Ejemplo aplicado: recorridos y crecimiento (Walker)
PV.15 VARIACIONES ,CATÁLOGOS Y TABLAS, PLANTILLAS DE PROGRAMACIÓN.
Herramientas: RH, Grasshopper, Monkey, Visual Studio,Eclipse, Python. **Duración aproximada**: 5-15h.