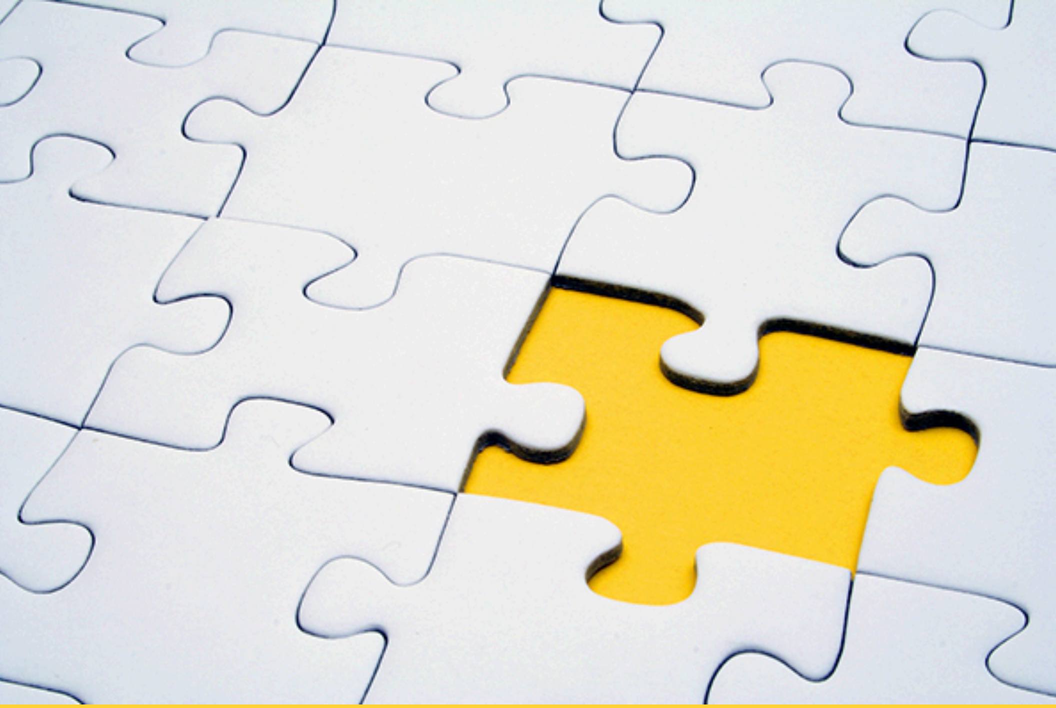


TANGERINE

TGN

a digital model
INTERACTIONS
format standard

TGN

a digital model

INTERACTIONS

format standard

Slogan-free! No slogans are written in this document, as far as I know. I made a real effort to avoid them. No banal ideas, just tools for getting work done.

\rob snyder
Helsingborg, Sweden
Linkedin Twitter
rob.snyder@tangerinefocus.com

[Tangerine] Makes Insight Tangible
Spatial Media Innovation and Cognitive Computing

To bring complexity into focus.
To make things clear.
To make oceans of information easier to understand.

Prologue **9**

Rigging for Insight **12**

Background: **16**

WORK 16

METHOD 16

PEOPLE 17

TGN Epics T0, T3, T4, T1, T2 .... **18**

TGN 0 [EPIC: TGN rig STRUCTURE] **20**

T0.1 TGN GUID 20

T0.2 TGN MODEL CALL 21

T0.3 TRE FILE 22

TGN 3 [EPIC: TGN rig Viewing] **27**

T3.1 TGN VOLUME 27

T3.2 TGN VIEWING ARC 28

T3.3 TGN UI "WHEEL" FOR SCROLLING ALONG TGN VIEWING ARC 32

T3.4 TGN CAMERA TRANSFORMS TO PARALLEL PROJECTION AT CENTER OF TGN VIEWING ARC 33

T3.5 TGN FADE AND GRAPHICAL TRANSFORMATIONS AT POSITION P2 OF TGN VIEWING ARC 34

T3.6 TGN SHOW/HIDE (FILTER) CONTROLS 36

T3.7 TGN MOTION CONTROLS 40

T3.8 TGN STYLE CONTROLS 40

T3.9 TGN ADDITIONAL GRAPHICS 41

T3.10 TGN GENERATE CUT GEOMETRY AT MODEL/TWIN ELEMENTS CUT BY TGN VOLUME..43

T3.11 TGN PROGRESSIVE DISPLAY ALONG VIEWING ARC.. 44

T3.12 TGN AUDIO..45

T3.13 TGN SOCIAL MEDIA 45

T3.14 TGN CONVENTIONAL DRAWING I/O (T2)...............46

T3.15 TGN LIVE OR RECORDED PLAYBACK.......................46

T3.16 TGN TRANSITION OPTIONS FROM ONE TGN TO ANOTHER TGN...47

T3.17 SHARING LINKS TO TGNS 49

T3.18 TGN OF TGNS (SUPER TGN)......................................50

TGN 4 [EPIC - TGN rig Authoring] 52

T4.1 TGN VOLUME DEFINITION..52

T4.2 TGN VIEWING ARC DEFINITION - AUTOMATIC AND ADJUSTABLE 52

T4.3 TGN UI “WHEEL” FOR SCROLLING ALONG TGN VIEWING ARC...54

T4.4 TGN CAMERA TRANSFORMS TO PARALLEL PROJECTION AT CENTER OF TGN VIEWING ARC.............54

T4.5 TGN FADE AND GRAPHICAL TRANSFORMATIONS AT POSITION P2 OF TGN VIEWING ARC 54

T4.6 TGN SHOW/HIDE CONTROLS 54

T4.7 TGN MOTION CONTROLS...54

T4.8 TGN STYLE CONTROLS...55

T4.9 ADD ANY EXTRA GRAPHICS ANYWHERE WITHIN PROGRESSIVE DISPLAY OF TGN ALONG ITS VIEWING ARC 55

T4.10 TGN GENERATE CUT GEOMETRY AT MODEL/TWIN ELEMENTS CUT BY TGN VOLUME....................................56

T4.11 TGN PROGRESSIVE DISPLAY ALONG VIEWING ARC.. 56

T4.12 TGN AUDIO..57

T4.13 TGN SOCIAL MEDIA 57

T4.14 TGN CONVENTIONAL DRAWING I/O (T2)...............57

T4.15 TGN LIVE OR RECORDED PLAYBACK.......................57

T4.16 TGN TRANSITION OPTIONS FROM ONE TGN TO ANOTHER TGN..58

T4.17 SHARING LINKS TO TGNS 58

T4.18 TGN OF TGNS (SUPER TGN).......................................58

TGN 2 [EPIC: TGN (P2) I/O to CAD] 60

T2.1 TGN I/O TO DGN/DWG/SVG.......................................60

T2.2 FLAT ARRAYS OF TGNS 61

TGN 1 [EPIC - Upgrade Legacy drawings to TGN] 66

T1.1 TGN UPGRADE OF LEGACY DRAWINGS PRODUCED BY CONVENTIONAL BIM (BUILDING INFORMATION MODEL) APPS..66

T1.2 TGN UPGRADE OF LEGACY DRAWINGS IN ANY FORM FROM ANY SOURCE 67

Conclusion 69

Phase 2 MACHINE LEARNING / AI COGNITIVE .. 70

Epilogue 71

THE LIMITS OF BIM 71

THE MECHANISM OF UNDERSTANDING 82

SO WHAT THEN? 87

About the Author 88

Prologue

This document is a software development specification. The purpose of the proposed development is to make things clear through interpretive interaction rigs, ***TGN rigs***, within modeled digital environments.

Improved mechanisms of interactive close study of digital models (including ***digital twins***), through ***TGN rigs,*** make user engagement with complex data more effective, more interactive, more clarifying, more communicative and expressive, and more revealing of insight. TGN might even bring the ***fun*** back into serious technical work by elevating the level of interpretive engagement in digital media.

TGN is a proposed model INTERACTIONS format standard specified by Tangerine. After specification writing this year, extending beyond my prior work on hypermodel drawing-model fusion, which is now over 10 years old and built into at least 7 commercial software platforms, I am sharing information about TGN with software companies individually, and with standards organizations. I intend to publish the TGN specification open source.

TGN is for everyone who wants to develop it. It's for any software companies, any developers, on any platform, open source and commercial. And we seek to incorporate it into standards endorsed by buildingSMART.org, if the idea catches on. And, it will catch on!

TGN: a model INTERACTIONS format standard: discussion and demonstration video playlist:

0 1 TGN: **rigging for insight** https://youtu.be/CGXrk9nGj0Y (2:16)
02 TGN: **what is TGN exactly?** https://youtu.be/byIW0T8MCsk (5:35)
03 TGN: **demonstration** https://youtu.be/wTh2AozTHDc (3:40)
04 TGN: **portability** https://youtu.be/Je859_cNvhQ (5:17)
05 TGN: **industry value** https://youtu.be/Ka0o1EnGtK4 (9:27)

(the dev platform I mention in the videos is iTwins.js, but TGN can be developed on every platform where TGN is useful and desired)

TGN = hypermodel + enhancements + portability + BCF

TGN includes the concept of hypermodel. I lead the team that proposed, designed, developed, and in 2012 first commercialized hypermodel. **But TGN is far richer than hypermodel in terms of graphical expression, user interactivity, communicative impact, and cross platform portability. TGN represents a major increase in user engagement in the close study of complex models.** And the TGN technical infrastructure makes the old hypermodel patents irrelevant and obsolete.

TGN actualizes drawing-model fusion in a more straightforward way. And built for more powerful user interaction, more effectively generating essential clarity about complex models, TGN leapfrogs again another decade ahead.

TGN has some similarity to the BCF format too, for model issue communications. But TGN is designed for model interpretation and clarification activities generally, fundamentally, not just in niche scenarios. TGN rigs, tuned by users, make models easier to interact with, interpret, make sense of, understand, and communicate about. And TGN rigs are not siloed within just any single app. The rigs are meant to be created and shared across all apps that support TGN.

Good ideas develop collaboratively. It just takes the spark of a group of people who say:

Yes. That makes sense. Now let's see how we can shape this!

TGN must develop at software companies willing to pioneer it. TGN includes and extends the 10 year old hypermodel concept and sets it on course for decades more innovation in sense-making clarifying interactive technique within modeled environments. With TGN integrated into models, complex data becomes more accessible and more useful to more people. With TGN integrated, models, apps, and platforms become even more valuable.

Rigging for Insight

Let's distinguish models from drawings. Models are spatial whole environments. Their medium is **mental** model, **physical** model, and more recently **digital** model. With computers and software, the digital medium has been added, not as a replacement for physical and mental models, but as a supplement, an assist. So what are drawings?

Drawings are fundamentally different from models:

Drawings are not wide, expansive, whole (and fuzzy) things like models are. On the contrary, drawings are acts of narrowing focus, articulating within a wider environment.

Drawings are attention-focusing technique. They narrow, from everywhere, to somewhere. They make clear something that matters, whatever it is that matters to the author. They do this in an economical and symbolic way. The ***form*** of drawing, the ***form*** of the medium itself, has been constrained, for centuries, for millennia, by limits that are both:

- fundamental to human perception *(drawings make sense to us, they're effective)*, and
- pragmatic *(we have paper, which is portable, and we have pencils, which make marks on paper, ...and so on)*.

Attention-focusing technique, the form of it, of drawing, remains stagnant, stuck where it had been. It remains where it was, and is, throughout 5 decades already of advancement in digital modeling. Modeling moves ahead. Drawing is stuck in the mud.

But no reason demands continued stasis. It's been stuck only due to inertia. There's no opposing force, which otherwise would have kicked it out of stasis already long ago. We might have expected this kicking force to be generated in the collective imagination of the software industry. But such imagination remains yet un-generated.

Perhaps TGN will generate it, kicking off a productive evolution in the ***form*** of the medium of drawing. The slope is downhill now anyway. It should be easy. Evolution in ***attention-focusing technique*** with complex environments is in fact eminent. Evolution is compelled as if by gravity, simply by the existence of the fertile rich medium of digital modeled environments.

TGN sheds some of the limitations of drawing's conventional form, but retains and elevates the principle ***function*** of drawing: the act of narrowing focus, showing something that matters, and showing it ***clearly***.

The mind isn't at work with models only. Cognition isn't happening when only environments (models) are present. The mind at work **IS** the mind engaged through ***attention-focusing technique*** within environments. One (environment or focus) without the other is meaningless.

Just one of the ***(environment / focus)*** pair, makes the sound of one hand clapping. Nothing is happening. No insight is generated. No understanding is gained. No work is being done. There's no mental activity. It's never one or the other. It's always both, environment and focus.

We can say something about the basic observable dynamic, of cognition:

> *Cognition is an interplay between wide and narrow, between environment and focus, between whole and part. Thinking happens, and understanding grows, in the back and forth, in the interplay.*

The assertion is often made, *that models replace drawing*. But that assertion is undermined by the basic observable dynamic of cognition. So we can abandon it. Let's say something more productive: a**s the medium of model evolves, the medium of attention-focusing technique** *(aka, "drawing")* **should also evolve.**

New software development should supply the technique, *the rigging*, so to speak, with which users may develop articulate acts of attention-focusing technique, ***closer looks,*** at things that matter, within digital environments. ***TGN rigs***, we think, will establish the ***focus pole*** of the **environment <> focus** ***interplay dynamic*** of thinking itself, thus providing a path from complexity to clarity, and from bewilderment to understanding.

It's in the interplay, after all, where thought happens and understanding grows.

ATTENTION-FOCUSING RIG (CAT) IN AN ENVIRONMENT

Background:

Work

Types of work in continuous use through an entire workflow:

W1	W2	W3	W4	W5
MODEL EVALUATION	MODEL AFFIRMATION	MODEL INTERPRETATION	MODEL QA/QC	MODEL / REALITY COMPARE
is model good enough? done? quality sufficient? what's missing? what's right? what's wrong?	what meets professional standard of care? where is standard met? where is it not? (necessary for model delivery)	what's going on? what is this? what does it mean? what am I supposed to understand and do?	W1 and W2 during construction	construction issue tracking

TGN makes work easier, faster, more effective.

Method

Methods for performing work:

M1	M2
MODEL INTERPRETATION TECHNIQUES	MODEL COMMUNICATIONS
These form a bond between model and user. By moving models from complexity to clarity, users make sense of models, understand them, make better use of them, and communicate better about them	formalized use of M1: deliverable model interpretive techniques, contained, linkable, shareable, persistent, referable, archivable, reviewed, affirmed, issued, signed, authorized

People

The people doing the work (W1, W2, W3...) using the methods (M1, M2):

Design Engineer, Architect	**PN 1**	design modeling and construction documentation services, e.g., electrical engineer, structural engineer, architect
Construction Manager (CM)	**PN 2**	General Contractor CM
Construction Trade (incl. fab/ shop)	**PN 3**	Construction trade contractor. e.g., electrical contractor, structural contractor
Construction Superintendent	**PN 4**	**r**esponsible for construction
Construction BIM Management	**PN 5**	construction BIM and Reality Model management services
Facility Manager	**PN 6**	post-construction, facility operations

These personas are from the building design and construction industry. But they're applicable to their equivalents in any of the physical asset industries, including:

- Building Design and Construction
- Energy, e.g., power plants (hydro, wind, gas etc.)
- Bridges, Road, and Rail
- Geotechnical

TGN Epics T0, T3, T4, T1, T2

Each TGN includes a scope box (view volume), a built-in viewing arc, a specified model filtering (what to show and hide, per category or search criteria), extra graphics, progressive display capability along the viewing arc, model element motion, and other features.

At the center of each TGN viewing arc, the viewing camera is transformed to parallel projection. Subsequently, the TGN looks like a conventional drawing.

TGN functions are grouped into five epics:

- T0: TGN rig Structure
- T1: Upgrade legacy drawings to TGN
- T2: TGN (p2) I/O to CAD and other graphics formats
- T3: TGN Viewing
- T4: TGN Authoring

They're ordered 0 through 4 for logical reasons but may be developed in a different order, T0, T3 and T4 first.

TGN 0 [EPIC: TGN rig STRUCTURE]

T0.1 TGN GUID

A TGN, uniquely identified by its GUID, is a discrete data container within a TRE file.. The TGN contains the T3.1/ T3.2 TGN rig definition including the TGN settings in the table below. Each TGN is assigned a GUID when authored (T4.1)

T0.0	**TGN name**	user named
T0.1	**TGN Guid**	937b444e-e3a5-45f5-b842-88679cd85884
T0.2	**TGN Model Call**	source location and model ID
T0.3	**TGN Container File Call**	Tangerine TRE file
T0.4	**TGN Author**	
T3.1	**TGN Volume**	scope box
T3.2	**TGN Viewing Arc**	arc, s-curve, other defaults, custom
T3.6	**TGN Filter Criteria**	filter by classification or filter by search criteria
T3.7	**TGN Motion (advanced)**	of volume, of elements (exploded, assembly sequence, etc.)
T3.8	**TGN Style Controls**	per element and per material standards, or library
T3.9	**TGN Extra Graphics**	examples:
		Townsend
		generate cut graphics
T3.11	**TGN Progressive Display Control**	
T3.12	**TGN Audio**	
T3.13	**TGN Social**	
T3.15	**TGN Live/Recorded**	

T3.16	TGN Transition Controls	
T3.17	TGN Share Controls	
T3.18	TGN of TGNs (Super TGN)	

T0.2 TGN Model Call

Each TGN makes a call to a model source and a set of models. A TGN calls a model source (user specifies the call). A model source may sit locally or in the cloud. A TGN provides source access, or a way to request access. Within a source, the TGN author identifies which models must be loaded for viewing that TGN.

In the simplest case, the models sit locally. The Tangerine app creates a TRE file (TGNs are discrete data containers within a TRE file). Models from a local source are loaded or linked into the TRE. The models and the TRE file may be merged together or unmerged. If unmerged, then the TRE and the model files travel as companions.

For old-school users, small shops who like to manage files manually, those who don't have and don’t need access to managed environments, for them, local models, local TREs with TGNs making model calls locally, are supported.

In managed enterprise workflows, model sourcing is managed. TGN model calls point to a managed environment model source. Likewise, workflow management systems may control access to TRE files and the TGNs within them. For enterprise users, controlled access to cloud based model sources, and managed access to TREs and the TGNs within them, are supported. The TGN SDK may provide managed workflow access functionality to assist TGN feature implementation in partner apps that lack these functions.

The Tangerine app may make use of managed workflow access functionality that exists already in the Bentley iTwins dev platform. For enterprise users, the TGN SDK may provide access to the managed workflow functionality that exists already in the iTwins dev platform. Of course the target app may have completely different managed workflow system requirements. So, the target app developer will tune this as required for using other managed systems instead of those provided in the iTwins platform.

TO.3 TRE File

TGNs “grow” on trees (in .tre files). A .tre file contains a collection of TGNs. Each TGN stores the name and server source of the TRE file.

TGNs are data containers hosted within .tre files. Any app that writes any file type that supports modeling, reality modeling, or digital twins, may act as a host for TGNs, and .tre files, if TGN SDKs are implemented.

TRE files per discipline (examples):

- TGN Structural TRE
- TGN Mechanical TRE
- TGN Architectural TRE
- TGN Curtain Wall Fabrication TRE
- TGN Electrical Fab-Install TRE
- TGN Site-Civil TRE
- TGN Site Utilities TRE
- TGN Subsurface TRE

a TRE (file) contains:

- **TGN**s: at least one, and as many as needed (tens, hundreds, thousands)
- a *tree view* (UI panel) listing all TGNs stored in the TRE
- Links (references) to digital twins and other models
  - TREs "grow" out of this "ground" (twins and models are the ground)
  - TGNs (many) *grow* on TREs

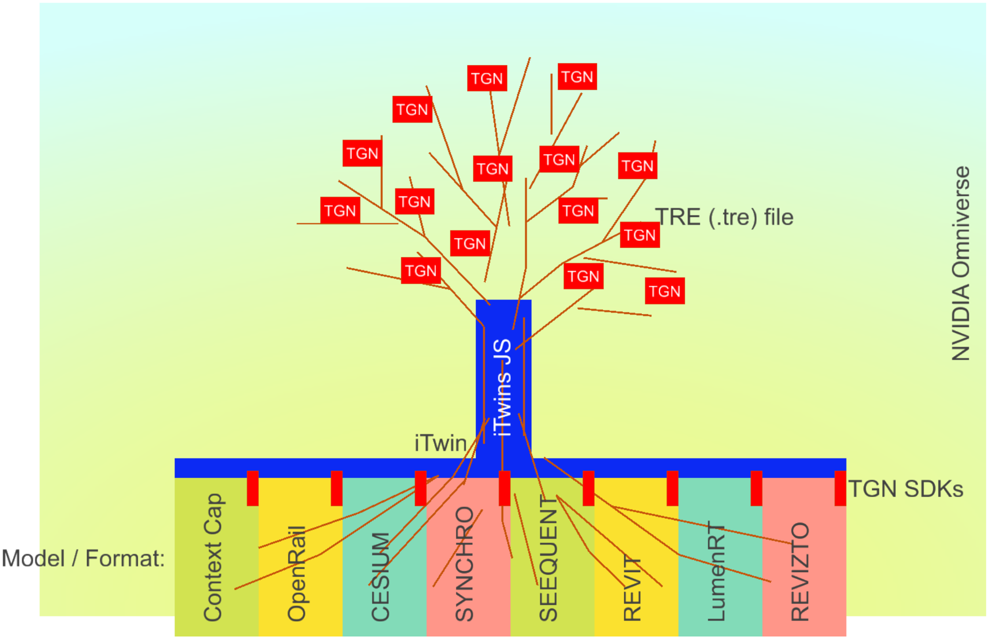
TREs support Multi-user access and edit

Different users may edit and view different TGNs at the same time within the same TRE file.

Share a TGN outside of its home TRE, for expression in another TGN-supporting app:

Where the TGN is expressed in another app, the TGN calls its home TRE (in the cloud or on the network), findsthe TRE's linked models/twins, and loads them. See video 03 TGN: **demonstration** https://youtu.be/wTh2AozTHDc (3:40)

- The call to home TRE enables checks for TGN updates *(at both the TGN and TRE level)*
- TRE-linked twins/models can be compared to the twins/models in the host target app (element GUID checks)
- Can share a single TGN, a set of TGNs, or an entire TGN Tree



TGNs are shareable across model formats

Caveats

Developers of apps that implement TGN SDKs may either implement the TRE concept, or use their own file format as the TGN host directly, omitting TRE. There will be advantages to implementing the TRE concept as an overlay over your app's native format, but depending on your app scenario, TGNs may be picked from the Tangerine TRE and transplanted into your app without need for implementing the TRE concept.

A TGN is a data container which, when implemented in an application or app via the TGN SDK, becomes a new data container concept within the host app. This container may be used in FILTERING (see T3.6 and T3.9)

But, the TGN container concept, while useful as a filtering parameter, may conflict with other filtering concepts (e.g.: DWG layers) in some applications. Care must be taken in TGN implementation such that use of the TGN container as a filter parameter is compatible with other filter parameters used in your app.

For example: Extra graphics of a TGN (T3.9) might be drawn on common layers in your app, say on layer X. If multiple TGNs use layer X for T3.9 graphics, and a TGN when activated turns on layer X, the additional filter parameter (THIS TGN) should be used to show only entities on layer X that are part of THIS TGN. However, developing your app to do that incorrectly may in fact break your file format standard (e.g.: dwg).

We do not have that limitation in the Tangerine App, because we can do it without breaking the TGN format standard (because there is none; we're creating a new standard). But you might have that limitation in your app, if your app is AutoCAD or AutoCAD-like. Solutions may exist for this in DWG applications. But if not, then those apps may not be suitable hosts for TGN implementation. Many applications do not have this limitation because many applications (Revizto, Unity, xeokit, web model viewers, NVIDIA OMNIVERSE.... ARE NOT BUILT IN DWG FORMAT) <<<< so these have no such problem.

MicroStation avoids the problem because although it has levels (layers): it can implement each TGN in it's own "model" (DGN's equivalent to "DWG's "modelspace"). Unlike AutoCAD, MicroStation allows an unlimited number of models (modelspaces) in each DGN file. This way, multiple TGNs in a DGN file eac sit in their own DGN model in the same DGN file, with level (layer) controls therefore isolated / independent and so not in conflict between multiple TGNs: extra graphics may be drawn in multiple TGNs using the same level / layer and yet may still be turned on or off correctly per TGN.

In summary, this kind of legacy format limitation in fact may argue in favor of implementation of the TGN host file format, the T0.3 TRE file from the Tangerine app, as an additional format that your app supports via [import / reference / xref / link]. This way your app supports TGN functionality, without deviating from your established file format standard.

TGN 3 [EPIC: TGN rig Viewing]

T3 is the Viewing side of TGN. T4 is TGN Authoring.

T3.1 TGN Volume

The very first, originating aspect of any TGN is that it embodies *the act of narrowing down*. See video 03 TGN: **demonstration** https://youtu.be/wTh2AozTHDc (3:40)

Within the ocean of information and geometry that is a model or digital twin, …that is, within the **WIDE**, expanse of an entire environment of information, the user **NARROWS** down for focus. That focusing occurs *somewhere* and involves some subset of entities less than the whole set of entities comprising the twin (model).

So the first question is, ***where*** is a given TGN focusing? Well, typically, as is also the case with conventional drawing, the "where" is defined as a volume, a definitive volume with boundaries, usually (almost always) smaller than the entire range of the environment of the model (twin).

The TGN volume can be defined in any convenient manner using any of the best techniques available in the computer graphics world for doing so. Tangerine has no preference (meaning, Tangerine's founder, Rob Snyder has no preference) for which kind of UI/UX user interaction defines this volume. Any typical method for volume definition well known in the engineering digital modeling industry, or innovative methods perhaps from the gaming industry, or the CGI industry, are welcome. What matters only is that the volume be defined, in a way that's clear for the user, and easy to perform. TGN Volumes of course will be editable *(push/pull on boundaries, or move whole volume)*.

Simple rectangular volumes will be supported first. More complex volume geometries can be added later.

As a hint or reminder, volumes within models are, in engineering apps, defined in conjunction with symbols like these, which specify the length and width of a focusing volume, with the volume's depth implied (by convention), and adjusted in 3D by the user:

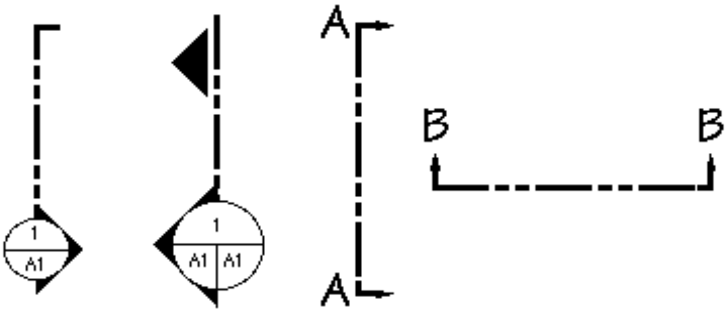


Figure 1: narrowing volume symbols (typ.: "drawing callouts")

Naturally, definition of ***where to look***, within a digital twin (model), may come from means other than the user explicitly defining the volume boundaries. TGN volume boundary can be derived indirectly as a result of properties search or other filtering that may originate TGN creation automatically defined to include the range of entities passing the filter. *(more about filtering in T3.6)*.

T3.2 TGN Viewing Arc

Here is the first distinction between a TGN model interpretation technique, and a conventional drawing. A conventional drawing represents the contents of the T3.1 volume, from a viewing orientation that is *normal* (facing, perpendicular) to the *front face* of the volume, with the graphics flattened onto that face. While such conventions are in fact useful, the scope of TGN is larger, and includes the convention as a subset of a viewing experience that extends beyond the limitation.

Built into each TGN is a 3-dimensional **TGN viewing arc**:

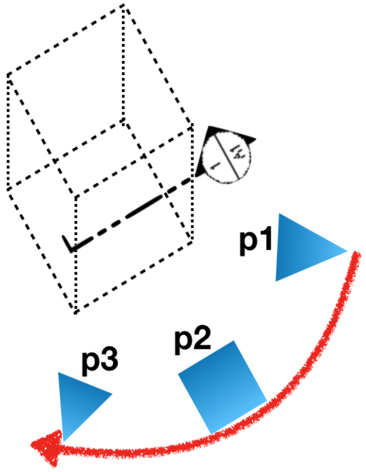


Figure 2: built-in TGN Viewing Arc

A viewing arc is automatically generated based on the proportions of the T3.1 TGN Volume. The TGN is viewed from the orientation of a camera that moves (T3.3) along the arc. The default starting position when viewing a TGN is at position 1 (P1) in Figure 2. The blue triangle at P1 represents a camera in perspective projection. The viewing "eye" sits on the red arc. The eye moves along the arc toward position 2 (P2) where the camera transforms to parallel projection, *hence the square*.

Although the viewing arc is automatically created, controls will be given to the user for adjusting the arc to suit the TGN author's preference. See video 03 TGN: **demonstration** https://youtu.be/wTh2AozTHDc (3:40) The viewing path, once adjusted for optimal clarity (to best communicate what is being shown at the TGN) is stored in the TGN. Thereafter, anyone interacting with the TGN, viewing it, will view it along its built-in viewing path. Of course one is free to leave the path to view the environment *(with or without the TGN narrowing)* from any position, but the TGN author has control over the built-in default TGN viewing path.

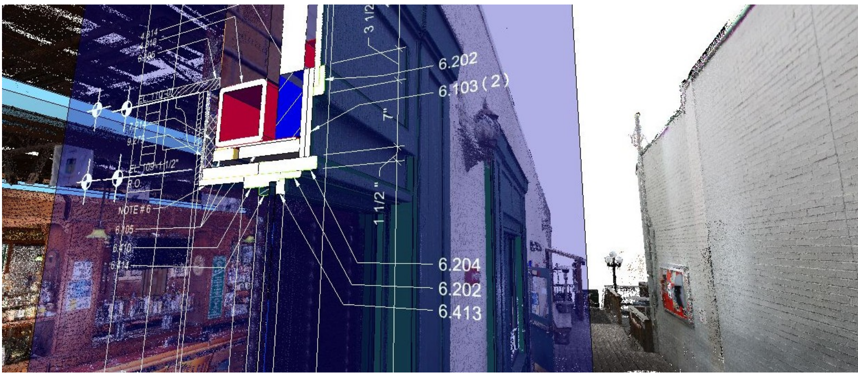


Figure 3: a snapshot from a position along a TGN's Viewing Arc

The image above, Figure 3, is a snapshot of a TGN, viewed from a position between P1 and P2 along its built-in viewing arc (the red curve below in Figure 4, with the camera at the position shown by the blue wedge). Figure 3 is zoomed and cropped such that only a portion of the TGN is shown.

Figure 4 shows the full extent of this TGN, as viewed from position P2 of the TGN's viewing path. As mentioned, the TGN viewing camera is transformed from perspective to parallel projection at P2, resulting in a presentation that looks like a conventional drawing. The transition from perspective to parallel happens in a zone around P2, starting right after the snapshot in Fig. 3.

While the camera eye transforms along the red curve, the camera target moves along the blue dashed line. Eye and target paths are created automatically by the TGN (based

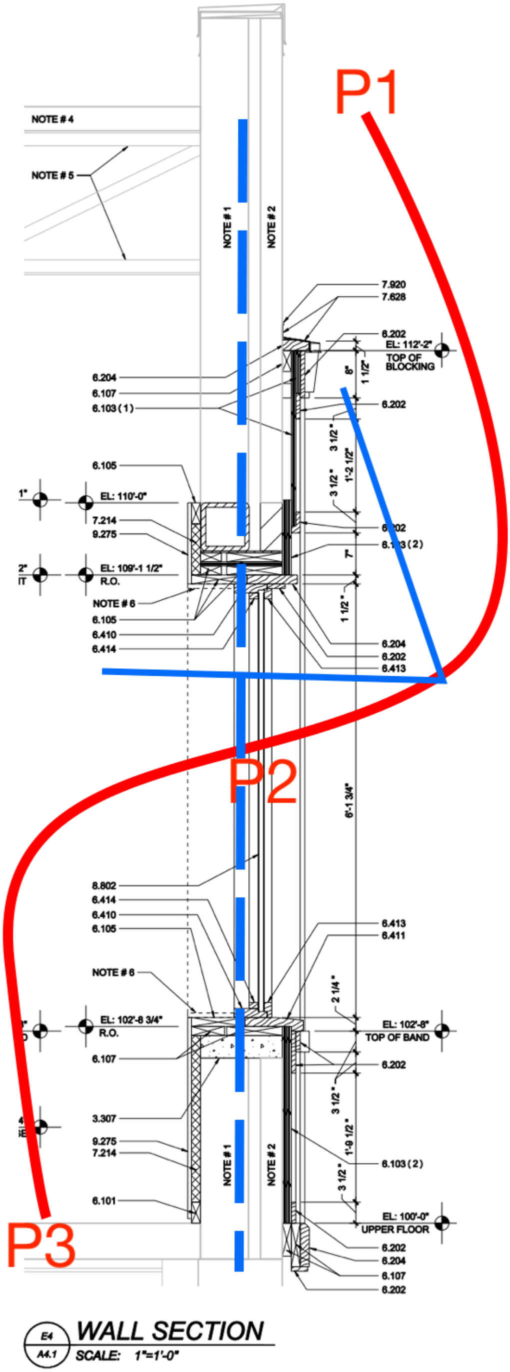


Figure 4: TGN at viewing position P2, showing its TGN viewing path in red

on the proportions of the TGN volume). Preset path options can be offered to the TGN author who may also adjust the paths and timings manually. The default presets will be designed to do the job well most of the time without manual adjustment.

This TGN, from P2 or other positions, may be presented in a document, such as this one, where it is the 4th Wall Section, on the right (number E4).

Note that this is just one type of TGN. TGNs are adaptable to many kinds of visualization, including for example, ***in-situ*** *piping and instrumentation diagrams (P&IDs), whose appearance, form, and generative origin, are very different from the example in Figures 3 and 4. However, TGN is perfectly suited to P&ID expression (ask me about this), and many other forms of expression.*

T3.3 TGN UI "wheel" for scrolling along TGN viewing arc

To move back and forth along the TGN viewing path, we use a mouse scroll wheel. Or, another UI device as appropriate for touch screens, touch pads, and other devices. Those of us old enough know what Figure 5 is, a 1970s stereo amplifier/tuner with weighted and dampened belt-drive mechanisms that gave an inertial effect on the controls, and not without a significant amount of delight given by that mechanism to those playing around with the controls.

This is an appropriate metaphor for TGN viewing, allowing the user to control view position along a TGN viewing arc. Back and forth as is required for close study, for inspection, contemplation, interpretive effect, for understanding what's communicated by the TGN, from P1 to P2 to P3, and back, at whatever speed along the path the user prefers, and controls by rolling the T3.3 TGN scrolling wheel.

Figure 5: Metaphor for TGN viewing position control, rolling back and forth along TGN viewing arc

T3.4 TGN Camera transforms to parallel projection at center of TGN viewing arc

As mentioned previously in T3.2, the TGN camera defaults to perspective projection but transforms to parallel projection at position P2 (see Figure 2). The result is a viewing orientation that gives the appearance of a conventional drawing.

T3.5 TGN Fade and graphical transformations at position P2 of TGN Viewing Arc

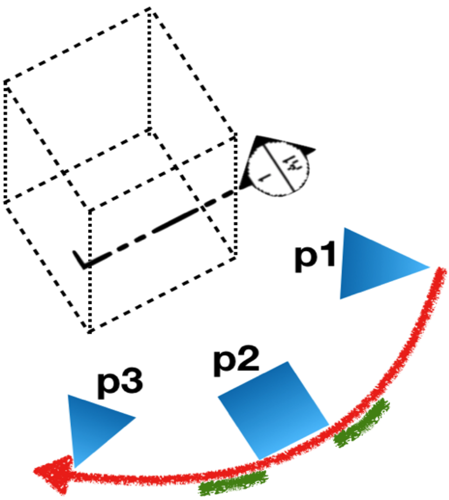


Figure 6: Graphical Fade and Transition Zone (Green, near P2)

As the TGN user (viewer or author) approaches position P2, in the green areas near P2, while moving along the TGN view arc, graphical transformations are invoked by the TGN in addition to the (T3.4) camera transformation. The possibilities here are diverse. The TGN author may set multiple preset graphical transformation options to be stored in the TGN. These can include anything possible with digital graphics, such as:

- Set background to white, with graphics in high contrast black, , e.g., Figure 4

- Preserve digital twin (model) textures/colors, and present added graphics (T3.9) to high contrast white on translucent background, e.g., Figure 7 below.
- Any other options authored and stored in the TGN, as described in other T3 subsections. See T3.6 (show/hide) and T3.8 (style) for example.
- The TGN user (viewer) will experience (see) any of the stored transformations preset by the author simply by inspecting the TGN while rolling the TGN scrolling wheel. The graphical transformations are applied progressively as the user approaches P2 and exits P2, in the green areas in Figure 6.
- A delay/pause, built-in, precisely at position P2 may enable the automated use of a simple, common, graphics FADE technique to good effect (fading from model style, to drawing style, for example).

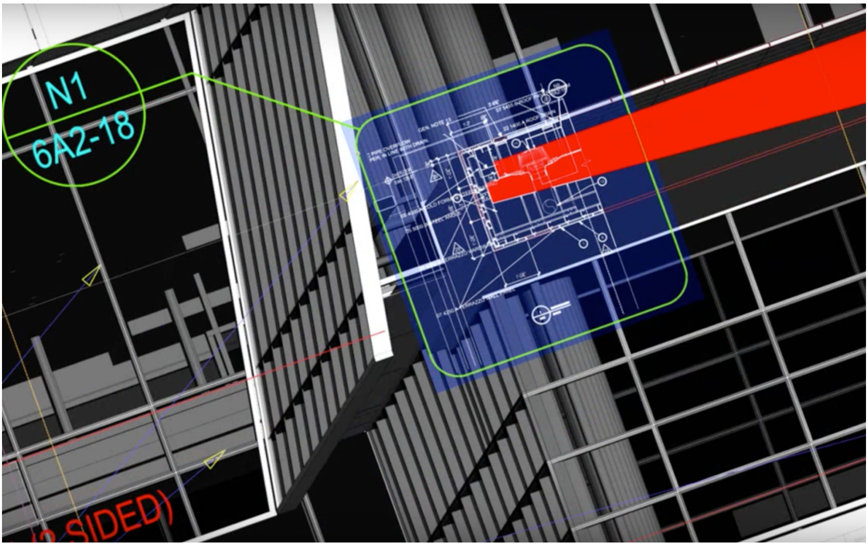


Figure 7: Another TGN example, viewer approaches position P2, along TGN viewing arc.

T3.6 TGN Show/Hide (Filter) controls

What's visible and not visible within a TGN is a function of:

A. the entire scope of the relevant models, digital twins or ordinary models, both vector and point cloud types *(with or without photo constellations)*, or a hybrid of both, and

B. ***Limits*** put on this scope, defined by the TGN author

It's the limits (B) that we're concerned with here, ***show/ hide controls***. We can show everything in (A), no limits, or we can show some subset of (A). To show a subset we have to define limits (B). Essentially (and probably in totality), this is a ***FILTERING*** function.

Within a TGN Volume (T3.1), we need to show some entities and hide others using Filtering criteria. For this we're going to leverage:

1. the filtering controls that are perhaps built-into the relevant development platform, e.g., the Bentley iTwin.js model handler: UI Trees

2. The categorization and properties structure of the loaded/present model (twin) data

No Industry Standard Data Structure

A common industry problem is that 2 (above) is unpredictable. There are multiple competing standard data structures in the industry, all of them are in flux, and none of them are utilized consistently, neither by app developers nor users. Therefore it is impossible to predict what the data structure will look like, or what it will consist of.

That's OK. TGN filtering will provide filtering controls over top of whatever data structure exists in the loaded models. And, TGN filtering controls will be developed using inspiration not from poorly developed examples, but rather from the best (that I know of) implementation of model filtering and criteria search in the industry. This is found in Revizto:

- Revizto Filtering, Search, tree functions, and graphics options: Revizto Video Demo

TGN Filter = Named Criteria Search

With search power of the kind demonstrated by Revizto, our TGN Filter is a saved, named Search. We may be searching for class, category, type or various property criteria, and a combination of all these and others. The result of a search becomes our TGN Filter when we name and save the search. With a TGN Filter, we control the show/hide display state of model(twin) entities within a TGN Volume (T3.1).

We say:

*"**this** TGN" filters:*
*"**these** models/iTwins"*

to show:

*"**these** entities that **pass** our Filter based on this class/types/properties search criteria"*

and hide:

*"**those** entities that **fail** our Filter based on this class/types/properties search criteria"*

TGN Portability

We take whatever data structure we're given, that the loaded models provide, and within a TGN we tag model entities with show/hide (and style, T3.8) tags. We're flexible. We can do the tagging against any data structure we're dealing with. We record our search criteria and Filter name, and the entity tags that result from the Filter, in the TGN.

Our intention is that the TGN is portable, that it can be viewed correctly wherever the TGN is shared, in any app/ platform that integrates the Tangerine SDKs.

Portability Scenarios

This portability concept involves two scenarios:

1. The TGN is authored in one app (like Catenda), and viewed in another app (for example, Revizto, or NVIDIA OMNIVERSE). And the TGN and the underlying models against which the TGN was authored, are both loaded in the target viewing app.
2. The TGN is authored in one app (like Catenda), and viewed in another app (for example, Revizto or NVIDIA OMNIVERSE). And ONLY the TGN and **not** the

underlying model against which the TGN was authored, is loaded in the target viewing app, because the target app maintains its own version of the model in another format.

Both of these TGN portability scenarios are solvable. We will map out our strategy for solving both, but probably prioritizing scenario 1 first. Scenario 2 support can come later. However, the two scenarios might not be as far apart as they first seem to be.

Of course, show/hide T3.6 and style, T3.8 tags are meaningless without effective calls to the underlying graphics and data platform, the iTwins platform, or any other normalizing platform on top of which the TGN SDK stack is built.

Or in the case of TGN SDKs built on top of non-iTwins modeling platforms, the possibility remains that still a part of the iTwins API, or any other model-normalizing platform (Catenda for example) would be used for normalizing non-model data, or/but otherwise:

1. GUID I/O consistency checks may survive format change anyway (two versions of same model in different formats), and
2. TGN show/hide and style tags are derived from Filters that are built on search criteria anyway, and
3. the consistency of search results based on TGN filter criteria ***may*** give equivalent results in both apps no matter that the search runs against two versions of the same model in different formats in different apps
    1. ***This anyway will be the subject of investigation***
        1. If the results are inadequate, then the search is on for the appropriate GUID match or GUID mapping checking methods

The goal is that TGN show/hide T3.6 and style, T3.8 tags always hit their target predictably and work.

All of that brings some further thought to what we've discussed about the SDK stack in general.

T3.7 TGN Motion controls

Optionally set modeled elements in motion. The TGN authoring user may optionally:

1. set visible elements in motion, ***to clarify complex assembly for example, exploded views, etc.*** Also,
2. the TGN Volume (T3.1) may itself be set in motion
    1. the (T3.2) TGN Viewing Arc will move with it.
    2. *Interesting possibility: a TGN Volume set in motion along a path defined through a model, with the TGN's built-in camera set in motion simultaneously* ***along two paths****:*
        1. *The path defined for the TGN Volume through the model*
        2. *The path of the TGN's built-in Viewing Arc (T3.2)*
        3. *like the moon path around the earth simultaneously moving also along the earth's path around the sun*

T3.8 TGN Style controls

See the entire discussion in T3.6 TGN show/hide controls. All of the same issues apply to TGN style controls. The difference is only in the action we perform after the filter.

- In T3.6 we hide entities that don't pass the filter, and show entities that do.
- In T3.8, we apply graphical styles to the latter, to entities that pass the filter.

Within the TGN are model (twin) entities of certain class, type, and properties. Against Filter criteria we apply style tags. Examples in the following table:

Criteria	Style
Class X	Texture per material name property (e.g., concrete)
Class Y	Arbitrary color (e.g., yellow)
All other classes	transparent
Fire Rating value (2HR)	Arbitrary color (red)

T3.9 TGN additional graphics

Additional graphics include lines, arcs, curves, shapes, dimensions, texts, etc. The following numbered notes pertain to the **Viewing** of additional graphics within a TGN. See T4.9 for **authoring** of these extra graphics, and T3.11 for progressive display of the same, along the TGN viewing arc.

1. Early iterations of T3.9 extra graphics viewing (and authoring, T4.9) may confine extra graphics creation and presentation to the plane of the ***primary face*** of the TGN Volume (T3.1), as seen in Figures 3 and 7.
2. Later versions of T3.9 extra graphics development may enable creation and presentation of extra graphics elsewhere in the TGN view, not confined to any single plane, and yet >>
3. Later versions may present off-plane extra graphics on-plane (see c. below), on the primary face of the TGN volume, as the TGN view approaches position P2 of the TGN viewing arc, which correctly preps these extra graphics for ***T2 TGN Epic*** I/O to and from other graphics editors and CAD formats. The concept here is

that extra graphics authored at any orientation in cartesian space/time, within a TGN, find themselves magnetically attracted to the primary face of the TGN volume, and reoriented logically in the process:

1. Text authored off-plane to face the TGN camera at some position along the view arc other than position P2, may face the camera continuously along the TGN view arc, including at position P2 in which the camera faces the primary volume plane in the normal direction in parallel projection.
2. Leader pointers, arrow heads and such, pointing to model elements, may be correctly anchored to an appropriate 3D point, and when viewing the TGN from position P2, these, and text, and other extra graphics, may all correctly align in the normal direction.
3. This shift of extra graphics from off-plane (if so authored) to on-plane, in fact may be one of the nicely illustrative features of TGN, with further clarity brought from rolling the TGN viewing wheel back and forth just before and after the "magnetic" shifting from off-plane to on-plane, of these graphics, and back, from on-plane to off-plane.
4. Do not exclude and do not forget I/O targets other than traditional CAD formats. Include more modern drawing apps and formats
    a. Here is an excellent example of a modern drawing app: https://www.linkedin.com/posts/mental-canvas_mini-challenge-5-create-your-own-coloring-activity-6849783379353837568-Md8o
    b. Here is another great example: https://www.morpholioapps.com/trace/

T3.10 TGN Generate cut geometry at model/twin elements cut by TGN volume

Generate cut geometry for planar clipping boundaries via support within iTwins API, else for example, from ODA: https://www.opendesign.com/blog/2019/august/part-1-2-generating-cut-geometry-planar-clipping-boundaries

1. A technological simplification is made possible here. Previous compute-intensive graphics operations mandated by the concept that was called "Dynamic Views" in MicroStation are obviated by TGN because the TGN graphics are almost entirely handled by the 3D graphics system with vector stylization made unnecessary except at the P2 position of the TGN viewing arc, and there, vectorization need not be real time and can be processed in batch with little inconvenience to the user.
2. Along the TGN view arc, other than at position P2, cut geometry fill graphics may be raster (see 5, below)
3. Cut geometry of model/iTwin elements cut by TGN volume will be converted at T2 I/O to CAD formats.
4. Graphics beyond the cutting plane are rendered per 3D style controls (T3.8), but at T2 I/O will be vectorized and flattened...
5. Cut regions of cut elements may be stylized by raster textures, or by CAD hatches, perhaps making raster file (of concrete, wood, brick, etc.) the favored stylizing method for TGN, yet reverting to vector hatching at T2 I/O to CAD formats. The latter need not be performed in "real time", but rather "when the system gets to it" then stored in the TGN.
    A. Both of these require:

- Creation of Tangerine material styles library
- Mapping interface and logic from model/iTwin element material name to Tangerine Material style per element material name property
- But Cut graphics can be solid fill color only; material graphics support can be added later

B. When models/iTwins are reloaded with new or different geometry, TGNs update cut fill vector graphics. Real time updating is no priority and does not inconvenience the user. Vector graphics that are out of date can be switched off, or simply ignored by the user until updating completes. They're not visible from the majority of the TGN viewing arc anyway.

T3.11 TGN Progressive Display along viewing arc

iTwin (model) **show/hide** (T3.6) and **style** (T3.8) controls may be applied progressively along the TGN viewing arc. Likewise, T3.9 **extra graphics** may appear progressively, as authored by the TGN author, as TGN viewing rolls along the TGN viewing arc. Model/iTwin entities may be shown or hidden and stylized differently at, for example, positions A, B, C, and D along the viewing arc, as shown in the figure below:

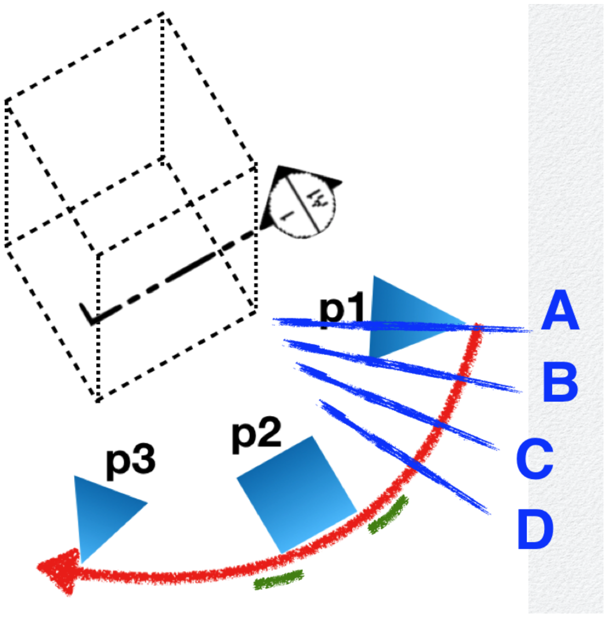


Figure 8: Progressive display of graphics along TGN Viewing Arc

T3.12 TGN Audio

Why not? Include ability for TGN author's audio narration during TGN viewing. Author can add spoken clarification, calling attention verbally to certain aspects of the otherwise visual TGN. And, or, sound effects. Why not. Particularly in conjunction with model elements in motion during TGN viewing.

T3.13 TGN Social Media

Integrate project team communication platform(s), whatever they are, with discussions focusable per TGN.

T3.14 TGN conventional drawing I/O (T2)

See T2

T3.15 TGN Live or Recorded Playback

Viewing a TGN is interactive and of course is contextualized within the entire model (twin) environment. Given that a TGN is a narrowing focused close study of a volume of space and its contents, transformed in various ways graphically, viewer-controlled close inspection of TGN contents via controlled movement along a TGN viewing arc/path, such activity is dependent on graphics hardware and so on, including a number of system software dependencies.

And that's well understood. It's kind of the nature of digital visualization. However, we can make a distinction between live play and recorded playback. Live play is what's described in this paper, and that is reliant on the hardware and software prerequisites just mentioned. There are of course scenarios where the requisite hardware is unavailable, or over the long term, changes at the graphics system level produce unexpected results. At the industry level, this is perhaps stabilizing and appears to be becoming more predictable with industry maturity. But one never knows.

And in any case, there is no reason we cannot support TGN recording as video and pre-recorded playback. Tangerine may develop as an automated system function that on a periodic schedule, each TGN in a project may be recorded to video, with the TGN camera moving at a default speed along the TGN viewing arc of each TGN.

In conjunction with the **T3.3 / T4.3 UI "wheel" for scrolling the TGN viewing arc**, the user viewing the TGN may select Live Play or Pre-Recorded Playback. The options may be indistinguishable, if the current state of the project matches the state when the TGN was last recorded. When they become out of synchronization, this can be flagged in the UI and the TGN recorded can catch up and re-record those out of synch, periodically or on demand.

At project delivery and archiving of digital twins for future use, the Live Play and Pre-Recorded Playback options make a good pair and add to user confidence that TGN graphics will maintain visual fidelity precisely to the graphics that were reviewed and issued at each TGN, long into the future. It's a kind of assured graphics stability.

T3.16 TGN transition options from one TGN to another TGN

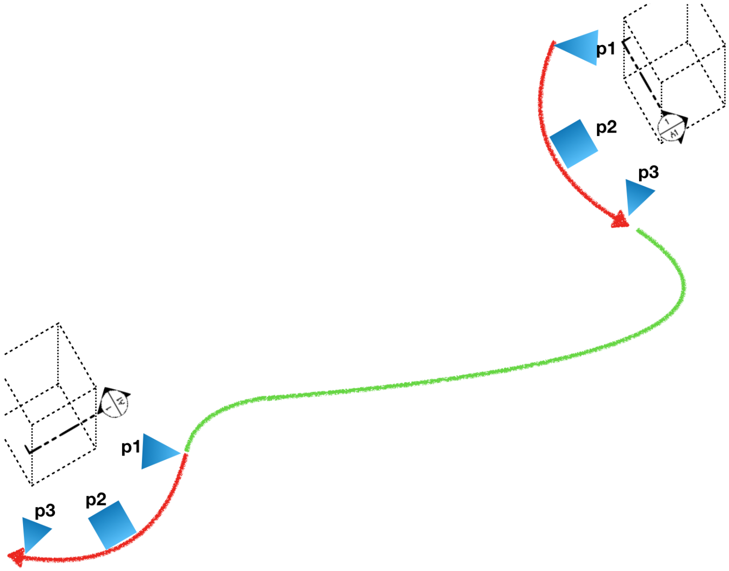


Figure 9: Moving from one TGN to another

There are two ways basically to transition from viewing one TGN to viewing another TGN. The Tangerine app (and TGN function implemented in other apps) will let the TGN Viewer choose an application preference (exposed in the UI) for transition between TGNs, preferring either instantaneous cut, or smooth transition when leaving on TGN to view another:

1. Instantaneous ***cut*** from TGN X to TGN N, with or without a fade to black (or other fade option). This option lacks spatial/temporal continuity. One doesn't actually "see" the continuous movement through the model from one TGN to another. However, let's be honest. This is not always needed, particularly after one is already somewhat familiar with a project. If Tangerine would FORCE the user to experience continuous transport along the green line in the Figure 9, this could become annoying indeed. The user may say *"thanks I already know this, stop wasting my time."*
2. Smooth transition through the model (twin) from the viewing arc of the currently viewed to TGN to position P1 of the next TGN the user wishes to take a look at (which one is that? See T3.17). There are different kinds of spatial logic that can be used to have this green path automated in such a way that it adds clarity to the experience rather than subtracting clarity, including:
    1. Preferred way points can be defined at the project level (or automated based on the range of the model)
    2. proximity conditionals so that the correct waypoints are used automatically
    3. Collision avoidance, preference for use of doors and corridors, etc (or preference for ignoring these).

T3.17 Sharing Links to TGNs

Project participants share links to various TGNs. They may share one TGN, or a few, or all the TGNs in a project. They share via links. The same links are used for TGN navigation, invoking a TGN and then moving to another and another. Sharing methods include:

1. Deliver a list of many TGNs (a list of links, an index, one link to each unique TGN)
    1. How is this delivery formalized?
        1. A list/TOC/index in a digital twin (model)
            1. The TGN code builds this list
        2. An emailable links list in a document
        3. Paste links into messages, chats, whatever...
        4. A table in Excel, etc.
    2. Clicking the link calls the user's default TGN viewer and loads the relevant models/twins and TGNs
        1. Index available as a panel/UI/table in the viewing app (Tangerine app or any modeler app that implements Tangerine SDKs)
        2. If no TGN list UI is implemented in host app, then TGN navigation from one TGN to the next (T3.16) is handled from external list

T3.18 TGN of TGNs (Super TGN)

Why not make a TGN comprised of 2 or more other TGNs? Certainly. Why not? A Super TGN:

Figure T3.18: Hall, R. DIAGRAMS: HANS VAN DER HEIJDEN IN CONVERSATION WITH RICHARD HALL https://drawingmatter.org/diagrams-hans-van-der-heijden-in-conversation-with-richard-hall/ 2021

A **Super TGN** combines 2 or more TGNs, each sub TGN locked in various states. Details for Super TGN will be mapped in more detail later, and will include:

- Define Super TGN Volume (T3.1, per usual)
- Select TGNs to include in Super TGN
  - **For each included TGN**:
    - Operate (T3.3) TGN scrolling wheel to set view orientation along T3.2 TGN View arc as desired.
      - Use alternate TGN arc path (to iso/axonometric view points if desired)
      - Then LOCK

        - Set TGN show/hide (T3.6) and style (T3.8) settings as desired (or accept settings as authored)
            - Then LOCK
        - Flattened graphics only **OPTION** (hide model; show P2 drawing graphics only)
- Experience Super TGN (T3.3) close study via scrolling wheel for viewing Super TGN along T3.2 TGN View arc as desired.
    - Use alternate TGN arc path (to iso/axonometric view points if desired)

TGN 4 [EPIC - TGN rig Authoring]

T4 is for TGN Authoring. T3 is TGN Viewing. T4 and T3 therefore are companions, like, a book is for reading; it's also for authoring. From the perspective of the author, the book is for writing in (and for reading the same as one writes). For the reader, the book is for reading, whatever it is the author wrote in it. T4.X corresponds to T3.X numbering.

T4.1 TGN Volume Definition

As a clarifying focused close study of a model (a model interpretation technique), a TGN begins with definition of the TGN's volume of interest within a model. The TGN author's creation/definition of that volume is T4.1. The resulting viewer's experience is described in T3.1. Possible authoring methods for TGN volume creation are mentioned there, so not repeated here.

T4.2 TGN Viewing Arc Definition - Automatic and Adjustable

The TGN Viewing Arc/Path described in T3.2 is created automatically based on proportions of the TGN volume.

1. **TGN Camera Eye**
    1. The TGN Author may choose from among several default viewing arc path styles, for the camera eye, with the choice stored in the TGN. Default TGN View Arc styles include:
        1. High path
        2. Low path
        3. S path (Diagonal 2-3)

    4. S path (Diagonal 1-4)
  2. Speed along path is controlled by the Viewing using the T3.3 UI scrolling wheel, but for auto-play, speed can be preset to fast or slow options, or author may choose from preset variable speed templates with varying speeds at different segments along the Viewing Path, for example, different speeds at first, second, third, and fourth quarter segments of path
  3. The TGN Author may manually adjust, with 3D grips, the automatically created TGN Viewing Path, altering the default path when needed in special (rare) conditions where the default view paths give an insufficiently clear view of what the TGN author intends to communicate.
  4. Multiple TGN View Arc/Paths may be defined and stored in a TGN. The UI will present alternate path options to the user, for example
    1. one regular T4.2 TGN View Arc path
    2. one for isometric, axonometric view points
    3. Or these may be combined into a single view arc/path

2. **TGN Camera Target**
  1. The camera target also moves on a path. By default, the camera target path will follow a line that is vertical or horizontal (author option).
  2. Speed options for target motion along path may be set by the author. Default options may be test viewed until the author chooses the best option among the optional presets.
  3. As with the the camera eye path, the target path may be manually adjusted by the author, using grips, in rare situations where the default path options give

an insufficiently clear view of what the TGN is supposed to communicate.

T4.3 TGN UI “wheel” for scrolling along TGN viewing arc

No action here is required by the TGN author.

T4.4 TGN Camera transforms to parallel projection at center of TGN viewing arc

No action here is required by the TGN author.

T4.5 TGN Fade and graphical transformations at position P2 of TGN Viewing Arc

The TGN author may accept Tangerine default preset graphical transformation options stored in the TGN and viewed at position P2 in the TGN viewing arc after a fade transition. Or the author may modify these presets as desired. Some of the possible graphical transformation options that we may develop are discussed in T3.5.

T4.6 TGN Show/Hide controls

The TGN author defines and saves the Filter criteria that controls the visibility of model (twin) entities.

T4.7 TGN Motion controls

Optionally, the TGN defines (or chooses from pre-built) model entity motion options. If elements are to in motion during TGN viewing, then those motions are defined here.

An interface may developed for that and may include automating logic for common motion types (like “exploded views” of assemblies)

T4.8 TGN Style controls

The TGN author chooses graphical styles options to apply to model (twin) entities per T3.6 model (twin) Filtering criteria. For example, style mapping per filter criteria as shown in **Table 3.8**.

T4.9 Add any extra graphics anywhere within progressive display of TGN along its viewing arc

Additional to the T3.9 discussion of **viewing** TGN extra graphics, consider possibilities for development of authoring tools for these extra graphics in TGNs:

T4.9 includes MODERN drawing tools

As part of T4 (TGN authoring) we have the opportunity to include new tools for simple drawing graphics that move beyond CAD graphics to include the best of 21st century drawing apps drawing tools. These are to embellish what's seen in each TNG within any set of models tuned by the TGN with specific element filtering, clipping, stylization and so on. Yes, we can, if available, migrate MicroStation drawing tools into iTwins/Tangerine app for authoring. However, we can create our own tools as well, and there are many great examples in the last 10 years of what's possible with such tools (e.g., iPad drawing apps).

Do not exclude and do not forget I/O targets other than traditional CAD formats. Include more modern drawing apps and formats

- Here is an excellent example of a modern drawing app: https://www.linkedin.com/posts/mental-canvas_mini-challenge-5-create-your-own-coloring-activity-6849783379353837568-Md8o
- Here is another great example: https://www.morpholioapps.com/trace/

T4.10 TGN Generate cut geometry at model/twin elements cut by TGN volume

Special graphics are displayed where model (twin) elements cross a plane of the T3.1 TGN volume. This will be an automated function performed by the Tangerine app. However, a TGN author may define and set overrides to Tangerine default presets defining these "cut graphics". The nature of these options is discussed in T3.10. There are several potential technical paths to choose from.

T4.11 TGN Progressive Display along viewing arc

This is an optional advanced capability. TGN graphics may be static throughout the TGN Viewing Arc/Path, or show/hide and style settings may transform progressively during the course of TGN viewing, changing as the viewer moves along the TGN viewing arc/path. If progressive display enhances close study, improves the interpretive quality of a TGN, according to the TGN's author, then show/hide and style settings may be set to be different and various positions along the TGN viewing arc.

So for example, some model entities may be hidden at the beginning of a TGN view arc (at P1), but may then become visible at a later position on the view arc, say between P1 and P2.

Tangerine can provide a UI that enables segmentation of a TGN view arc, and therefore different positions at which TGn show/hide (T3.6, T4.6) and style (T3.8 and T4.8) may be set differently.

T4.12 TGN Audio

Optional author narration or sound effects

T4.13 TGN Social Media

Tangerine may integrate social media capabilities into TGNs. If so, the connections would be enabled at the app levels and user participation involves obviously not just a TGN author but all projet participants with access to a given TGN. Integrate project team communication platform(s), whatever they are, with discussions focusable per TGN.

T4.14 TGN conventional drawing I/O (T2)

See T2

T4.15 TGN Live or Recorded Playback

Optional TGN recording to video. If required, Author sets TGN recording schedule: at a date and time, or recurring (daily, weekly, etc.)

T4.16 TGN transition options from one TGN to another TGN

No Author action is required here. It's an application preference set by the TGN Viewer. Although possibly the automated path waypoints described in T3.16 may be author-adjustable.

However, it may perhaps be a good idea to let the Author establish (or adjust, if we automate these) TGN entry (P0) and exit (P4) waypoints that can assist the automatic inter-TGN camera movement path described in T3.16.

T4.17 Sharing Links to TGNs

TGN links table/list is built automatically. A TGN Author may choose from options for TGN links list/table format and from among various share options, as described in T3.17.

T4.18 TGN of TGNs (Super TGN)

Why not make a TGN comprised of two or more other TGNs? Certainly. Why not? A Super TGN. Both Viewing and Authoring concepts for ***Super TGN*** are described in outline (sketch) at T3.18. The idea will be detailed further later.

TGN 2 [EPIC: TGN (P2) I/O to CAD]

While TGNs have full power in-situ within models, industry requires them also ex-situ in many workflows, unfolded into flat arrays. Tangerine specifies an automated workflow between in-situ (fusion) and ex-situ (flat array), with an easy I/O, into and out of various modeling platforms (like iTwins and Catenda, and portable to and from other modelers), as well as to and from traditional CAD apps. As shown in T3.2 and T3.4, a *conventional drawing* is an integrated subset of a TGN.

T2.1 TGN I/O to DGN/DWG/SVG

At the center of the viewing arc (at position P2, T3.2) of each TGN, drawing's past is effectively integrated into drawing's future, within the TGN. Drawing's past is an integrated subset of drawing's future. We don't leave its past behind or discard it. Instead we amplify its power. At that center position of the TGN viewing arc, each TGN offers an automated in/out interface to traditional CAD formats, DGN, DWG, SVG:

1. Do not exclude and do not forget I/O targets other than traditional CAD formats. Include more modern drawing apps and formats, for example, any of the new generation of iPad drawing apps.
    1. Here is another great example: https://www.morpholioapps.com/trace/
2. Graphics are exportable from any position along the TGN Viewing Arc/Path
3. Export and Import from the P2 position on the TGN viewing arc/path
4. Primarily, support I/O from the P2 position on the TGN viewing arc/path
    1. normal to the TGN volume primary face, in parallel projection
    2. T2.1 Export to DGN/DWG/SVG preserves the true orientation of each drawing
        1. These exported CAD files, if referenced (X-REF'd) into any model, are positioned at their true orientation within the model (*classic hypermodel form*, drawing at true orientation in model)

T2.2 Flat Arrays of TGNs

T3/T4 TGNs are positioned at their true orientation within models (twins). Inherently so; TGNs are built where they are. They're authored in-situ, and viewed in-situ. For the reasons discussed throughout this paper, in-situ orientation is valuable for improving clarity and understanding. But this is no dogma. **In-situ TGN orientation is not always the most clarifying possible orientation**

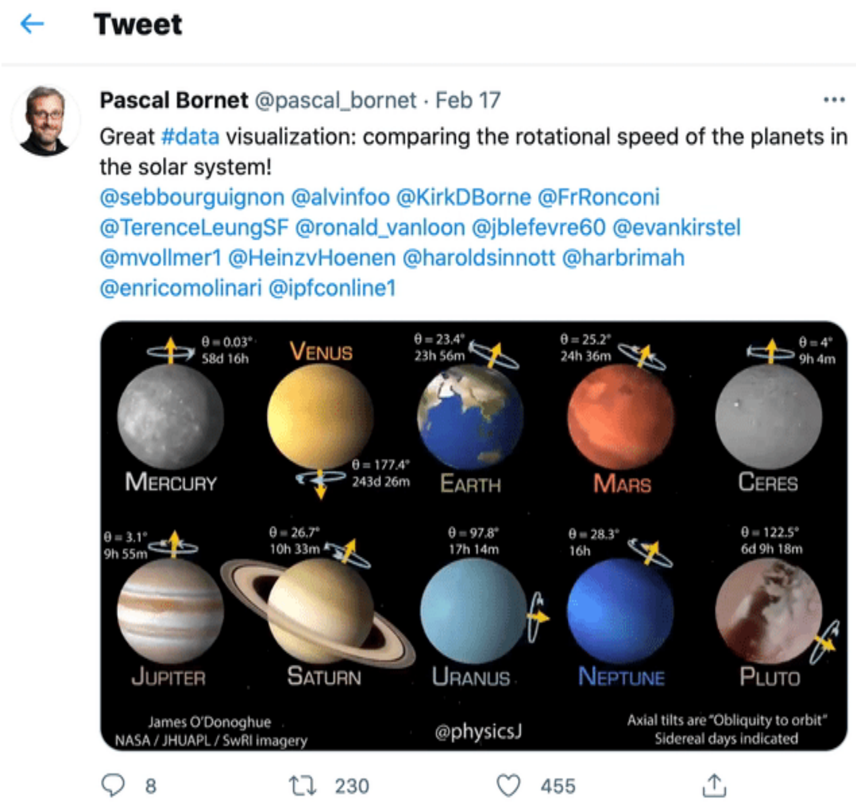


Figure T2.2: https://twitter.com/pascal_bornet/status/1361915813016477699?s=20

Figure T2.2 shows planets ***not*** at true orientation (they'd be **very** far apart), but rather positioned in a flat array. The Tweet, Figure 2.2, makes clear the obvious value of flat arrays (of drawings, or of TGNs). The flat array reveals things otherwise not easily detectable at true orientation. T2.2 flat arrays of TGNs make it easy to compare many TGNs, and clarify relationships between them.

Flat Aray Typical Workflows

There are many typical workflows in architecture, engineering, and construction (AEC) using flat arrays:

Fabrication Collaboration:

Design drawings are shared with fabrication shops (e.g., curtain wall fabricators/installers). At the fabrication shop, design drawings from the engineers and architects are tuned, altered, further elaborated, and then sent back to the engineer and architect for checking. Performing the coordination and quality checks of these shop drawings against design drawings and models/twins requires exactly the kind of back and forth interplay that is the theme of this paper: the T3/T4 fusion of drawing within twins/models and the ***interactive*** ***close study*** enabled by this fusion **is the raison d' etre of TGN**. Both the engineers and architects, and the fabrication shop, need to make close study of what they're doing, using both the T3/T4 in-situ orientation *(which they're doing as a mental exercise unassisted by digital media if software doesn't do the fusion for them)*, and the flat array ex-situ orientation.

When the architect and engineer receive the edited shop drawings, they need to (W1) study and visualize the edited graphics back at their true orientation within the model (twin) to verify whether or not the shop drawings are a good fit with the model/twin, and to detect meaningful discrepancies between model (twin) and shop drawing, and therefore to see, understand, and therefore think about what to do:

- Edit and improve the model? (W1)
- Seek further tuning of shop drawings? (W2)
- A combination of both? (W1) and (W2)

The fabrication shop, likewise, is doing similar work. They're evaluating their own drawings as they develop, and in the process, they're evaluating the architecture and engineering models they received along with the design drawings. Shop drawings are not made in isolation. On the contrary, fitting it all together is the basis of shop drawing development. So on the shop side, users use in-situ (T3/T4) representations and ex-situ (T2.2) flat array, in order to see, understand, and think, about what to do:

- Edit and improve the shop drawings
- Seek further tuning of the design models and drawings
- typical flat array types in the AEC industry: https://tangerinefocusdotcom.files.wordpress.com/2021/05/dominic-seah-drawing-arrays.pdf

T2.2 TGN Flat Array Functions:

1. Custom Transformation: Unfold to Flat Array
    1. Unfold any set of TGNs into a custom flat array (at true scale, 1:1)
        - User options:
            - Select TGNs to send to flat array
            - Select array type
            - Array Settings (spacing, order, number of columns and rows)
2. In the resulting flat array, write the transformation values of each TGN, as hidden attributes, for later re-use:
    1. (X, Y, Z) transformation values of TGN from true orientation to position in flat array
    2. rotation values around X, Y, and Z axes of TGN from true orientation to position in flat array
    3. markers on some "*key points*" of the TGN graphics, and perhaps a TGN bounding box
        1. markers may assist in worst case "wild" editing in CAD app after export (3)
3. Export to CAD formats (DGN, DWG, SVG)
    1. TGNs in (T2.2) flat array form can be exported to DGN or DWG or SVG

2. Users may view and edit graphics in flat array form in their preferred CAD application, with TGN maintaining workflow FLOW:
3. The transformation values and key point markers of (2.3), above, will be written into export formats also as hidden attributes

4. Import from CAD formats (DGN, DWG, SVG)
   1. When edited CAD files are returned to the TGN user, the TGN user imports the DGN, DWG, and SVG files back into the (T2.2) TGN Flat array **IF** T2.2 transformation values were stored in the CAD file when exported, else at T2.1 true orientation.
5. Invert Custom Transformation: Transform TGNs back to True (T3) Orientation
   1. On demand by the TGN user, TGNs that had been unfolded into a flat array, will be refolded back into their true (T3) orientation within the model/twin. Tangerine will reverse the transformation stored at T2.2 (.2), above.

TGN 1 [EPIC - Upgrade Legacy drawings to TGN]

T1 Tangerine functions expand the T3/T4 in-situ closer study techniques to any kind of legacy drawings. Legacy drawings need not remain stuck eternally confined to the original form only. T1 gathers the world's legacy drawings and positions them at their correct orientation in models (and digital twins) automatically. We'll deal generally with two types of legacy drawings:

T1.1 TGN UPGRADE of Legacy Drawings produced by conventional BIM (Building Information Model) Apps

All of the following drawing-handling apps have implemented the hypermodel concept first deployed by Bentley in 2012. They all take flattened drawings and position them at their true orientation, automatically, within the model, or digital twin.

1. Revit (drawings are at true orientation within model at all times but this is hidden by Revit)
2. ArchiCAD (likely the same situation as Revit above, but published to BIMx, below)
3. Graphisoft BIMx hyper-model (in "hypermodel" form: drawings at true orientation in model)
4. Bentley Open Buildings (in "hypermodel" form: drawings at true orientation in model)
5. Revizto (in "hypermodel" form: drawings at true orientation in model. Revizto enables this for Revit projects)
6. Dalux (in "hypermodel" form: drawings at true orientation in model)

7. TEKLA (in “hypermodel” form: drawings at true orientation in model)
8. Shapr3D/Morpholio in tandem (in “hypermodel” form: drawings at true orientation in model)

As TGN acquires reads any of these formats, the already in-situ drawings within the models are ripe for **UPGRADE TO TGN**. TGN will automatically enhance these drawings by adding the T3/T4 TGN functions to them automatically, within any modeling app that implements TGN SDKs.

But TGN can do the same thing for ANY drawings >>

T1.2 TGN UPGRADE of Legacy Drawings in ANY FORM from ANY SOURCE

For ANY set of conventional drawings, automatically orient drawings in-situ within 3D models (twins), and then **UPGRADE TO TGN**. Tangerine will automatically enhance these drawings by adding the T3/T4 TGN functions to them, automatically, within the Tangerine app, and within any modeling app that implements TGN SDKs.

This is achievable. We can automate the detection, within any set of drawings, and automate the *reading* of **drawing titles** and the **drawing callout symbols** that call those titles. This can be used to semi-automate the assemblage of drawing-model fusion, with user controls for tuning the automated drawing orientations within the models, as a first step toward **TGN UPGRADE**.

Conceptually this is straightforward enough that we can evaluate T1.2 project feasibility as: *likely achievable*. There are technical difficulties, but as we have no problem with the idea of aiming in T1.2 for semi-automation, instead of total automation, the difficulties can be minimized and likely will be only small speed bumps.

Generally, when we say we can **TGN UPGRADE** any legacy drawings in any form from any source, what we mean is that we can TGN Upgrade these:

1. **Any drawings of any age (drawn yesterday, drawn 300 years ago) that were drawn by hand** that have been or can be photographed
2. **Any CAD drawings of the last 50 years** in formats that we can access or that can be converted to formats we can access.
3. **Any drawings in any other (non-CAD) digital formats** that we can access or that can be converted to formats we can access.

Conclusion

We understand complexity through close study. It's always the case, isn't it?

TGNs are methods of close study, built into complex data models requiring understanding that drives decision and action. Quite simply, TGNs are the means for *taking a closer look* at models, and digital twins. They are the means engaging more effectively with complex data. People engage complexity through interplay between the wide expanse of an environment of information (a digital twin, model, etc.), and the narrowing act of articulating focus used to interpret and clarify it. In the interplay, thinking happens, understanding grows.

This is true both for human and machine cognition. The latter is another frontier -- the intersection of TGNs, modeled environments, and machine cognition -- Tangerine will address in a second phase of development.

Clarity through Cognitive Interplay

Cognitive Interplay is the driver of media innovation. Tangerine recognizes the intrinsic value of digital twins, and models generally. We also recognize the inherent need for sense-making through high quality interpretive technique within complex modeled environments.

TGN is a step in this direction. It's the direction that matters, not final solutions.

We think TGN is a good step among many steps on the path to clarity. TGN will keep walking that path, evolving along the way.

Phase 2 MACHINE LEARNING / AI COGNITIVE

Tangerine Cognitive, in a second paper. Some notes here: PART 2

Epilogue

The Limits of BIM

Digital 3D modeling as it is used in the design and construction industry (and similar industries) has obvious and great value. However, decades of evidence show that its value is commonly overstated, and that the farther one travels down the path established so far for BIM (or for "digital twins"), the farther one gets from utility, and the closer one is drawn into a never-ending slough through the muck, the purpose of which seems to be only some kind of competition to see who is more macho.

I'll show what I mean in a few graphs. Before the graphs, just one statement that's very typical of BIM industry rhetoric. This one appears in an interview by Randall Newton in his article about blockchain for construction, What does blockchain really mean for construction?:

> *"BIM is very data-centric, and the best way to establish trust is to trust the data itself. ...(the) goal is to make the model the unique source of truth. "Today BIM data is not contractual; there are many sources of truth: BIM, PDM, DWG, paper," says Gueguen. "We believe using Revit as the single source of truth will make the model far better."*

“Bimchain’s first step is a Proof of Contribution module, where contributions, agreements, and validations are certified on the blockchain and made part of the Revit BIM model. “It is a ‘proof of handshake’ to prove the BIM Manager, the architect, [and others] agreed on the model,” says Gueguen. “We believe this will be able to replace the scattered papers-and-signatures process with an indisputable system.”

Blockchain will make its impact on the AEC industry, no doubt, and Randall’s article explains why. But its impact will be greater, and sooner, the sooner this idea of models replacing drawings is jettisoned.

This comes to the point I want to develop. Industry experience reveals very obvious inadequacies of the medium of digital modeling itself, and yet the response always is to ignore the inadequacies, fail to address them appropriately, blame someone/thing else, double down on bullheadedness, and latch on to the next shiny object that *this time* is going to justify and validate the inadequate.

Very macho indeed.

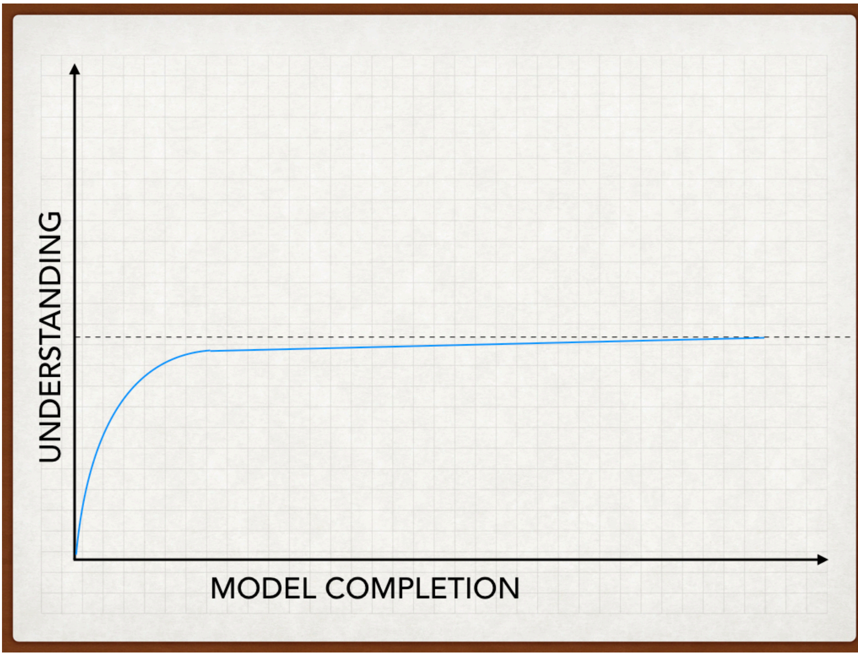
All things in this world have both scope and limits. Digital modeling is no exception. It’s scope (*the extent of what it is and can be*) is vast and highly valuable. But it has also, like all things, limits. The limits of modeling are both real, and very substantial. They need to be addressed, unless an endless crawl through the mud is the desired goal and reward.

I discuss in detail both the scope, and the limits, of both the medium of drawing and the medium of modeling in Chapter 2 of Tangerine Media Innovation Spec 2018. The book is free – read it here on Apple Books or here as PDF. May I humbly suggest that the book is worthwhile reading? It should be of interest to practitioners and software developers alike. Likewise for Chapter 3, which is a specification for software developers. It specifies (*I will be proven right on this eventually*) the future of digital media. Any software company interested can read this and pioneer the future of media itself starting from the ideas and instructions I develop in the book.

The limits of digital (or physical or mental) modeling are two:

1. as models are wide and expansive whole things, they surpass our human ability to wrap our minds around them. To the extent that we require to understand models more than just superficially, models overload our ability to grasp them. We require certain *devices* (other media) to assist us in understanding them.
2. models, by themselves, provide no means with which we can assert and affirm that, at any particular location within them, that what should be shown there is shown there and that nothing that matters there is missing.

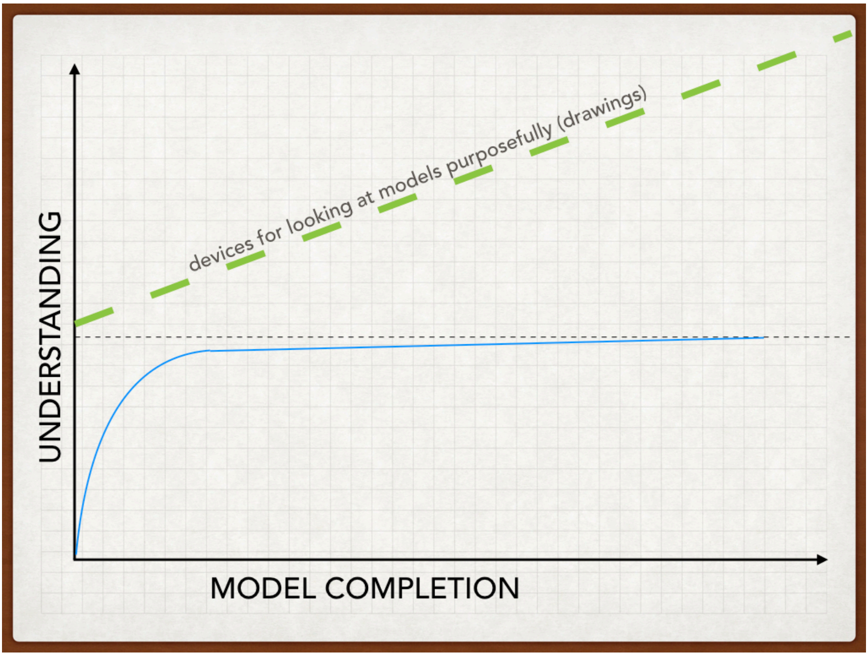
Moving on to the graphs now that show where "BIM" becomes intellectual machismo. Let's draw a curve plotting ***extent of project understanding*** on the Y axis against ***extent of model completion*** on the X:



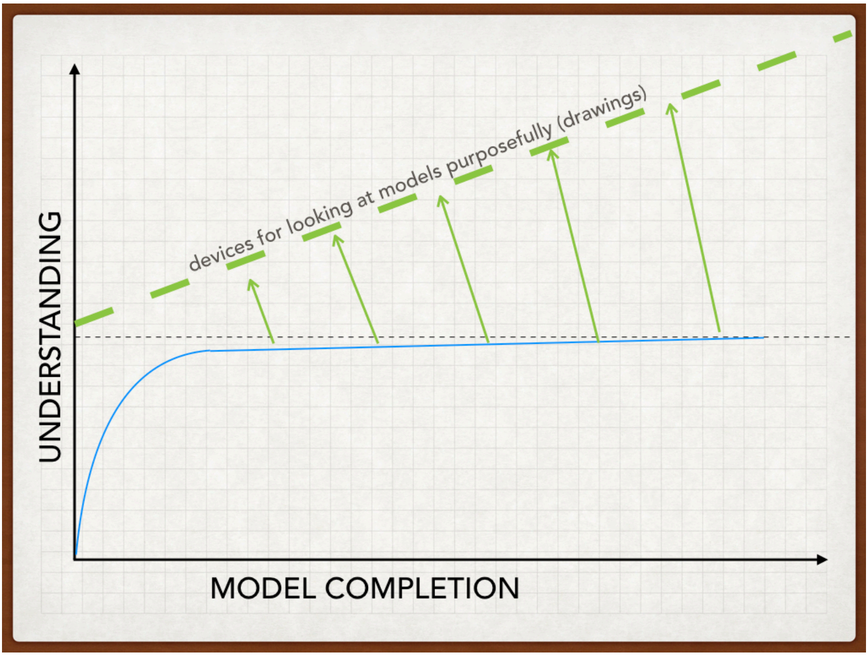
As modeling begins, understanding rises rapidly. As model completion increases toward the right, gains in understanding slow and then approach an asymptote. Greater and greater effort put into more and more modeling delivers diminishing return in terms of increased understanding the project. The diminishing return, and finally no return, is a function of confrontation with the fundamental limitations of the medium of modeling itself (1 and 2 above).

BIM industry machismo can be observed in the determination to continue modeling with negligible gain in understanding. I should emphasize here the extent of this kind of macho determination that's now typical. Though anyone who's been in this business doing this themselves, well and truly knows this already. What remains, only, is the extent to which they're ready to admit it.

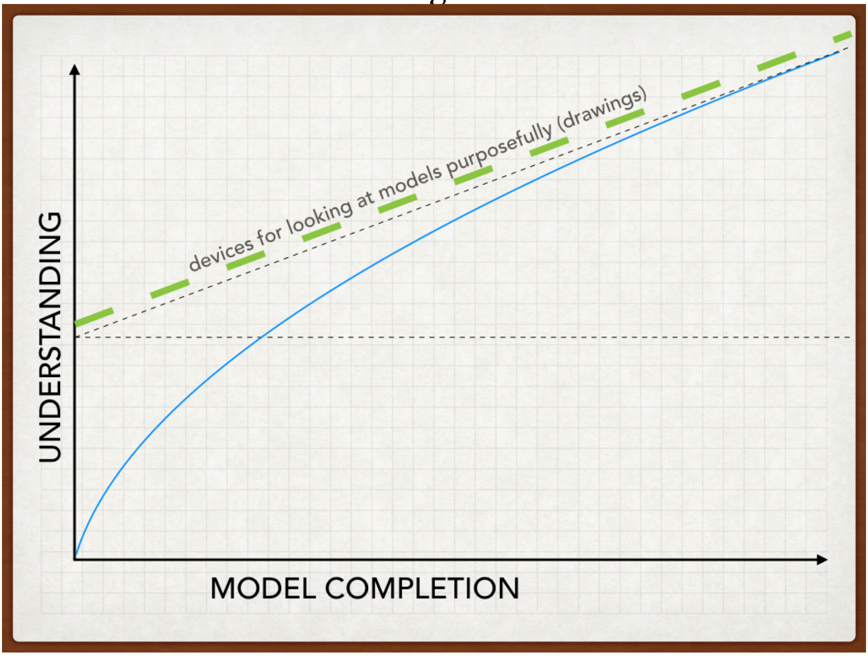
Let's add the medium of drawing into the graph. Let's call drawings "*devices for looking at models purposefully*":



Drawing acts in a very energetic way to pull the understanding curve in the positive Y direction toward greater understanding.



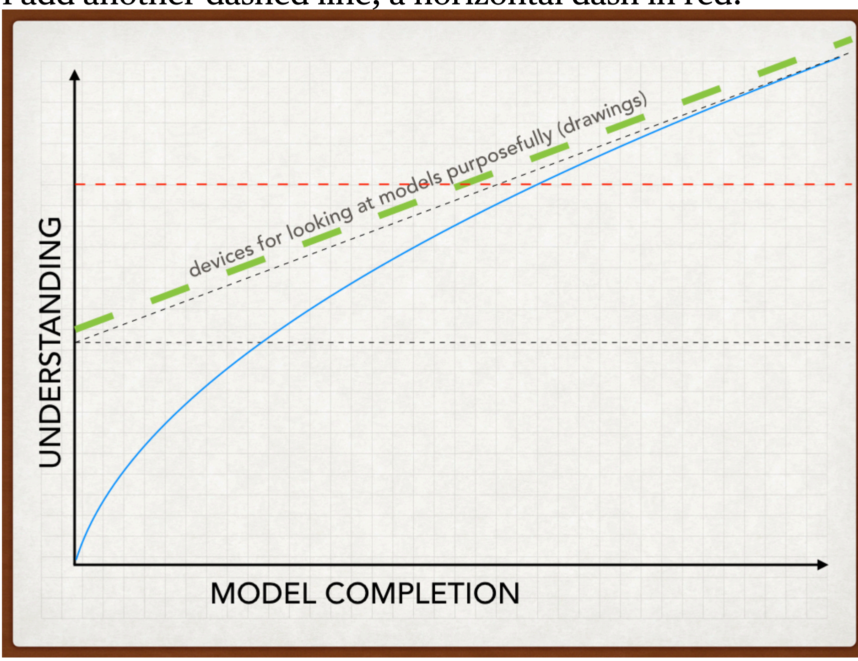
We're gaining two strong positives. Increased understanding (which is kind of the point). And increased value derived from our modeling efforts:



We can adjust the slope of the new asymptote line. The angle of course will vary depending on many factors. However, it's the general effect that's of interest, as well as the mechanism for it. I'll describe the mechanism of the *uplift in understanding*.

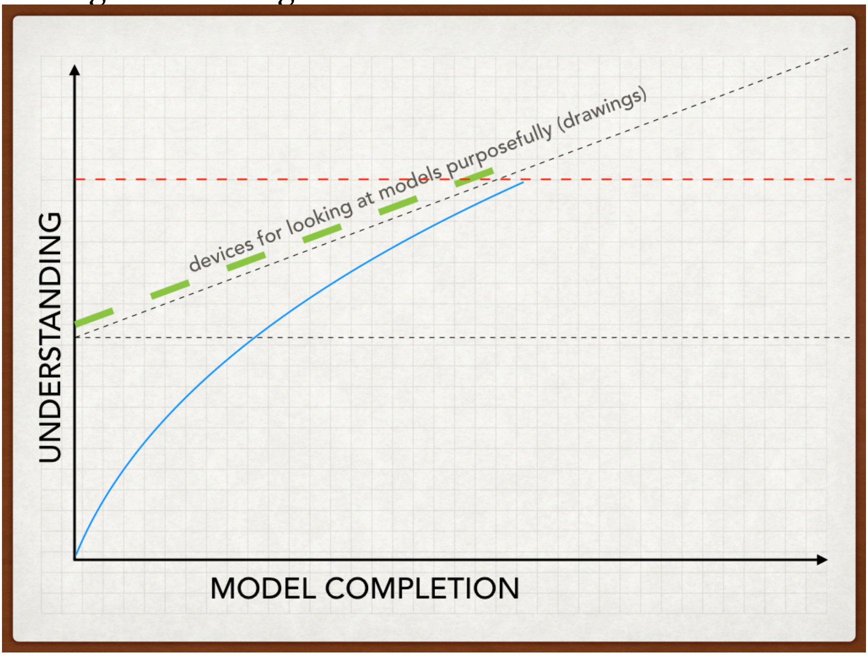
But first, note that the more we model and draw, the more we understand! There's a win-win situation. Our understanding still has a limit, but, the more effort we put into study through drawing and modeling, the more our understanding grows. That's the kind of curve we need, as opposed to the earlier curve where after a certain point quite early, our efforts fail to return gains in understanding.

I add another dashed line, a horizontal dash in red:

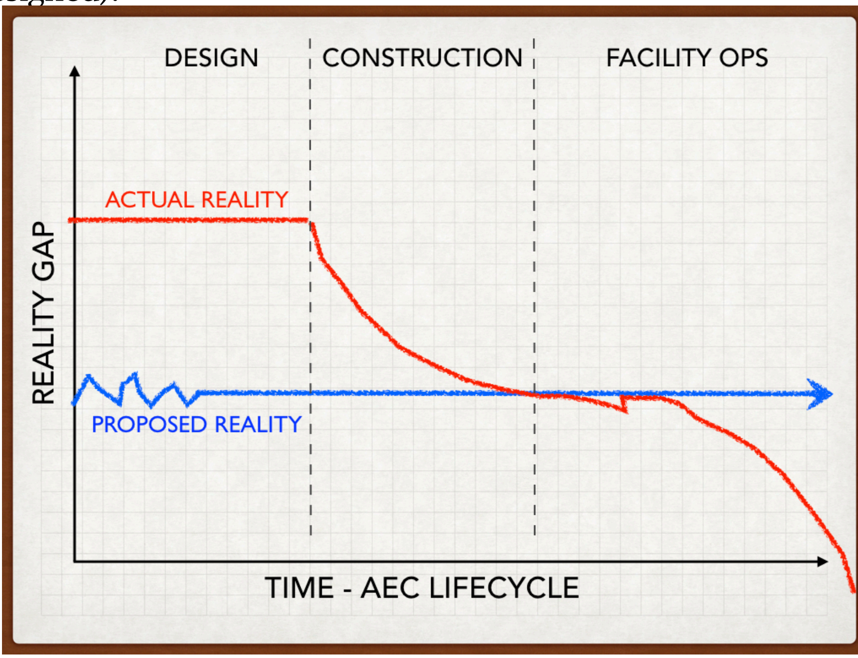


This line is important as it indicates, for a phase of a project – design, construction, *or a blend of concurrent design and construction as is the case sometimes* – it indicates where the level of project understanding is sufficient, beyond which further understanding is unnecessary. If anyone hyperventilates at such an idea, suffice it to say that, for example, when construction is completed and handed over to an owner, the construction team reaches a practical limit at which it will no longer continue to invest in further understanding of the project, other than for ancillary purposes. Naturally, the owner will take over investment in understanding the built asset, but this would be represented by a new graph, with, as for the design and construction team, understanding rising through study of models and drawings *in tandem*, together with direct experience with the real facility.

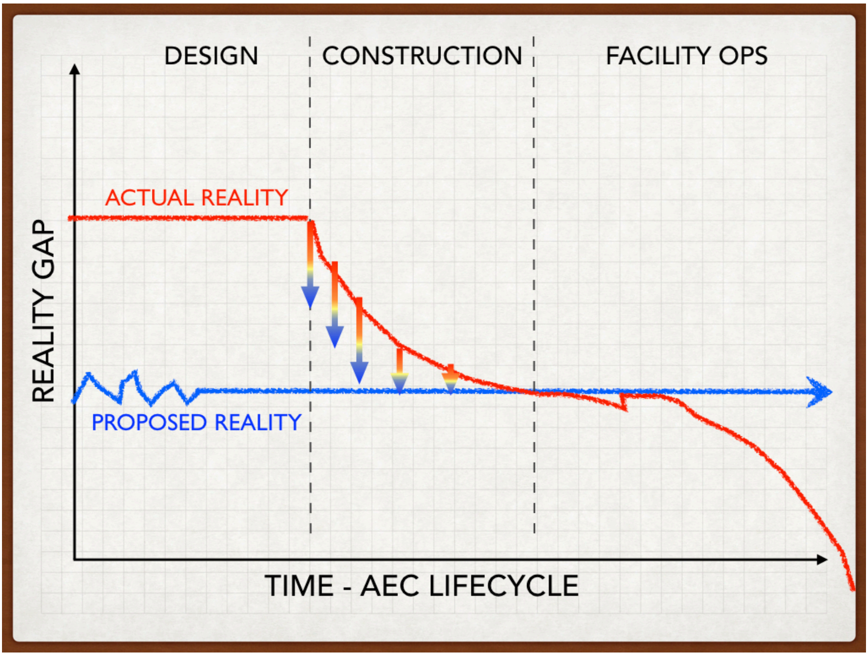
In any case, in such a scenario, the construction team may (will) cease its knowledge investment at the red line, so modeling and drawing would cease there:



Such is the basic nature of the design, construct, operate lifecycle. And for that matter, here's another very simple graph. It shows the relationship between actual reality (things as they are) and proposed reality (things as they are designed):

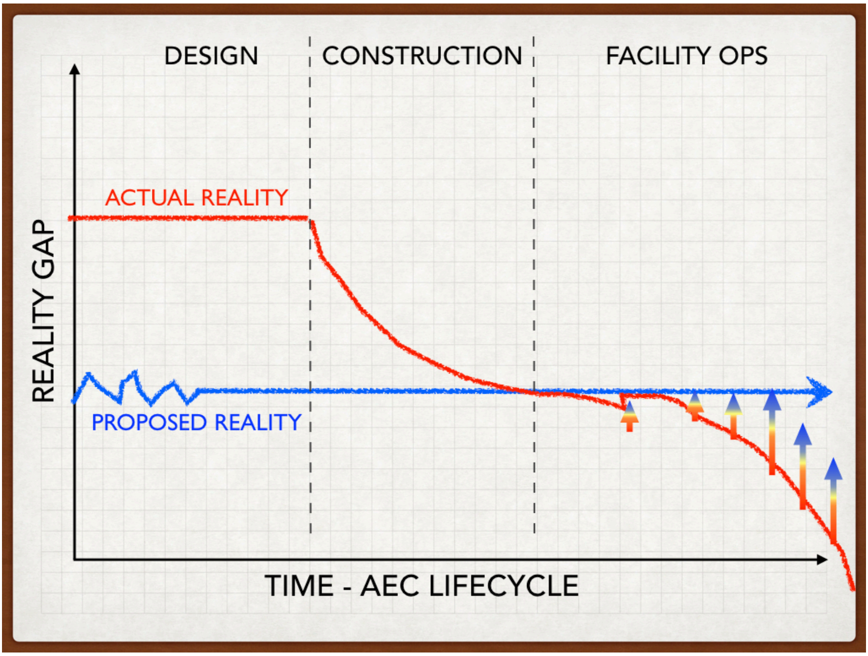


Things as they are, at the start of a project lifecycle, are different than things as they come to be proposed. There is a wide gap between what is, and what's designed. Hence the blue and red curves are far apart during design. The purpose of construction is to move the red curve (things as they are) toward the blue curve (things as they're designed):

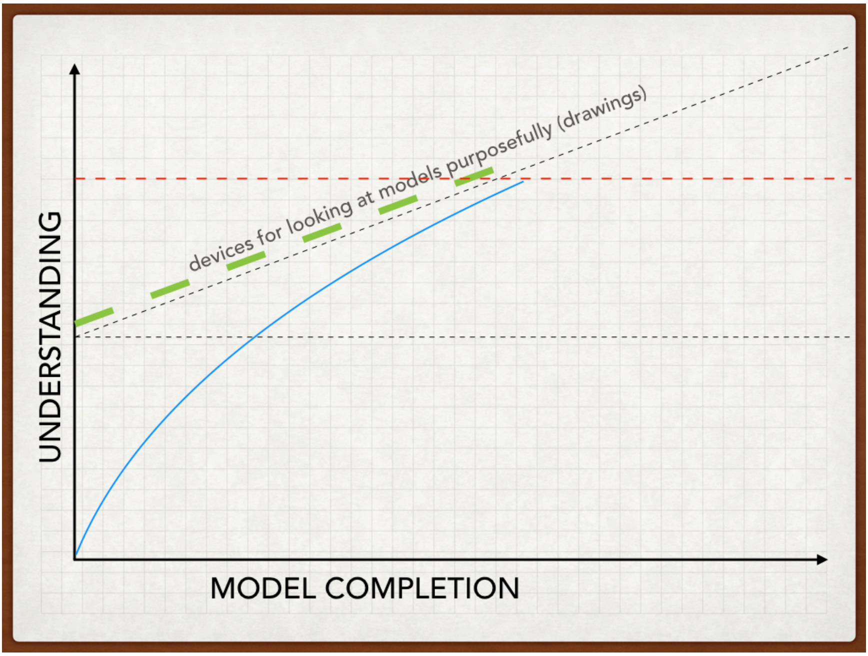


At the completion of construction, things as they are, actual reality, becomes *things as they're built*, "as-built". So at the end of construction, the as-built reality, red curve, comes as close as it will come to what was designed (the blue curve).

Thereafter, during facility operations, actual reality moves away again from proposed reality as entropy takes over. Nature has its say and nature says its going to degrade your facility and thus maintenance is required to push back against decay and keep systems in optimal condition in conformance with designed reality. Naturally, if maintenance is not performed, over time the facility will degrade to ruin, which again creates maximum distance between actual and proposed reality:



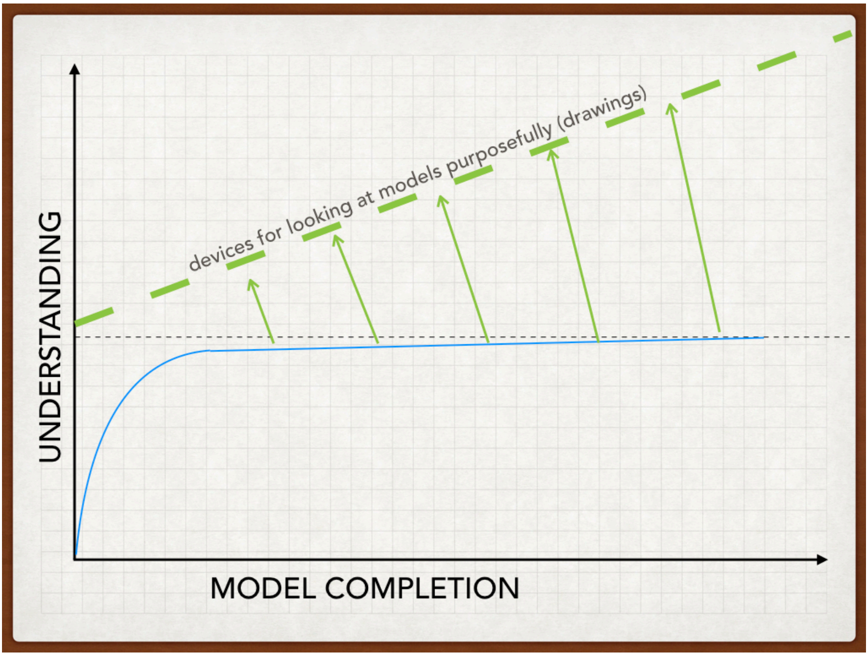
This segues back then to the ***model completion / understanding*** curve. During each of the major lifecycle phases in AEC, there is an appropriate cutoff line after which further investment in understanding is no longer made. Though handoff is made to the next phase, at least part of the team that made its investment in understanding can appropriately retire its efforts, as again shown here, with efforts cut at the red dash:



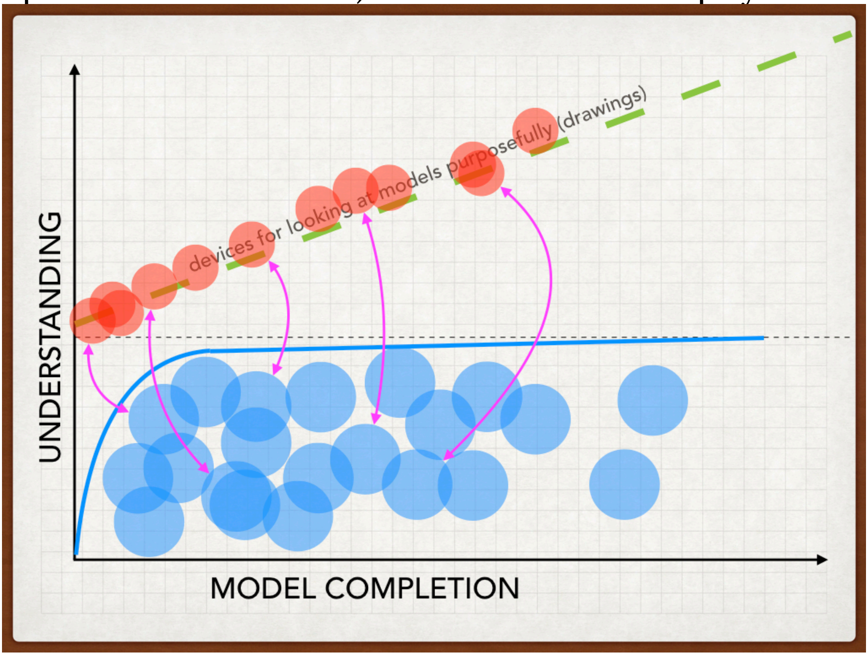
The simple point being that these two sets of graphs are related in the following way. The purpose of drawing and modeling is to gain understanding. And the purpose of understanding is to contribute effectively in the AEC lifecycle – during design to make an appropriate and effective design, during construction to move actual reality toward proposed reality, and during facility operations to keep the facility in optimal working order. It is toward these aims that understanding is directed, and beyond these aims, investment in understanding is ordinarily not funded.

The Mechanism of Understanding

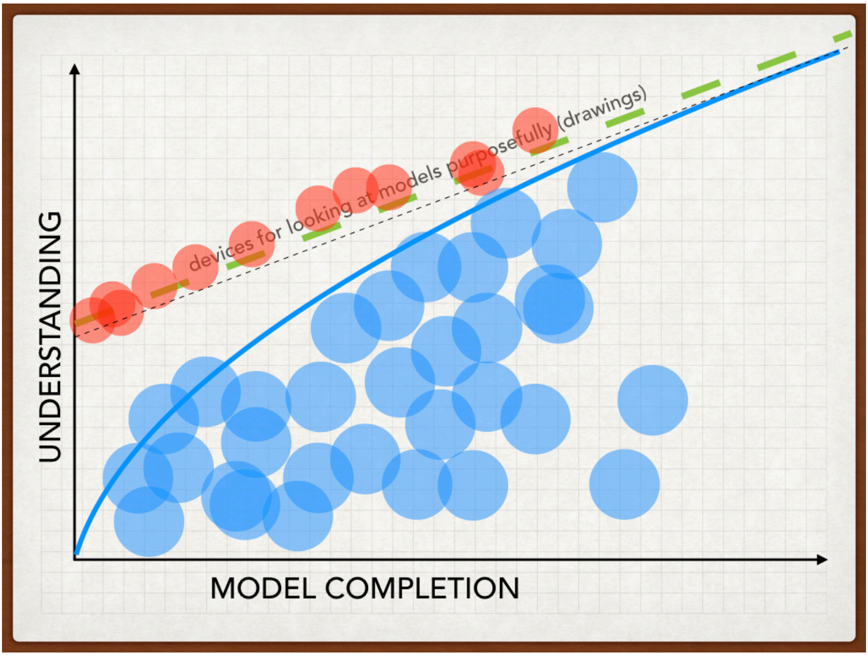
So what is the mechanism of understanding? By what means does drawing pull the modeling/understanding curve toward increased understanding? How does it do this?



The answer goes to absolute fundamentals. The activity of *thinking* is represented by the pink arrows (next image, below). Thought, of course, produces understanding, and thought, it seems, functions through some kind of back and forth ping-ponging between the wider perceptible environment (like a model) and articulating acts of narrowing focus (like drawings). So in the graph below, drawings are developed over time (the green dash, now supplemented for emphasis by the red circles), while models are developed below the blue curve. The mind at work (the pink arrows) bounces back and forth between these and from this action, evidently, thinking happens, and understanding grows. Certainly I am no neuroscientist, and nor would anyone in the science of the study of mind presume to even begin to say how cognition works (it remains an almost total mystery). However, we can observe some of the most basic dynamics of its function: there is wide (environment), and there is narrow (focus, taking a purposeful and closer look). And these are in interplay.



In this interplay, some kind of energy is generated that results in the inexorable growth of understanding, and so the blue curve is moved in the positive Y direction, toward greater understanding:



So now in the field of AEC, or the field of software for architecture, engineering, and construction, when one talks of jettisoning the medium of drawing in favor of the medium of modeling, then one is denying the basic observable dynamic of the growth of understanding, and hacking at the root of cognition itself. No small error indeed.

Let me give an example, a very nice one in this 35 second film clip recently posted on Linkedin by the film maker Jim Cummings. Here's the video link.

Take a cue from this, the essence of narrowing focus and taking a closer look:

film clip posted on Linkedin by Jim Cummings: video link

I mean, we all relate to that viscerally. It goes right to the core of what we do when we understand anything anywhere. We're ping-ponging our perceptions between the wider expanse of the environmental whole of a place, and the various articulating acts that we undertake of narrowing focus and taking a closer look at something that matters. In this back and forth, meaning is formed, understanding grows. We come to know a thing, and a place.

This of course relates directly to AEC media in the sense that the wider environmental whole of a place is the model, while the articulating act of narrowing focus, to take a closer look at something that matters, is an act well and truly embodied in the medium known as "drawing".

Note that we don't understand one without the other. We don't understand the *closer look* at the woman giving directions in the cab, without the wider environment of the city within which this narrowed focus finds its subject. And the reverse is true too; we fail to grasp and make sense of the wider environment (of the city in this case) without a continuous series of *closer look events* that we engage ourselves in. Our minds are drawn to clarifying moments, and orienting views high and low, large and small. The sum of these, within the context of the dynamic interplay between environment and focus - well, that seems to describe, hopefully to some useful extent, the engine of cognition.

So what then?

None of this matters to those committed to a continuous contest of *¿Quién es más macho?*

But for anyone who wants better media that's better fuel for cognition, new innovations in media that will define the future of media itself, and for any software companies that want to pioneer that and bring it into existence, well I can only say, the future is there, waiting for you to define it.

So is it a machismo contest? Or do we move forward in ways that matter?

About the Author

\rob snyder
Helsingborg, Sweden
Linkedin Twitter
rob.snyder@tangerinefocus.com

[Tangerine] Makes Insight Tangible
Spatial Media Innovation and Cognitive Computing

Tangerine: https://tangerinefocus.com
LinkedIn: https://www.linkedin.com/in/robsnyder3333/
Experience: https://tangerinefocusdotcom.files.wordpress.com/2021/01/rob-snyder-jan2021-cv-.pdf