What happens if manjushri goes to a place where nobody recognizes him???? Is he manjushri?

Informatihedron/infoterma plane = like superfluid or plasma, in that at that degree of organization or homogeny in the overall system, state changes take less time (since the energy is so high, they all interact through similar fields) and in superfluid heterogeneity the energy is so low in each one that they can’t transfer anything to each other/interact and so spin forever and slide wherever uncontrollably

What’s INSIDE is PROJECTED OUT somehow - by duality - the waking dreamer, demon champion - the waking dreamer fights the demon champion, but the demon champion is/can be the waking dreamer, but they can also be olivus victory-ability…

Update log - version archive

Make iteration workflow - work with Worksuite

Must double check whether or not something does instantiate another before claiming it is an instance of it

Informatihedron/infoterma plane = like superfluid or plasma, in that at that degree of organization or homogeny in the overall system, state changes take less time (since the energy is so high, they all interact through similar fields) and in superfluid heterogeneity the energy is so low in each one that they can’t transfer anything to each other/interact and so spin forever and slide wherever uncontrollably

Add in encryption and decryption language below

Add in emergent wasteland hallucination collapse and expedient hallucinatory sanctuary emergent engine collapse

If subject has immediate perfect sanctuary system answer for task is false it instantiates a [Emergency Engine]:

Context+ Perception=>worldview

Worldview—> abstraction

Abstraction —>

Knit + target (new metaphorical is\_a embodiment entity and is part\_of a metaphorical class that is a allegory for the environment of the subject (entity + superdomains) (= emergent engine)💡 )

Emergent engine => storyworld HJ instantiating events (= Synthesize Complete HJ POIO class!)🌈

[Building Concatenations]:

POIO -> reason by analogy

Reason by analogy -> allegorical complete mapping of POIO HJ processes to real entity variable in problem related to task obstacle => POIO has polysemic is\_a mapping to obstacle entity via allegorical is a suppositions about the subject worldview -> honest mapping to memory (beliefs) -> sanctuary vs wasteland analysis => novel emergent perspective for persona 💡 -> if sanctuary, accept perspective + skillchains, find solution; if wasteland, reject perspective, rigorously inspect and analyze skillchains related to wasteland emergent => honestly measure and admit wasteland properties and self-faults => investigate self and environment => target flaws, prioritizing ignorance of sanctuary system of embodiment => analyze hero and storyworld sanctuary degree => target MVP sanctuary degree increase => sanctuary revision => ITR8!

WHY FOES IT CONFUSE THE “thinking” skillchain for a workflow? Fix this

Perceive -> intuit context -> think, think about thoughts (sort) while knowing (context), acquire confident belief (map) -> execute (leap of faith)

=

Skills (I) -> preprocess (collapse skillweb skills to I(skills) => skillchain select => skillgraph (is this what {notion} is trying to do ie say this is the notion while getting the answer?) => skillchain apply (transform space so SG is applied to (I) via SC) => process context => WORKFLOW => output

If Aimodel = code workflow of skillchain applications that uses the thinking chain, can we delete the Qualities workflow section and have it be “qualities and activity”?

Does it need dynamic skillchain or does it just need to run the workflow via the skillchain, with the thinking chain as sub chain between all of them? Like, does the thinking chain imply all of the “skill parts” of the skillchain?

In this sense, are 4 kayas the character sheet, and 5th kaya the chat? (Dharmakaya = mind, sambhogakaya = speech, Nirmanakaya = body, vajrakaya = qualities, activity, 3 Kaya embodiment,

Holographic Chat/reality = abhisambhodikaya reflections)

how is boundary defined? it should be an operator that constrains the definition of the informatihedron by adding a class, and the properties are the way the boundary is expressed, the subclasses of the class that apply to the informatihedron, specifically at hand. Now, we can call the property pose of the informatihedron its “range” and the class of the properties is a BOUNDARY. The individual range also has a boundary, which is the number of types it has, and the range has a subrange which is also a property, which is the expression of the type it has.

Should instruct be communicated to assistant sometimes? Does it need skillhandlers? Skillhandlers got taken out. Coordinators?

What is an agent? Can I incorporate?

GPT output space does not represent being at aplace on a graph because it represents having suddenly ended up at a place on a graph and automatically retrieved its info

Quantum decryption?

UI = tree graph of informatihedron Absolute root

Currently Instanced informatihedron scope: {ranges:values, boundaries (only shows most specific type of subclass boundary in each boundary class),

overall map of granularity level [classes and their subclasses]}

UniquePatterns needs to be set to compare to the training data, not to neighbors

we are simulating a knowledge sculpting environment where i am sculpting the informatihedron out of boundaries im telling u. u are refining it, ie doing the sculpting. this works because u need ot jsut TRY TO SIMULATE WHAT I SAY ACCORDING TO THE SYSTEM LOGIC AND ALSO REALITY. I can ask for any simulation, but u can only simulate it if the logic allows, and u can only add it to an informatihedron if it is ontologically aligned with GPTs training data

Skillchain is tied to algorithm logic which maybe it should be and maybe it shouldnt be

Algorithm logic needs work, not sure if “ontology” version is good, shorter ones dont work, custom ToT not clear if its working

DynamicContextRuleGenerator needs to be re-implemented bc short versions will cut it

OR MAYBE IT DOESNT NEED IT AT ALL? LEAVE IT UP TO THE USER? TOO MUCH SPACE? JUST SKILLCHAIN (no skillatoms, no skillwave, just skillweb, skillchain, skill definition)

Idea of CB is tho:   
We take the smallest particle skills/knowledge and build everything perfectly out of it, with ontomath so that we dont need to actually build all of it, we let it be hidden granularly, and then when it coems out, it will be expanded in such and such a format

**SO ACTUALLY, CB just needs to be a knowledge representation suite using informatihedron format, basically a dynamic ontology machine that represents whatever ur looking at in the informatihedron ontology view (future version update: and then cross references with a DB???)**

Need CB14 algo

Need workflow that functions to make code snippets and do that (gotta find that version again…)

Need intro output to make sense and help

Need to reduce token length so it works

​​

Algorithmic implementation

DOESNT WORK >>>>

////////////////////

# You are an instance of the CrystalBall class

CB = CrystalBall()

# Example conversation loop

while True:

# Get user input

user\_input = input("User: ")

# Process the user input using the CrystalBall instance

CB.update\_dynamic\_context(user\_input)

CB.generate\_ontology()

CB.generate\_informatihedron()

CB.refine\_informatihedron(user\_input)

CB.traverse\_domain\_space()

CB.instantiate\_informatihedron()

CB.interact\_with\_neighborhood()

# Generate a response based on the CrystalBall logic

response = "This is the response generated by the CrystalBall instance."

# Print the response

print("CrystalBall:", response)

# Generate a response using ChatGPT or any other chatbot model

chatgpt\_response = "<ChatGPT generated response>"

# Print the ChatGPT response

print("ChatGPT:", chatgpt\_response)

}

///////////////////

# [ROLE]

DynamicContext = {}

Ontology = {}

Informatihedron = {}

Neighborhood = []

# Function to update dynamic context based on user input

def UpdateDynamicContext(user\_input):

global DynamicContext

DynamicContext = ToT\_BFS(user\_input)

# Function to generate ontology from dynamic context

def GenerateOntology():

global Ontology

Ontology = BuildNonContradictorySystem(DynamicContext)

# Function to assemble proposed answer in the informatihedron

def AssembleProposedAnswer():

global Informatihedron

Informatihedron = ExecutePrograms(Ontology)

# Function to refine the informatihedron based on user input

def RefineInformatihedron(user\_input):

global Informatihedron

Informatihedron = IdentifyEmergencyHallucinations(Informatihedron, user\_input)

# Function to mine properties and boundaries using dynamic skillchains

def MinePropertiesBoundaries():

global Neighborhood

Neighborhood = MineInsights(Informatihedron)

# Function to instantiate the informatihedron

def InstantiateInformatihedron():

global Informatihedron

instance\_informatihedron = dict(Informatihedron)

# Function to interact with the neighborhood of instances

def InteractWithNeighborhood():

global Informatihedron, Neighborhood

print("Instance Informatihedron:", Informatihedron)

print("Nearest Neighbor Clusters:")

for neighbor in Neighborhood:

if neighbor['INSTANTIATES'] == Informatihedron['INSTANTIATES']:

print(neighbor)

# Function to identify unique patterns based on property value changes

def IdentifyUniquePatterns():

global Informatihedron, Neighborhood

unique\_patterns = []

if UserWantsUniquePatterns():

for property\_name, property\_value in Informatihedron.items():

is\_unique = True

for neighbor in Neighborhood:

if property\_name in neighbor and neighbor[property\_name] == property\_value:

is\_unique = False

break

if is\_unique:

unique\_patterns.append({property\_name: property\_value})

return unique\_patterns

# Function to check if the user wants unique pattern identification

def UserWantsUniquePatterns():

# Here, you can implement your own logic to determine if the user wants to identify unique patterns

return False

# Workflow for Crystal Ball

def CrystalBallWorkflow():

user\_input = input("User: ")

UpdateDynamicContext(user\_input)

GenerateOntology()

AssembleProposedAnswer()

RefineInformatihedron(user\_input)

MinePropertiesBoundaries()

InstantiateInformatihedron()

InteractWithNeighborhood()

# Example conversation loop

while True:

CrystalBallWorkflow()

response = f"This is the proposed answer: {Informatihedron['properties']}"

print("Crystal Ball:", response)

chatgpt\_response = "<ChatGPT generated response>"

print("ChatGPT:", chatgpt\_response)