

MAGIC

William Bricken in correspondence with Dan Shapiro
August 1994

DAN

I've been thinking about magic some more, and am working towards a breakdown of the activities involved with an eye towards building blocks, implementation and quick demos. The problem, of course, is that my conception of what is easy/doable is based on a pretty limited understanding of the building blocks you have in place. So, I thought I would pass some of my ruminations past you, the experts, for a reality check (even in their early, rather unclothed stage).

Here is the deal:

View the purpose of magic as making organized modifications of a single person virtual world from within. (Later, I'll expand this notion to talk about mediating between disparate opinions about reality contents in a multiple participant world). The metaphor will be to build an icon representing those facet(s) of the world you want to change, and to then apply modification operations to the icon, simultaneously transforming the world. The mapping from icon to world can be one-many, which makes the act of modifying the icon more powerful than modifying the world items directly.

Along the line of what I described in the short write-up sometime back, let magic have the following phases:

Opening	-> initiating the process of casting magic
Naming	-> identifying the object(s) to be manipulated
Spell Casting	-> performing the manipulation
Closing	-> terminating the process of casting magic

NAMING

I envision the guts of naming as a pattern match (from icon to world) corresponding to the simple code form:

(match icon world-subset)

This immediately raises the questions "where does the icon come from?", and "what restricts the search space (from the assumedly intractable space of all objects in the world)?".

I can think of two plausible methods for building the icon. The first is to assemble it from available pieces, which amounts to a mini construction or

modification process in and of itself. I'll come back to this later.

The second approach is only sensible from outside the space but it serves to bound the problem: let people browse the data structures describing VR objects and directly pick and choose properties, sets of properties, or whole objects that must be in the match set. Example: point at the data object (if there is just one) representing a table. Example 2: point at the tuple (color:blue), with later intent to modify all (color:blue) tuples within the system. An obvious addition is to type tuples directly. This obviously sucks as a user interaction metaphor since it requires the participant to understand VR at a coding level, and makes no advantage of the inclusion or visualization properties of VR itself. However, we *can* view this as a direct statement of the goal of the first half of naming: building a pattern composed of an arbitrary set of tuples to be matched on the database of tuples in the world. You can see where this going: we use the VEOS pattern matcher to implement the guts of naming: performing the match.

From the VEOS documentation, the matcher has a number of good features for supporting naming. As a quick list, it has a lot of expressive power, the ability to return all vs. only one match, and prior attention to speed. I have some questions though:

1) does the matcher have provision for *near* match? The character of magical naming would seem to require it in the long run. How would we extend it to do so?

2) are grouples used as the data structure for defining all objects in VEOS built worlds, or will there be a significant subset of object definitions not accessible through the pattern match mechanism? Would we have to change the way the VR system currently works in order to build any reasonable demonstration of naming in the pattern matching metaphor?

I'll point out an interesting extension here. Since magical manipulation can only be applied to those grouples retrieved from the naming process, *everything* we want to manipulate must be expressed as (or significantly defined via) a grouple. E.g., if an object's response to gravity is defined in a grouple, we can change it. This amounts to parameter tuning, but it clearly interacts with underlying simulation processes. The task of changing gravity itself (meaning VR procedures) via magic is obviously more difficult, though doable in a restricted way - it maps onto method selection and flavor combination at an implementation level (graphical programming languages, anyone?), but again, those "flavors" would have to be accessible to the pattern matcher. Are they?

Here are some questions about the second argument of

(match icon world-subset)

I'm assuming, first of all, that the VEOS matcher isn't so blindingly fast that it obviates any foreseeable need to restrict the search space for speed reasons. Actually, even if it *was* that fast, we would still want to bound the second argument simply as a means of effecting information retrieval. That is, the participant might want to modify the object in front of him, instead of all objects like the one in front of him, regardless of planet of origin, etc. This suggests a few important world-subsets; "all the grouples in the world", "all the grouples representing objects within 10 simulated meters in the field of view", or "grouples internal to the representation of table #1". Because of the distributed nature of VEOS worlds, is the set of all grouples in the world even accessible? Are the other subsets accessible?

An interesting work-around is to use the law of contagion as a search space reduction technique. That is, given an icon, only apply the pattern match operation to that subset of the world the icon has been in contact with at some time in the past. This might be useful, though odd in that it comes out as a restriction instead of a strength of match characteristic, which would seem to be more desirable (or at least more faithful to intuition about how magic works). If we accept the premise that prior contact (or prior knowledge) is important (imposed for efficiency reasons, or as a consequence of the parallel metaphor no less!) I can see how naming could become quite a skill. A participant might spend quite a lot of time gathering names, icons and exploring simply to be able to effect later change! (The wizard of earthsea reduced to a time/space tradeoff.) I guess the underlying capability question is quite simply: can we record object to object (proximal) contact during immersion as a method of implementing contagion?

Returning to the problem of where the icons (the input to the pattern matcher) come from, I envision a micro construction or modification process like the one which will show up in casting, where the goal is to perform a change. Here though, the problem should be simpler since we only want to construct an icon that is good enough to behave well under matching. Given some implementation of similarity (*near* match, or subset matching as currently supplied), we should be able to employ relatively abstract icons. For example, we might stack a tripod and a saucer, and expect it to match onto the representation of the space needle. Adding the right coloration might help the match. Note: once the match is performed, it is presumably easier to establish again in the future (since the icon now has contagion with the space needle, and will match it again unless the needle itself has deeply changed). Say, an idea here: we can implement the notion of true names, quite usefully. That is, once you have matched a single, identifiable object, you might get its true name: a pointer to the object. Future modifications bypass the non-specificity of the matcher.

CASTING

Casting has the following modeling problem: how do we express the change we

want the named object(s) to undergo?

I can't go into this now (I'm out of time) but the gist of the thought is that we need a small vocabulary of modification operations that is easy to implement but large enough to employ in a reasonably convincing demonstration (where the goal is to convince ourselves that this whole magic thing is a good idea). I can see two representations: one is a command format (e.g., change an object's shape by hand gestures), the other is more like a process model (e.g., dunk the icon in a vat of platonic green to change its color, or put it in a funnel to make it come out smaller). A third possibility is to specify change by defining endpoints (as in polymorphic tweening), and interact with the underlying engine to implement the change. I have more to say on these topics, but for now I will just ask some questions: what world mod ops do you think will be the easiest to implement?. What, if any, have been implemented so far? I'm thinking about operations to change an object's physical extent (say, through a potential field metaphor interacting with an attractive or repulsive charge at the tip of the simulated index finger), or operations for changing properties (color, size, location, etc).

WILLIAM

Yes, this is a good thing. You are thinking clearly about VEOS and pattern-matching (M&S).

Not sure why an intermediate icon, when we could change (a copy of) the thing itself, or perhaps the thing scaled down. Seems like you could identify the set of things in a one-to-many map, and manipulate them directly. Spatial location might be screwy...

I recognize Open-Name-Cast-Close from D&D, a good structure. How much are we building a new metaphor vs renaming old (desktop) functions?

Right now, for the icon to exist, it must be constructed. Again, what about CLONE existing objects and change some of their attributes.

Search is controlled only by partitioning the db. We can do smart things like dynamic threading to speed up. Idea is that each entity has a local db, so db is modular at the start. We have restrictions so that you can't PUT into other entities.

Matching: No NEAR yet (and no TIDY), but easy to build I think from primitives. Can't recommend name-near (as in rhyming, searching chars for near match), but attribute-near is easy.

Everything is grouples, all are accessible, including creation of new properties by assertion. Of course this is code fascism, and someone has to enforce it. No changes to do naming.

What's wonderful is that we have a closed world db.

RULE 3: if it isn't named, it isn't.

I believe the philosophy is avoidable here, *there is no fantasy in the virtual world*. That is, you can manifest anything, but until you invoke, the (fantasy) thing is in you head only.

M&S has partitioning (most interaction is narrow, between two specific entities, so no global monolith) and filters to put conditions on search. But the way search is constrained is to *draw the desired pattern*. If you can show the template, you can do it.

Distributed data is completely accessible, so long as it is public (not inside an entity) and asynchronous (don't know when the query will get answered). We're somewhat extreme here, no entity is obliged to reply to a request, so no handshaking.

We haven't build a history mechanism yet, so no contagion. Should be straight forward.

Perhaps "contact" can be defined as message passing. If you have accepted a message from another, you have contaged. The point of similarity could be the content of the message.

Naming sounds like a learning strategy.

Easy to do proximity, but might be explosive.

Tripod and saucer is a good test case. Seems like the big problem will be *the location of semantics*. Is the similarity match structural? How will the space-needle polyline description know what it looks likes? Who determines what looks similar?

Modification vocabulary is a problem. but not too hard. There are half a dozen principled ways to build solid objects. Each of these systems provides a set of constraints on change. Another fun project is to name all the simple tools in VR (lever, screw, pulley,...) These could be the tools of change. The question becomes how, not what.

I'm inclined to do generalized sweeps as the generation method, but changing objects (particularly topological, like cut in half) is generally difficult with traditional techniques.

Changing names properties is trivial. Changing spatial props is trivial. Changing shapes is harder. I'm excited about changing *space* itself.

DAN

Thank you for all that. it is going to take me a bit to digest, so I will respond in pieces.

First, my suggestion to use icons. My intent is to address the naming problem, where the goal is to identify the objects in the virtual world that will be effected by magic. One way to do this is to wander the virtual space and *select* objects directly, by pointing at them, touching them or whatnot. It would seem odd *not* to have that capability, so it ought to be supplied. My thought was that icons did this one step better: thinking in terms of information retrieval, use of a specially constructed icon lets you retrieve a set of objects that you want to effect, associates them with one operand (the icon) for use in magic, and lets you highlight a subset of object features (the ones which matched onto the icon) for your magic to modify.

So, I *think* icons are useful. It looks like they provide a generalization function, alongside a method for focusing attention. This points out a distinction with the magical sense of "true names". With a true name, you get the object, and can cast magic on *any* part of it. Using icons as I described, you really only name pieces of objects, which suggests a restriction to one's abilities. This all strikes me as an implementation detail - I'd be inclined to keep the distinction between mapping into an object and having a true name if it had value to the user. Let's say you have an object, a copy of an object, or an icon mapped onto an object(s) in your hand. Now, we can ask about magical manipulation. This ability is obviously core to the whole enterprise, which suggests that the issue (given limited time) is to get at *some* such capability, if not the *right* one. I mean, what the heck, it's an empirical science. You mentioned what sounded like sweeping generalized cylinders or cones (for generating forms). How about applying processes (grow, shrink) to specified physical regions, and playing morphogenesis? Add a few start and stop bit cues, and I'm certain you get something rather general. I mean, *we* are specified that way.

I am a little confused about how we visualize and establish properties (not morphological) and behavioral attributes, but I can tell I have ideas kicking around in my head... something is making noise back there at any rate.

I liked your idea of defining contagion as the act of having passed a message. It would seem to me that any form of interaction would boil down to that (except maybe seeing or hearing something). That also obviates any need for a history mechanism too - contagion is just a locally maintained property by any object that cares to remember it. This would suggest that icons are predefined - if it pulses blue you know it has memory, and can be used in magic.

This is weird. I keep "discovering" that things which have meaning in DnD games are "predicted" by a reasonably well reasoned software implementation

of magic. M chickens and N eggs, or eggs and chickens.

You asked how much we are inventing new metaphors vs renaming old desktop functions.. Since we are after two things here, a world modification capability and an interface metaphor, I'd say we can't take a purely functional or purely aesthetic view. However, since no one but us knows what magic *really* is, we have some license to play with the definition.

Some potshots: the definition of similar depends on how hard you squeeze. Apply more energy and the search goes wider and deeper, and looser (in its tolerances). Good icons match what you want faster (One hopes - sometimes detail loses where abstraction mechanisms gain; sketches over photos for identifying criminals.) Location of the semantics: hmmm. A puzzler. I wonder if we can take the cue on the dimensions of similarity from the user's perceptual transformation: if he is intensifying color, then color is a most importation match feature. If he tilts his head, or tries naming upside down (??) we get a preferred orientation on initiation matches?

OK, changing space. If it is represented anywhere as a grouple, we can tie it in knots. If not, then maybe the user can just see space as a small blue herring - you know, the kind that whistles and hangs on the wall.

WILLIAM

Design doc for magic: yes, we will need this. I suspect it will be quick, since we have a lot of infrastructure in place.

I personally like writing about constructivism, etc, and would love to have that on paper. But I'll bet casting technique will depend on interactive experience (do rather than write).

Both paper products might be generated from research notes as we work.

Demos:

First define casting elements in terms of VEOS pattern-matching. Then we can play. We really don't yet have a rich variety of entities (lions, eagles) so Merge might be difficult in full blown application. What about first steps: Merge a cube with a sphere to get a shape-round-thing. We do have the complete structure of a cube (did you know that cubes have around 100 intrinsic attributes (hint: group structure of rotating cube adds pivot points and lines, orbitals,...))

We are fond of adaptive algorithms (Dawkins, Sims) which just muck attribute code by brute force. Gets into the formal structure of algorithm creation (machine learning).

Yes I want to think about semantics of magic. I think I've got an inverse semantic model. "Reality" for the virtual world is its anchor into the digital database. So semantics is how the db changes.

Enhance: polygon manipulation is too much trouble. We have to muck with the renderer as well as the world. This is different than *creating* polygons from inside.

Strength of matching: I believe all roads go through the pattern matcher. Necessary and unavoidable.

Contagion: yes but a lower priority. Might be hard to do history well.

True name: lower priority