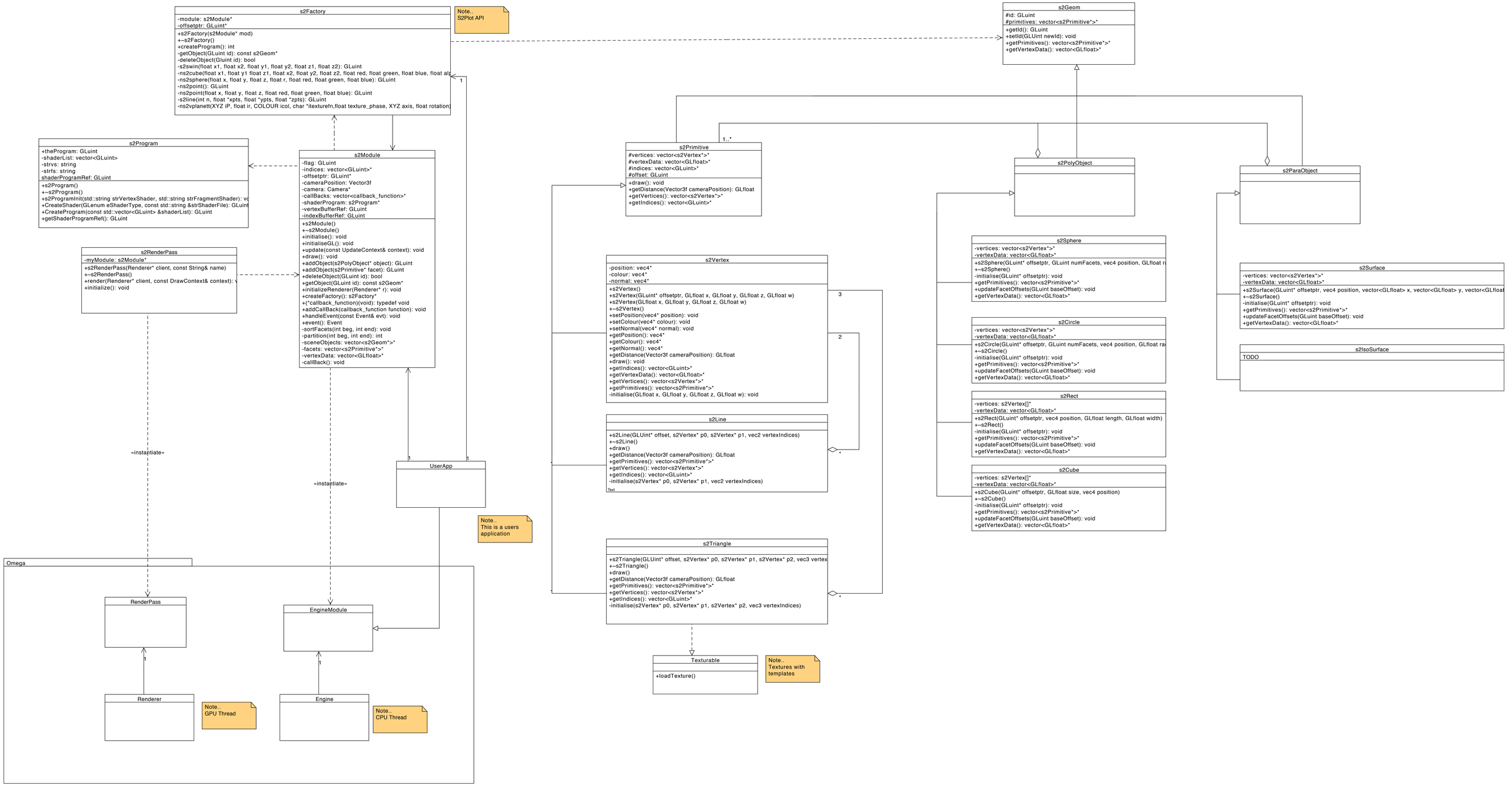
s2Factory
-module: s2Module*
-offsetptr: GLuint*
+s2Factory(s2Module* mod)
+~s2Factory()
+createProgram(): int
-getObject(GLuint id): const s2Geom*
-deleteObject(GLuint id): bool
-s2swin(float x1, float x2, float y1, float y2, float z1, float z2): GLuint
-ns2cube(float x1, float y1 float z1, float x2, float y2, float z2, float red, float green, float blue, float alp
-ns2sphere(float x, float y, float z, float r, float red, float green, float blue): GLuint
-ns2point(): GLuint
-ns2point(float x, float y, float z, float red, float green, float blue): GLuint
-s2line(int n, float *xpts, float *ypts, float *zpts): GLuint
-ns2vplanett(XYZ iP, float ir, COLOUR icol, char *itexturefn,float texture_phase, XYZ axis, float rotation)
Note..
S2Plot API
s2Geom
#id: GLuint
#primitives: vector<s2Primitive*>*
+getId(): GLuint
+setId(GLUint newId): void
+getPrimitives(): vector<s2Primitive*>*
+getVertexData(): vector<GLfloat>*
s2Program
+theProgram: GLuint
-shaderList: vector<GLuint>
-strvs: string
-strfs: string
shaderProgramRef: GLuint
+s2Program()
+~s2Program()
+s2ProgramInit(std::string strVertexShader, std::string strFragmentShader): v
+CreateShader(GLenum eShaderType, const std::string &strShaderFile): GLuint
+CreateProgram(const std::vector<GLuint> &shaderList): GLuint
+getShaderProgramRef(): GLuint
s2Module
-flag: GLuint
-indices: vector<GLuint>*
-offsetptr: GLuint*
-cameraPosition: Vector3f
-camera: Camera*
-callBacks: vector<callback_function>*
-shaderProgram: s2Program*
-vertexBufferRef: GLuint
-indexBufferRef: GLuint
+s2Module()
+~s2Module()
+initialise(): void
+initialiseGL(): void
+update(const UpdateContext& context): void
+draw(): void
+addObject(s2PolyObject* object): GLuint
+addObject(s2Primitive* facet): GLuint
+deleteObject(GLuint id): bool
+getObject(GLuint id): const s2Geom*
+initializeRenderer(Renderer* r): void
+createFactory(): s2Factory*
+(*callback_function)(void): typedef void
+addCallBack(callback_function function): void
+handleEvent(const Event& evt): void
+event(): Event
-sortFacets(int beg, int end): void
-partition(int beg, int end): int
-sceneObjects: vector<s2Geom*>*
-facets: vector<s2Primitive*>*
-vertexData: vector<GLfloat>*
-callBack(): void
s2RenderPass
-myModule: s2Module*
+s2RenderPass(Renderer* client, const String& name)
+~s2RenderPass()
+render(Renderer* client, const DrawContext& context): v
+initialize(): void
«instantiate»
«instantiate»
1
1
1
UserApp
Note..
This is a users
application
Omega
RenderPass
EngineModule
1
1
Renderer
Engine
Note..
GPU Thread
Note..
CPU Thread
s2Primitive
#vertices: vector<s2Vertex*>*
#vertexData: vector<GLfloat>*
#indices: vector<GLuint>*
#offset: GLuint
+draw(): void
+getDistance(Vector3f cameraPosition): GLfloat
+getVertices(): vector<s2Vertex*>*
+getIndices(): vector<GLuint>*
1..*
s2PolyObject
s2ParaObject
s2Vertex
-position: vec4*
-colour: vec4*
-normal: vec4*
+s2Vertex()
+s2Vertex(GLuint* offsetptr, GLfloat x, GLfloat y, GLfloat z, GLfloat w)
+s2Vertex(GLfloat x, GLfloat y, GLfloat z, GLfloat w)
+~s2Vertex()
+setPosition(vec4* position): void
+setColour(vec4* colour): void
+setNormal(vec4* normal): void
+getPosition(): vec4*
+getColour(): vec4*
+getNormal(): vec4*
+getDistance(Vector3f cameraPosition): GLfloat
+draw(): void
+getIndices(): vector<GLuint>*
+getVertexData(): vector<GLfloat>*
+getVertices(): vector<s2Vertex*>*
+getPrimitives(): vector<s2Primitive*>*
-initialise(GLfloat x, GLfloat y, GLfloat z, GLfloat w): void
3
2
s2Line
+s2Line(GLUint* offset, s2Vertex* p0, s2Vertex* p1, vec2 vertexIndices)
+~s2Line()
+draw()
+getDistance(Vector3f cameraPosition): GLfloat
+getPrimitives(): vector<s2Primitive*>*
+getVertices(): vector<s2Vertex*>*
+getIndices(): vector<GLuint>*
-initialise(s2Vertex* p0, s2Vertex* p1, vec2 vertexIndices)
Text
s2Triangle
+s2Triangle(GLUint* offset, s2Vertex* p0, s2Vertex* p1, s2Vertex* p2, vec3 vertex
+~s2Triangle()
+draw()
+getDistance(Vector3f cameraPosition): GLfloat
+getPrimitives(): vector<s2Primitive*>*
+getVertices(): vector<s2Vertex*>*
+getIndices(): vector<GLuint>*
-initialise(s2Vertex* p0, s2Vertex* p1, s2Vertex* p2, vec3 vertexIndices)
Texturable
+loadTexture()
Note..
Textures with
templates
s2Sphere
-vertices: vector<s2Vertex*>*
-vertexData: vector<GLfloat>*
+s2Sphere(GLuint* offsetptr, GLuint numFacets, vec4 position, GLfloat r
+~s2Sphere()
-initialise(GLuint* offsetptr): void
+getPrimitives(): vector<s2Primitive*>*
+updateFacetOffsets(GLuint baseOffset): void
+getVertexData(): vector<GLfloat>*
s2Circle
-vertices: vector<s2Vertex*>*
-vertexData: vector<GLfloat>*
+s2Circle(GLuint* offsetptr, GLuint numFacets, vec4 position, GLfloat ra
+~s2Circle()
-initialise(GLuint* offsetptr): void
+getPrimitives(): vector<s2Primitive*>*
+updateFacetOffsets(GLuint baseOffset): void
+getVertexData(): vector<GLfloat>*
s2Rect
-vertices: s2Vertex[]*
-vertexData: vector<GLfloat>*
+s2Rect(GLuint* offsetptr, vec4 position, GLfloat length, GLfloat width)
+~s2Rect()
-initialise(GLuint* offsetptr): void
+getPrimitives(): vector<s2Primitive*>*
+updateFacetOffsets(GLuint baseOffset): void
+getVertexData(): vector<GLfloat>*
s2Cube
-vertices: s2Vertex[]*
-vertexData: vector<GLfloat>*
+s2Cube(GLuint* offsetptr, GLfloat size, vec4 position)
+~s2Cube()
-initialise(GLuint* offsetptr): void
+getPrimitives(): vector<s2Primitive*>*
+updateFacetOffsets(GLuint baseOffset): void
+getVertexData(): vector<GLfloat>*
s2Surface
-vertices: vector<s2Vertex*>*
-vertexData: vector<GLfloat>*
+s2Surface(GLuint* offsetptr, vec4 position, vector<GLfloat> x, vector<GLfloat> y, vector<GLfloat
+~s2Surface()
-initialise(GLuint* offsetptr): void
+getPrimitives(): vector<s2Primitive*>*
+updateFacetOffsets(GLuint baseOffset): void
+getVertexData(): vector<GLfloat>*
s2IsoSurface
TODO