Conversation on Code 2015-03-03

Code: Emergence, Art, and Literature with Andrew Max Maxwell and Casey Reas
LACMA Art + Technology Lab
March 3, 2015

> *The reason I started to code was because I wanted to be able to draw in a different way.* - *Casey Reas*

> *I became less interested in the poem as an object, an objet d'art, towards smaller and smaller units because I was interested in minimal definitions of intelligibility or sense.* - Andrew Max Maxwell

Peggy Weil: This is the Second in the LACMA Art+Technology Conversation Series and tonight's talk is titled *Code: Emergence, Art, and Literature* with Andrew Max Maxwell and Casey Reas.

We wanted to acknowledge the role of code and process within the tradition of 20th and 21st century art. Early 20th-century practitioners experimented with dice and mechanical approaches to random and chance operations; late mid-century artists and composers introduced computation. Currently there is enthusiasm for techniques such as generative software and code to explore emergence in literature and art. Our two guests tonight embrace code in their practice and their broader community work; Casey Reas as a visual artist and co-founder of Processing and Andrew "Max" Maxwell as a poet and founding director of the *Poetic Research Bureau*.

Andrew "Max" Maxwell wears (at least) two hats; he studied linguistics and natural languages and came to a career in software development by accident 15 years ago,

joining a Southern California early semantic web start-up to do ontology development and taxonomy design.

At Google, he a product manager working on machine learning and behavioral modeling initiatives, while also coordinating arts and museum exhibitions and partnerships for the Google LA office - and it is in this capacity that *Max* is a Art + Tech Lab advisor.

As *Andrew* or *A. Maxwell,* he is a poet, aphorist, and founding director for the *Poetic Research Bureau* (in downtown LA) which focuses on expanding the scope and definition of literature and language-based inquiry to include experimental, provisional, digital, and algorithmic forms of literary production and distribution.

Casey Reas writes software to explore conditional systems as art. Reas' software, prints, and installations have been featured in over one hundred solo and group exhibitions internationally; also the author/contributor of several books and particularly apt for this discussion, the co-founder of Processing, an open source programming language and environment for the visual arts. He is also a professor at UCLA Design Media Arts.

Casey Reas: Okay. Thanks, Peggy.

Andrew Max Maxwell: I just happened to be walking out of work today and noticed that in today's New York Times in the Science Times section, there was actually an article on termite ecology and termites are often one of the metaphors that we use to explain emergence – novel features that emerge from complex systems where the individual components wouldn't be able to produce those features in themselves.

And when I was driving over here, I started thinking again of Manny Farber's famous essay, *White Elephant Art and Termite Art*, where he champions film character actors as an example of termite art because they're focused on the localized and the personal

and I thought that was interesting overall because initially I was going to set up this conversation as a talk about how in the ‘60s and ‘70s there was this conceptual turn and people moved from object to process based art.

And as I was thinking through it, I realized that many of the artists, who did ultimately start moving towards process based art still had this sense of the *Gesamtkunstwerk,* the total artwork, this totalizing impulse. Many of the artists today even among ourselves have this sense of a more personal termite-like art practice where there’s less of this need to ultimately synthesize and totalize.

And I was wondering if you felt the same way ultimately and whether or not you think maybe there’s greater freedom with our generation than earlier generations of conceptual artists?

Casey Reas: I’ll pick up the mic.

Andrew Max Maxwell: And that was definitely a random walk, sir.

Casey Reas: Yeah, And I don’t know if I followed you all the way through the walk. I think we have ways of self-publishing, ways of self-distributing now that are very powerful that really weren’t available to the same extent then.

And so, rather than needing to go through the gateways of official institutions or curators, we have the ability to put our work out there and to make it accessible. I think that allows for maybe this more fragmentary nature of work. We can do small pieces and put them out. They don’t - not everything has to be a part of - a larger, body with a unified theme. We can really experiment and do different kinds of work simultaneously, I think, to more of an extent.

Andrew Max Maxwell: So, you think that ephemerality or the ephemeral gesture is more frequent now? I mean, it seems like even people that were interested in process based arts in the past were also very concerned with the document because ultimately, they wanted to preserve some sort of fixity or some sort of permanence with the document. But I know at least in these sort of interesting currents within literature now, the ones that I'm interested in now, there is this greater tolerance for impermanence, ultimately.

There are a lot of people publishing in PDF form exclusively online, sometimes only publishing for - keeping the PDFs up for - a short period of time, on the Snap Chat model and so forth. And that seems, in many ways, very liberating - right? There is the notion that the text could just disappear ultimately and that's okay.

Casey Reas: Yeah, I feel like that kind of work emerged from that time, and in a very enthusiastic way. And I feel like now we're all the children of that experimentation, but there are more and more people who are building on top of that shift. I think there's definitely more happening.

One idea that I like from Sol LeWitt - is the idea of the conceptual artist with a capital C and a small c, where a Conceptual artist with a capital C is really interested in the ideas first and foremost and has really no interest in the aesthetic outcome, and what he referred to himself as being, a conceptual artist with a small c, where actually the process and the ideas are important. But in the end he really does care about the object, and the material, and the surface, and the form of the piece.

I think it's a continuum, but there were those interesting breaks at that time, and I think that was a forerunner to this idea of moving into the immaterial which was the foundation we needed in order to really move into software as its own unique medium for production.

Andrew Max Maxwell: I guess my interest in code is more Bletchley along the lines of communication theory and sort of balances between intelligibility and encryption. And then perhaps the notion of code is maybe the algorithmic definition where it's a set of instructions that organize a process. And I think that definition has certainly been generative for generations of artists.

I definitely backed into code and to software development from a completely weird angle. I was studying linguistics, in natural languages in school, and I always had this dream as a child of being a lexicographer, you know, somebody who wrote dictionaries, But, strangely, in my late 20s, I was on my way to the Art Institute of Chicago to join their sound program, I came down to L.A. to say good-bye to some close friends, and took a job off the UCLA job board for a startup, that at the time was called *Oingo* (which would later be called *Applied Semantics)* that was advertising for lexicographers.

Little did I know at that point that they were actually building out databases, like concept maps. They were working on Princeton's WordNet and trying to build early semantic web technology; working on disambiguation and language..

There were many things that they were working on that were very exciting for me and ultimately I worked for a few years on building out ontologies, and then that company four years into the process got acquired by Google where I started working on, computational linguistics and then became basically a technical program manager managing software development related to text classification.

And, now I, for better or worse, work largely in the advertising space doing, as you would expect, user modeling, trying to understand long-term interest based on how people traverse the web, and I am still working developing linguistic software, but often applied towards recommendation systems and things like that.

Casey Reas: One thing I would like to talk about tonight is the public or community work that we both do. I think another thing I'm really excited about tonight in the talk is that we both work with code, we both work with systems, but I'm a visual artist, while Max works with language, and I know that whenever I have conversations with people who work with some of the same ideas that I do but work with language and not in the visual arts I just find it really interesting to compare and contrast the differences between the two.

My background is really in drawing. Drawing's what I spent all of my life doing really before I started to code. The reason I started to code was because I wanted to be able to draw in a different way. I got really interested in artificial life and artificial intelligence, and I wanted to use those ideas as a way of making marks, and then the drawing became less of a manual action and became more of an automated or mechanical way of drawing with thousands or hundreds of thousands of different autonomous agents that I give behaviors to. So, that was my way in.

And I didn't really start programming until I was about 26, 27 and that took me to a lot of other ideas. And the work that I do now, the work that I've done for the last 15 years, has all fallen out of this new way of thinking that I learned through wanting to pursue these different drawing projects.

It's not highly personal, but it's slightly personal; for me, learning how to code was huge, it was a struggle. I had tried to do it a few different times, and eventually the need to do it was so strong that I suffered through it. I took computer science classes at evening school and that led me in different ways. And then, eventually, I went to graduate school at MIT where everything really started to happen in a more mature way. But it was really painful.

The way that you learn how to program - and the computer science classes for someone who's a visual artist - the two don't match at all. You end up doing math problems, you end up working purely with numbers and text, and what I really wanted to make were images. And, so, after going through that as a graduate student at MIT, our research group there was called the *Aesthetics and Computation Group*, which was led by a professor named John Maeda. There was a history there of writing tools for artists and designers.

A fellow researcher at that time, Ben Fry and I started making a programming language called Processing. The idea of Processing is to make a programming language which is entirely different from the way programming languages are usually built because it's all based around images and drawing in the visual arts. Instead of working with math and text, you work with lines, you work with color, you work with animation and interactivity.

Through using this platform for thinking, it's more how I think of it than a programming language, you're able to learn and follow all the ideas that I think are really interesting and powerful from this sort of conceptual area that computer science pioneered, but you're always doing it within the context of the visual arts.

Andrew Max Maxwell: One of the things that seems very interesting about Processing is that it is also this platform for the introduction of programming fundamentals to an audience that, like yourself, may struggle ultimately with learning programming, and that in this way it's also very constructivist; right? It's learning by doing and the reaction that there is no sense of, again, the fixed artifacts. You're constantly dynamically learning

Casey Reas: Yes.

Andrew Max Maxwell: That seems like a very optimistic and kind of a wonderful thing that you've come upon.

Casey Reas: The goal was really to empower people to have this experience and to have a way in, that's interesting for them. But in regard to constructivism, *Logo*, which was the language designed by Seymour Papert, was a very important prior work that really led into how we were thinking about doing it.

Andrew Max Maxwell: Oh, yeah that's great. I love the notion of reflective, programming in general and Papert's work, too. Constructivism is big for me. Earlier you were talking about the distinction between, lower case C conceptualism and big C Conceptualism, and I often am thinking about the distinction between capital L Literature and the literature in the sense of a scientific body of text, a legal body of text.

The idea of a capital L reputational literature based on brands and authority control, is to me something to militate against. I like the notion of a peer contributed body of literature that we're constantly evolving, where the author, or the idea of attribution, plays less of a role. So, I think it's a fundamentally constructivist notion of literature.

We've got models of that within the web that have evolved in the last 15 years; the wiki and so forth. But if you even go back to pre-Stallman to the early days of the proto-free software movement, it was really about the hobbyist and ex-hobbyists and experimental learning and there was a lot more of this allowance for peer education, spontaneous mentorship. That's a very exciting era of software development that would be nice to import to the arts.

Casey Reas: Artists have been working now with code for nearly 50 years, and there were a lot of programming languages made for artists through the 1960s, '70s, and '80s. But the one thing that was really missing there was the internet as the

way of distributing that information. And so things that we really take for granted now, things, like discussion boards, things like IRC, that's really what's enabled this work to carry so far, I think.

And, for example, when you're putting your work online, the source code for that work is then available for everybody to see through the web browser. And so, people are always looking at how other people are doing things, learning from that in a really autodidactic way; not copying it directly, but learning, and then applying that to their own work and really making it their own. So I think that that the network condition has accelerated this tremendously.

Andrew Max Maxwell: I was actually going to show a slide, there's this conceptual artist some people here in the room are familiar with, Kenneth Goldsmith, who says that the internet is the greatest poem ever written, only none of us can read it ultimately, so as a tribute to Aaron Swartz, a year ago he attempted to encourage everybody to print out the internet, which is an imaginary solution to an imaginary problem.

Casey Reas: It horrified me.

Andrew Max Maxwell: It horrified you, yeah. There's a lot of things about Goldsmith that horrify people and probably should. He's a proponent of the idea of uncreative writing and ultimately he says that literature now really only exists as data being poured into differently sized containers. I think on the one hand, there's this data liberation component of this that is exciting.

But on the other hand, I think there is a role for the creative among us, and we don't have to get to the point where we're just mere bureaucrats with language. But the idea that the internet is this wonderfully sort of spontaneously evolved construct - it's like the world is replete; right? It's a very optimistic notion of the Internet at the same time.

Casey Reas: We mentioned the Calder sculptures coming in. And so for me, a really important text that got me somewhere, that really opened ideas that allowed me to think about my work in new ways was Umberto Eco's *The Open Work*.

That book is about the idea of a text; a work is a text where, of course, that text is open, with every different interpretation for everybody in the room, but I think the idea of the open work is that the text actually can change, which leads to then different interpretations.

And almost all the examples he used were examples from language, examples from music. For example, a lot of the experimental scores from the 1960s were used, but the one example from visual arts that he used was Alexander Calder's mobiles, we have a few out there, with the idea being that the work is really a set of elements with relationships and how the pieces are able to move.

And then as those elements shift, it's continually remaking and reforming itself. And that's in a very material form. I use that example a lot as a way of explaining the idea of the system, but then moving into a more immaterial, a more software based version.

Andrew Max Maxwell: Yeah, To go back to literature, I think some of the most interesting currents in literature in the last few generations have thought similarly, ultimately trying to think about, literature as a conditional system ultimately a set of parts that, ultimately we don't entirely control.

I think there's an analog here; that famous almost canonical text of new media from Lev Manovich, the data base as a symbolic form, We've moved away from this linear way of thinking to a collection, ultimately, to an unordered list. And ultimately, at this point, the

index to the collection is becoming more prominent or more interesting than the content itself in some ways.

I know that in my own writing I moved progressively from long form verse, because I became less interested in the poem as an object, an *objet d'art,* towards smaller and smaller units because I was interested in minimal definitions of intelligibility or sense, but I began radically annotating every single thing that I wrote to the point where the taxonomy or the improvised taxonomy becomes more interesting than the content itself. So that's the building of a spontaneous system, spontaneous index or the metadata itself to the content.

Casey Reas: Yeah, the system itself becomes what's of importance rather than what it produces.

Andrew Max Maxwell: Yes.

Casey Reas: I think that was a big shift; that software really is able to continue moving forward with.

Andrew Max Maxwell: Yeah my favorite sort of recollection, or I guess this is entirely mythic, my favorite recollection from Joseph Beuys was when he was talking about the Nazi book burning parties in 1933 in Kleve. And he said that he claims to have been at one of these book burning parties and rescued Linnaeus' *Systema Naturae* from the flames. So I think that that gesture in a chaotic moment, when something could go terribly wrong, where you're rescuing a system or a taxonomy from this dynamic chaotic moment, is a wonderfully sort of beautiful gesture, a beautiful moment to be replicated again and again in the process of creating art.

Casey Reas: Yes. Well, we have this idea of the object or the material and the totally immaterial, and, of course, it's a little bit false because you always need some sort of physics involved in order for color to be to happen, et cetera, for these phenomena to be.

I think that it's the de-emphasis of the material that's important. So for many years, I would never show work on something like this [*pointing to the video screen]* because it was such a physical object, and the presence of the object was such a strong part of the work. Everything was always projected so it was purely light. Showing work that was continually in motion, always moving, always changing, always iterating, so that it was new permutations that were coming out.

Like a lot of my work, it's different from one second to the next, and where it was one second before will never be retrieved again. They're all continually unique moments. But, at the same time, I do produce photographs, prints and work that is static, that is an immutable object as well.

Peggy Weil: Do you consider these works to be performances in the sense that they are live and singular instances in time?

Casey Raes: I don't. I think I get myself into trouble because I don't know enough about performance…

Andrew Max Maxwell: I think for me, I'm always a little uncomfortable with the notion of version control in terms of the text. So, for 20 years, I refused to publish anything, although I was writing regularly and also hosting a literary service to the *Poetic Research Bureau* for traveling scholars, writers, and so forth.

But let's say if you publish or distribute editions of my writing, and have five or ten copies, but every time I would print them, I would forcibly change the contents so that these would only exist in a sort of *samizdat* or made art aesthetic; like,with Walz Berman's *Semina* aesthetic where you basically create an ecosystem of friends.

Friendship is very critical, and you're exchanging ideas among friends, I think maybe it's just because, over time, you want to reach out to more and more interesting personalities and you want to have to dialogue. Sometimes the distribution system of traditional publishing is the only way to do that to some extent. , So, my values have broken down and I've, published a few books of late.

But even then there's sort of a gesture. This is the latest one, *Candor is the Brightest Shield*, and I basically created this gesture against version control; this is a multiplication table of animals where, you basically have a hybrid among each one of them. You can imagine what the animals would be on this side and in this side, but it's just again, militating against the version control of the book, even if it's only symbolic.

I very much enjoy, actually, reading the work because it's never the same way twice and the idea of the voice text, the inner intersection between a text and consciousness, is fascinating to me. So, trying to perform the work in a voiced way differently each time is fun.

Casey Reas: I think it can be seen as a small shift, but I think a really important one in the way software enables doing additions where each piece in the addition is unique through parameters, through permutation, where, for example, with making prints there's maybe 12 pieces in the series, but each one's different from the other because the software is able to produce variations within the theme, and the moments that are captured from the larger realm of possibilities or the search base of the parameter space that's set out by the system that's being built. It's small shift, but I

think it's an interesting one that really was possible in a different way before, but scales in a different way now.

Andrew Max Maxwell: I think there's a time right now where there's movement between agency and control and nondeterminism in randomness that's quite interesting that allows a sort of space for disownership.

I'm very interested in this idea of disownership, It's a very trendy concept among tech startups right now, right? I mean, we're going to give up our cars, it's all going to be ride sharing, we're going to share our homes, ultimately there are now these communities for even apparel exchange, , For me, who's politics are definitely on the left libertarian side - I'm an old schooled 19th century libertarian socialist - this is the very utopic aspect of Silicon Valley. But, I would love to find a space where we could take that notion of disownership and import it into arts and literature.

I think we have, to some degree, maybe in media arts and in visual arts, less so in literature. It's still an ecology of the author and the authorial representation outside of these radical communities of literary anarchists who are publishing entirely online - who I love. These are my friends.

Casey Reas: With software we have this idea of building common or shared tools. You know, the reference in terms of our largest society is that we need shared infrastructure in order to make things work better for everybody.

People are making software tools that other people are allowed and able to use. I think one of the goals in this community has been to lower technical barriers to allow people with really good ideas to be able to realize them without six years of computer science education or without $100,000 in collaboration engineering time and things like that.

Over time, the tools and the ability to do things at a high level, like, know where somebody's face is or to be able to track a card on a table, those kinds of things, which were at one time really interesting research problems for computer scientists, now are something that people have a really low barrier way to begin making things.

And with open source, with free software, people are, as you know very well, sharing software, sharing code, putting licenses on that code that requires them to be free; essentially taking away a freedom in order to encourage, to really enforce legally, that that software will always be open and putting that work out. And it's been building and aggregating to the point to where students now can come into a program, learn, I think, basics within a semester and be able to make work that they're really engaged and interested in making.

And I think that's all been because of the open source aspect, it's all been about groups of artists taking the responsibility, to make it happen, and then sharing back to the community. And the larger software corporations, the Autodesks and the Microsofts of the world, have had a very different approach and so we're now really trying to make tools by artists for artists. Over the last decade, it's been a tremendous change.

Andrew Max Maxwell Yes, I've seen that change even within Google. You know Google has these famously strict hiring practices. But, early on (I've been there for a long time now) they were still hiring people largely from Ivy League colleges, from maybe one of 30 colleges. But over time there's been this widening of the net towards trying to find these brilliant autodidacts, really what the software industry was built on, this sort of community again.

I think that's a really great trend, ultimately, and now some of the larger software communities are investing quite a bit in public education as well and trying to create, or seed communities of potential software developers as well.

Casey Reas: Yes. Right now there's a larger project called *code.org* and every December now there's the *Hour of Code*, which is a huge initiative trying to encourage high school students and middle school students to take an hour out of their day and school life to explore what code means.

But the tutorials that we make are arts focused and arts oriented, but 95 percent of those tutorials are, again, engineering focused, and what we're really interested in doing is making a new culture where we don't adopt the software culture from the other areas, but we really try to grow one from scratch and totally integrate it into things with the idea being that code is no longer for the technical.

It's now a way of thinking and it should be a part of the humanities, it should be a part of the arts. It's a part of music and it's really just a way of thinking and making in the 21st century rather than an obscure geeky technical sort of genre.

Andrew Max Maxwell: It's interesting that there's still resistance to that. In education in K - 6 levels or K - 5 levels, there was this STEM movement for a long time and now that has been permutated to become a STEAM movement.

Andrew Max Maxwell: STEM stands for Science, Technology, Engineering and Math. Because of progressive defunding of the arts in public schools, now there's the STEAM movement where we'll add A, for the Arts, which is a great idea; right? Because ultimately you want to find applications for science and technology as well and there's a sense of making and craft that is important, I think, to engineering.

But I'm on the Community Affairs Board at Google as well, where we're often giving grants to local schools or trying to to further the mission of STEM in local education. But even as I've tried to explain the STEAM movement to software engineers at Google,

there's some resistance. It's just like this is just sort of a hokey way of funding arts education. But ultimately, I think it's critical. We need a more generalized sense of the maker rather than just the engineer.

Peggy Weil: Do you want to elaborate on your comment earlier that your interest in code was more "Bletchley" in the sense of ciphers, as it relates to your writing?

Andrew Max Maxwell: First I should say, I would never call myself a developer; you know? I'm an autodidact who's developed skills in Perl and Python like just basic scripting languages, some JavaScript, I use SQL and so forth, but it's entirely pragmatic for me in moving data around and doing some basic shell scripting and that sort of thing.

I more manage software processes, but I've always been fascinated with Bletchley, and the early code breakers. This book *Peeping Mo* is actually dedicated to Leo Marks, a lesser-known code breaker who was also a script writer and wrote the screenplay for *Peeping Tom*, the Michael Powell film, which kind of destroyed his career. He would create poem code; effectively, he would work with book code and send messages into the field to these agents and these resistors who were in these terribly dangerous situations. And the notion that they would be carrying a small poem with them that also carried, or further transported, an even deeper message that was more critical was, is, just a powerful conceit for me.

As I've gotten more into aphorism, I think about the ambiguity and the ambivalence in all texts and how a single piece of text can carry multiple meanings, multiple senses. And so that's why I say, that is effectively sort of the Bletchley model of code; right? An ambivalent piece of text that based on one's level of access control; one can access the signal or get purely noise.

Casey Reas: Well, I think, Peggy, in regard to your question, it's all those things. It can be used for all those things. It general, like language; you know? Language can be a law brief, it can be a medical report, it can be a poem, it can be a novel, it can be everything. It's a way of recording thoughts. And I think code's the same way, so it can be used in all different ways, and has been.

I try not to think of code as a specific programming language or a specific technical infrastructure; I think of it really as a general way of thinking. And lately I've been thinking of it as just as a list of things. So, those things are things like statements, ,variables, conditions, functions, objects, arrays.

And these are all models for how to structure a system, and they're not really specific to how to operate a computer, and I think for me it's important to think about it at the level above, like a specific machine implementation. But in the end, I spend hours a day writing code in order to make the work I want to make. I need logic and process and conditions.

Peggy Weil: Do you want to comment on the open source movement as it relates to collaborative authoring, and in particular the collectively authored book, *10 Print?*

Casey Raes: *Processing,* the really quite large now open source software project that I built, there's no compensation at all for that. It's all done as a volunteer effort in that particular case. This book, *10 Print* - and I see Mark Marino over there, who is one of the 10 co-authors on this book - was written as a wiki. There aren't 10 separate voices, but it's written in one voice. Different people contributed with a lead on different chapters and then different people did passes of edits and final copy edits. And in the end, there are issues with that because some people take more responsibility than others, and there's more of their individual ideas in it than those of other people.

But everybody knew that going into the endeavor. So I think it's just a matter of having the right expectations when you start a project like that.

Andrew Max Maxwell: I spent some time reading *10 Print* and there's definitely many, many voices in there. I was curious, you must have written the chapter on chance and randomization or did you not?

Casey Reas: I think I also wrote that with Mark and also with Nick Montfort.

Andrew Max Maxwell: Okay.

Casey Reas: Well, I guess, actually, we tried to have one voice. And the sidebars were written by individuals.

Andrew Max Maxwell: I'm interested in that, the trying to write with one voice; like, how did you go about that, or what were you attempting to do?

Casey Reas: I think it was more of an experiment to see if it could be done, and if we could do it, why would we want to do it? Just to have an artifact like a stake in the ground as something to examine; if it could be done and if it was a good idea.

Andrew Max Maxwell: It's a really fascinating book, by the way, you should get it. There is, something, very utopian about 10 authors ultimately trying to create a collective text and publishing it without attributions laced throughout the text.

Mark Marino: Well, every utopia is a dystopia seen through candy-coated glasses. As a writing teacher, I could say a lot about writing with one voice. It was a

challenge, but when we submitted to be part of the collective we signed on to be part of a piece of conceptual writing -- to be the 10 of *10 Print*. And yet we weren't all equal. If you look at the book -- which is available for free for download -- you'll see that Nick Montfort's name is listed first because he was the one who made sure that it happened. There wouldn't be a *10 Print* without Nick. So in response to your comment there was a sense of payment through the way we gave credit, since Nick's name would appear alone in a citation because only the first name would appear.

Casey Reas: Well, actually you bring up another really good point, which is that there are other economies other than the monetary economy, and that a lot of this work, the software that is free, meaning that it doesn't cost money to download and use the software, but it's not really free in the sense that the bargain there is that you're expected to contribute back in whatever way you can either through making, posting bug reports or helping other people out.

So the communities just operate on different economies. And I think another important thing about these kinds of large collaborative projects is there's an idea that's talked about, which I think is largely true, of the benevolent dictator; you really need somebody who has the vision and is the sort of gatekeeper for what the core of this project is, and that happens, well, in the majority of these cases. In the Open Source Software project that really worked, that presence exists there. And Nick (Montfort) was the presence for this book; with the Processing language it's myself and Ben Fry, arguing internally to really decide - defining what is the core of the project or not.

Mark Marino: Stakeholders, I'd call them. Someone who has a serious investment in the project seeing the light of day.

Andrew Max Maxwell: It sounds like you guys were almost evolving in a variant of agile software methodology, or software development. You guys needed a scrum

master, right? You're creating these temporary, priorities, and you're in smaller groups, collectively determining what must be done. So that would be interesting to think about. Agile is a form of collective authorship.

Casey Reas: Yes

Audience Comment: How would you characterize your process at the beginning of the work?

Casey Reas: I personally work in iteration, and I choose not to think a whole lot before I start making. I don't think and then make. I think through making, and I just start doing what I can do at the time and that leads to the next thing, the next thing, and the next thing. And I build these trees of work where one idea will spawn other ideas, and that idea might spawn other ideas, and I have this huge diagram of a different potential. And then I kind of prune the tree and then follow one to the end. And there's typically multiple Aha moments throughout that process, and there are also periods of slog; of making and being really frustrating with what happens. So, that's how I usually go about it.

Amy Heibel: How would you describe the role museums should play in collecting, preserving and perpetuating code-based works of art, particularly in terms of managing obsolescence?

Casey Raes: I think it's totally necessary, and I'm worried about losing decades of work because museums haven't been on top of that one in terms of software. But in reality, I personally think emulators happen because people want to keep running that software, and that seems to be the way these things are being preserved in the wild. So we have people like Brewster Kahle who made the *Internet Archive*, who's preserved so much, and thank you to him for doing that.

And I'm really excited for the institution, for me, that's the role of an institution like LACMA, preserving our cultural heritage; when you assess you bring on a work into the collection, it's your role to continue or to preserve that work for perpetuity as much as that's possible.

Amy Heibel: That's my understanding of it. And so, the question is, how do you do that with software?

Casey Reas: Yes, I think it's also really the artists who need to be very responsible as well. Because we need to be using open standards, things that aren't proprietary and closed by specific corporations, because it's likely the source behind that work will never be in the public and never really be accessible. And so I think if artists are doing their part by working with the open standards that then can be reproduced and emulated later, then hopefully that will be what we need to make it happen.

Andrew Max Maxwell: I feel the same way. Ultimately, I'm much more worried about the privatization of knowledge and the privatization of various repositories that I am familiar with - the notion of there's going to be too much abandonware or something; whether or not we're talking about artistic artifacts or digital artifacts or code.

Casey Reas: That's actually a really great website, abandonware.com.

Andrew Max Maxwell: Abandonware.

Casey Reas: They have all the DOS ROMs, yeah, for DOS simulators.

Andrew Max Maxwell: Yeah. I'm really fascinated with these institutions that have been popping up to preserve gray literature; the notion of this sort of institutional literature that is not subject to commercial publishing. You can think of white papers, you can think of patents and things like that. There are now these organizations that are entirely collecting just old institutional video and stuff. There's actually somebody in the audience here, Jim Fetterley from the video mixing duo, *Animal Charm,* which is really great. They're real rescuers and fetishists of industrial video and what would be the gray literature of video. If you like, remix culture, they're sort of the Kanye West of remix culture.

Casey Reas: One approach is to put all your work online and to have all your work available for a browser, and that way it's there for everybody to see. What you're giving up when you do that is control over how the work is viewed. But some works really need, you need to be in the right space and in the right mind to really be able to receive or interpret the work. So I think it depends on the piece and what the piece is really about.

Andrew Max Maxwell: You do displays, the artifacts of your processes, in the exhibition environment and museums. How do you feel about that? Ultimately, how do you select, if you have an ongoing process, some of the things that you posted on Vimeo? How do you choose a snapshot from that process to display in a physical environment?

Casey Reas: Well, hmm. I think I'm going to pass on that question.

Andrew Max Maxwell: Oh, okay.

Casey Reas: Yeah. I could do that; right?

Steve Anderson: You made a very provocative and interesting comment earlier about the transition in your work from presenting it in a radically dematerialized form -- as just electrons on a reflective surface -- to being more accepting of the display apparatus of the monitor as a physical part of the presentation. At the same time, your work has undergone a transition from foregrounding the underlying technological process by which your work is created -- specifically, your early works that were presented in a diptych or triptych form, where one screen presents the "instructions" being executed, another screen shows the background work being done by the software and a final display screen presents the visual output of those processes. I'm wondering if you see any relationship between these two trajectories in your work?"

Casey Reas: Yeah, I think for people who know me (most of you don't know me, I know some of you and it's nice to see the people that I do know) I spent a lot of years writing, the goal was to write these very concise sets of instructions that define behaviors that would be applied to these basic elements of geometry. And then, from that, these visual environments would emerge. And so, in that situation, the work was really about the relationship between that short text or set of instructions and the way that that process unfolded in time.

And I guess what I found is that right now I have ambitions for doing works that are far more temporal and visually involved. And right now I'm working with all images from mass media, and parsing those, interpreting them in different ways and that work could never be broken down into a few lines of instructions. And so, I guess my focus on that process in making it explicit in the exhibitions, needed to go away in order to pursue this other kind of work that I really needed to pursue.

And in terms of displaying on objects like these, these objects have gotten, so much better in that they're starting to disappear now; I feel like the material environment is

catching up with my ambition of how I want to present the work. Like, I'm still waiting for the paint on the wall, that sort of infinite resolution pixels – and it's not here yet.

Question from audience: Do you begin by visualizing the works or sketching as part of your process?

Not always. There's kind of a two-step process, the first is sketching on paper. And in my own work, it's not this image - this, you know, 'a piece will look like this,' it's more like, 'this is the system and how the different pieces and parts relate.' I'm usually far more interested in the relationship of things, that's the starting point of things of the pieces. Then, it goes to a second step, which I refer to as sketching in code, which is writing little pieces of code as fast as possible in order to produce visual results.

And so it goes along. But really, it depends on what kind of body of work that I'm producing. Sometimes there's a period of research and then there's some really strong ideas about what it should be, and then that gets implemented.

For example, we spent, on one project, about a year doing collages first and then figuring out how to turn that into a code-based system. And then other times it's more just following your nose and intuition and seeing what literally emerges from the process. I do a lot of collaborative projects. An example is *Chronograph,* which was for the New World Symphony in Miami. My collaborator on that was Tal Rosner.

Audience Comment: Coding is sometimes considered a craft, what is your sense of the relationship of the coding as a craft and art?

Casey Raes: That's true. But many of us, myself included, strive to make works of art and I think of craft in a little bit different way. I think craft is always a part of making and a part of the studio, and different media materials have different kinds of

craft. And for me, I think of coding as the craft. Like, if I'm doing ceramics, then how to work with that material is my craft, but I utilize that craft in order to realize this larger vision or goal and idea. And, so, for me, coding is my craft, but that's not the end. It's the means to the end. And I think really if you think about coding as a craft, now that comparison can go pretty deep.

Andrew Max Maxwell: Yeah, This is a really tough question. No, it's interesting. I have a good friend that thinks about this a lot. We were thinking about this in terms of potential renaissances. I have this friend, Aaron Kunin, who's both a scholar and a poet, who was wondering why there has not been a renaissance. And when he thinks about renaissance, he's really thinking about the ability for a large population to produce a lot of art as opposed to craft. And he was wondering why there has not been a renaissance in Portland, which is the great craft city, but there have been renaissances in, say, Kansas City. In Wichita there's been many, many interesting poets, for instance, that have come out of those two cities that are making great art.

But thinking in terms of software, I would agree that ultimately, coding is form of craft, and taking it to that level of art is the great mystery.

End of recording.