



Identifying Technical Art by its habits

Robbert-Jan Brems
Senior Solutions Technical Artist at  unity

Hello wonderful people.

Before I start, I would like to quickly introduce myself :

My name is Robbert-Jan Brems and I am from Belgium, currently living in Vancouver.

I graduated 7 years ago as a technical artist from a school in Belgium called Digital Arts and Entertainment, and I have been a tech artist ever since.

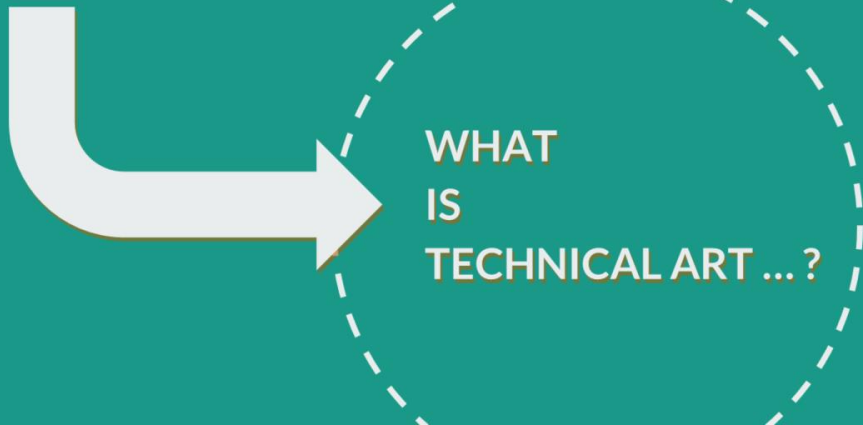
I worked for companies like EA, Eidos-

Montreal and Codemasters, but now I am a Senior Solutions Technical Artist at Unity. If you are curious on what that ob means, feel free to come talk to me after the talk.

In my free time I teach people the change mindset. Whether they have a hard time dealing with change or they want to introduce some more change in their lives.

I use habits a lot a lot to translate the change mindset to people, which is what inspired me to do a talk about this for technical art.

The main problem



We already had a previous talk going more into depth about this topic. So I am not going to spend too much time about what is technical art.

What I am going to be pointing out is that the question of “what is technical artist” is really just a source of opportunities.

Just imagine this problem space to be visualized as a big open circle where we are going to explore underlying problems

that hold new opportunities.

Sub-problems



In the problem space of “what is technical art” lies many smaller problems that describe more the bigger picture of what we are dealing with.

The ones i listed here, are the ones I had personal experience with and always love using as an example.

- How do we currently recruit in already a hard market? Are we having a hard time to connect with potential people that might not be

even aware of technical art in the first place ?

- How do we market ourselves inside of the company ? Aren't there other departments that could benefit of the technical art mindset ?
- How do schools currently train future potential? Do they even understand correctly what technical art stands for ?
- How to support a rich environment in our team for self-development and career development ? How can we measure growth ?
- And many more subjects that falls into this space.

All of these sub-problems gives us more an idea of what our current work environment is, but also its potential to improve.



What connects
these sub-
problems?



Habits

Whether **good**, **bad** or whatever you want to call them!

We all know what **habits** are. But we don't always know what habits we currently have whether good, bad or whatever it is.

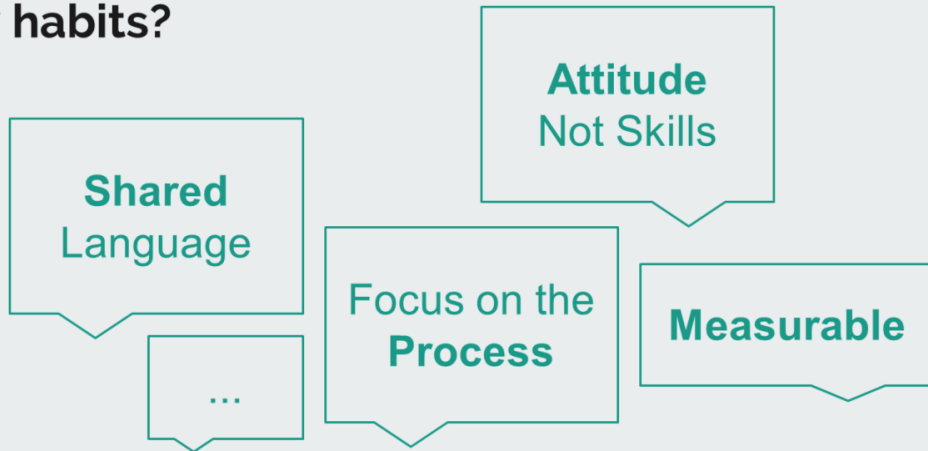
This moment of reflection does not happen often.

Habits do seem to appear in those well know articles : "**list of top 10 habits for entrepreneurs or successful people or ...**" But it tends to be missing in our daily conversations, so hopefully

this talk can be a good start.

Right now, habits will be the building block of what we will use to have a conversation about technical art and to explore of what these sub-problems all have in common.

Why habits?



There are many reasons why habits are a great way to describe an identity:

- Focussing on the **process**, not the end result. If you have a good process, the probability of having a good end result increases. Great for dealing with uncertainty
- Training towards good **attitude**, not directly hard skills. You can learn all the skills you need with a good attitude, but teaching a good attitude

is harder.

- Creating a **common language** out of habits, to improve communication between different personalities, departments, ...
- If you can't **measure** it, it does not exist. It makes it easier to follow up with people their own progress or measure your personal progress.

And many more reasons ...



Using habits
as a **safety net**
for constant change



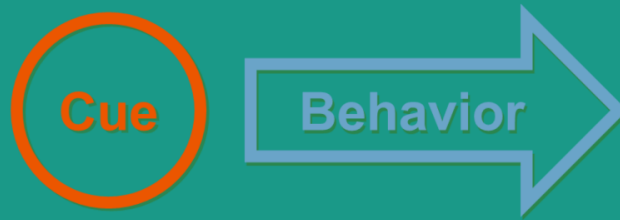
We are using habits to use as a **safety net** to **self-reflect** on **constant change**. This allows us to have a **system** to deal with change that can happen all the time. Whether chosen or just randomly happening to us. Strong habits allows us to defend ourselves against change where we would react emotionally and lose a lot of our time and energy. It allows us to place these experiences in some kind of context and see much quicker what the

opportunities are.

Image :

<http://www.westendmagazine.com/wp-content/uploads/2017/03/Georgie.jpg>

How to build better habits ?



The easiest way to describe habits are with a cue and its linked behavior.

Cue's can be like emotional cues, physical responses, environmental observations, ...

Example : morning walk, anger, confusing, nervousness, overwhelming, lunch time, ...

Behavior examples :

Learning something new (Mooc's) = discomfort, teaching, breathing, writing

down, going for walk, clean up email, ...

But more related to technical art:

- Whenever an artist comes to ask for help at our desk (cue), you think first, who you could bring from your team that could **learn** from this experience (behavior)

Result : Including the team, lowering the risk by spreading knowledge, educational tool to ramp-up juniors and introduce them to the team.

- After working for x amount of time and you are feeling tired (cue), take a break but ask others to join to enforce the importance of a good break.

Result: People don't tend to take their breaks, even though this is quite important. Allowing to do it together can create a sense of community and giving an environment to talk about problems that would not be discussed otherwise.



What kind of habits ?

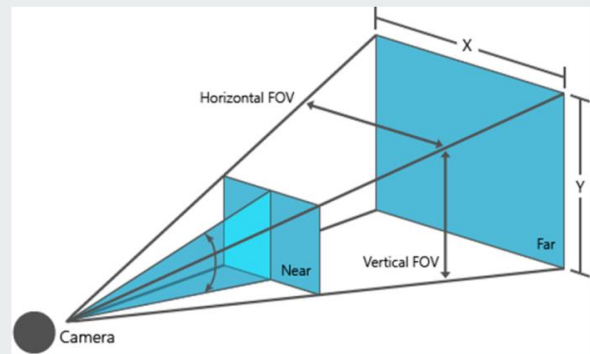
- Reframing
- The Overview Effect
- Breadcrumbs
- The Power of Ambiguity
- If you can **teach** it, you **own** it

These are 5 **fundamental** examples of habits when dealing with technical art in the video game industry.

There are a lot more, but for me I always go back to these habits when doing general problem solving in an always changing environment.

Reframing

- The 5 Why's
- Underlying problems
- Using other **people** as different perspectives



For those that are not familiar with the word reframing. It is all about changing the perspective in which your problem lies. You cannot change the problem, but you can change the context in which the problem exists.

In psychology or neuroscience, it is also called cognitive reframing or cognitive restructuring. But I am going to save you from the scientific definition.

As a tech artist we deal with **uncertainty** in every project we work on. I have always seen myself as the one creating opportunities out of this uncertainty.

Uncertainty holds many challenges and problems. Allowing us to reframe these challenges makes use see new opportunities for improvements on how we make games.

Some things in these perspectives you can control, but some you cannot.

One of the techniques I use to see a different perspective on the problem I am trying to solve is to ask myself **the 5 Why's**.

For example: An artist comes to me with a Perforce problem ...

- **Why** did this perforce problem happen in the first place ?
 - The person did not have a good understanding of how Perforce works and made an assumption

that caused a new problem.

- **Why** does this person not have the right knowledge ?
 - Because we either assume they know, they should learn on the job or we just never trained them as training is the first thing that gets cut because of time constraints.
- **Why** don't we have a proper training for all the performance knowledge we assume people have ?
 - We put this responsibility on their lead, but they are the ones always too busy.
- **Why** do we assume it is their lead's responsibility ?
 - The lead is supposed to be the one showing the examples and knowing what he is asking.
- **Why** do we assume that ?
 - Because this was never a proper conversation as we did not look for other opportunities as long term investments are hard to be

made tangible.

The solutions we went for :

- A tech artist made an introduction class by video and on our internal wiki. We made this part of their onboarding and did several checkups on how the training went.

The tech artist was the one paying for this issue in the first place, so they became the one empowering the artist to be more independent and not rely on their lead.

So the first problem was not where the opportunity existed. It was a couple of why's away from the beginning. We discovered the underlying problems and used someone else as a way to generate a new perspective to find new opportunities.



The Overview Effect

The overview effect is a term that comes from astronauts that describe this **shift of awareness** when they look down the earth from space and have this feeling of utter connectedness and constant change. A moment where you feel you are being part of something bigger.

We don't tend to take a step back to look again on what we are part of, and how our small contributions can make a big difference.

Especially for tech artists that deal with getting the performance right to run a game at 30/60 frames per second.

Being the bridge between all departments that have an influence on this challenge, we can see how the end result can drastically change by a submit from one of the disciplines.

For example, a designer changes the camera on a cinematic, just a little bit and does not tell anyone. What he did not realize is the because of the change, the hair of a second character with very curly hair, was taking up more screen-space.

Which in technical terms means that we will have to calculate more pixels of transparent hair meshes with more overdraw on an already expensive shader, with multiple lights affecting that area.

This problem only gets noticed several days later by QA who were asked to

monitor the framerate. By then design had already signed off this shot and we were told to fix it without changing the design ...

A great example on how a small change can cause a ripple effect as everything is so closely connected, but people tend to lose this awareness as they are totally focussed on their craft.

Having the habit that whenever you take a decision, you should reflect on the bigger picture and inform the right people that will be affected by this change.

Educating all the people involved, and see how their contributions do make a big difference is key to a successful collaboration to push the quality of the product.



Breadcrumbs

- Leaving a trail of evidence
- Action over time, allows us to see patterns
- Including people into the solution.



Reason

With this habit I would like to point out, you cannot always see the opportunity directly with every problem. Sometimes you need to develop a timeline that shows you a pattern you can work with.

Example

I get a lot of emails with requests for support. And every time I get an email, I

classify it so I can keep track how many times this problem occurs, if I know the solution or not and whether it is important.

This has a lot of time times to do whether this is a local problem, or maybe many more people are having the same issue.

Having this kind of trail allows you to support certain decisions you want to make, or just build a context around something.

And when you start asking your colleagues to do the same thing, you end up with a database of very useful information.

When there is a problem that we do not know how to solve, we go to this person their pc and do our investigation. We then reply to that same email this person send to our team, to continue the trail of evidence.

Sometimes we find a workaround, a

solution or sometimes we decide to wait it out to get a sense on how severe the issue is.

Once we have a solution we documented the problem on our internal wiki, with as title the error. And then we give access to everyone to this wiki so they can go through this wiki first before contacting us.

You end up empowering this person with the knowledge to solve their own problem and even support their own colleagues. Long live teaching **proactive behavior!**

Not only can you use this database of problems with their solutions to give super fast support, but you can also use this to send to the engine team on how we can improve the current engine to avoid these kind of issues.

Conclusion

So the habit out of this topic is, try to create timelines of all the problems you have to deal with. As they can reveal other problems worth solving.

The power of **ambiguity**



Reason

I show you a picture of a tree! Even though we both speak the same language, we perceive it different based on our previous experiences and bias. I might think about the challenge of rendering a beautiful realtime tree in a video game , or the other person might think about that moment that a tree fell on his house and he needs to either remove the tree or build a constructions

around it.

Many times, this difference in perceiving communication, it causes conflicts. One of the many reasons why tech artist are here is to translate the message from one to another, to get efficient to our mutual goal.

But why not benefit from this challenge where we currently lose energy of misunderstanding each other? When ambiguity occurs, we could see a **spectrum of opportunities.**

Example

A good example is when developing a tool. It's great to have a prototype as soon as possible out there to catch the perception of people to get their understanding on how this could be beneficial to them. As I am biased and I will need to fish for other opportunities that this tool can give.

I once made a feedback tool that allowed me to give quick feedback on how to improve performance. Quickly this tool was used by art and design to give feedback to their team, and then management wanted use it to create estimates and manage tasks to people ASAP. It seems that the idea of **feedback** has many differences within video game development. So by teaching yourself to be open for these opportunities, you can quickly improve your work.



If you can teach it,
you own it

Reason

It is really hard to **measure** you own **progress**. A lot of time needs to pass before you can see any result or difference. Especially when dealing with problems that people have not solved before.

Example

A great habit for measuring your own

progress is to always **teach someone else** what you have learned or what you are working on. Even in the beginning stage when you are exploring new knowledge. Whenever I start investigating a problem or developing a new tool, I start a new document to put down notes, screenshots and references. Even email of people I could contact or snippets from old slack conversations or emails.

Eventually this **scrap book** of information becomes an in between **medium** for teaching people about my current process and thinking style. Having this kind of feedback moment allows you to not only build your own confidence in what you are doing, but you include much more people into the process and you lower down the risk of only one person having this knowledge.

With this type of documentation that you build along the way, you will gain many

different insides you would not have thought about before. From my experience this increases your quality of documentation and brings a team mindset to have everything documented so your process is **transparent** towards the whole team.

I highly recommend to use primarily visuals and colour code your structure so people could be able to quickly scan through and get the essence of what you are trying to achieve.



If we reflect back on the sub-problem that we started this talk with, we can have more concrete image on how we use habits to create opportunities out of all these challenges.

When discovering a new habit in one sub-problem, it could be very useful in other areas.

Using habits as a common language to bring more structure when dealing with uncertainty allows you to get much

faster in a state of flow when developing computer graphics.

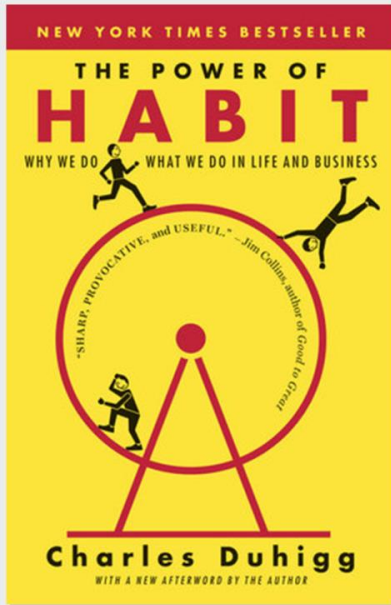
And a much better flow in your way of working, allows you to focus more on what really matters to you. Whatever part of game development it is.

robbertjan@unity3d.com

www.linkedin.com/in/robbertjanbrems



QA?



The Neuroscience of Reframing & How to Do It

Reframing should be in the water we drink, scientists say. But how does reframing work - and what happens in the brain?

★★★★★ 4.5 (1,296 ratings) 50,847 students enrolled

Created by Anette Prehn Last updated 1/2017 English English [Auto-generated]

<https://www.udemy.com/the-neuroscience-of-reframing-and-how-to-do-it/>