mezzònomy white paper

April 2014

Table of Contents

1 Introduction

This white paper to explain the basic principles of the Blockchain Hypertext Machine and mezzònomy key contribution to its ecosystem management : the **Blockchain Hypertext Pilot**.

This document is public and thus does not divulge any information critical to the construction of the Machine and its Pilot, and more specifically, absolutely no information on its algebraic foundations, without which its building is not profitable.

All the difficulty of this document is embedded in the previous statement. Without the key, you can't open the door. To get the key, you have to reach the desired level of confidence to ask for an *Non-Divulgation Agreement* by looking through the hole.

1.1 Context

mezzònomy develops since its foundation in 2008 the concept of an *Ideal Hypertext Machine* able to provide sound and solid outputs to any purpose by mean of an incremental story construction. This concept was remarkably close to the concept of Blockchain machine, a federation of "*same*" instances of the "*same*" machine performing the "*same*" computations at a global scale to provide the "*same*" representation of the past to its users.

Blockchain came with one issue, how to manage the ecosystem it creates. The first ever blockchain, bitcoin, and many of its copies, even when they carry meaningful contributions to the object, like Ethereum or Hyperledger, chooses to manage the skin resistance against aggression of the ecosystem through a monetary approach. This is clever, but has drawbacks – and specially one, its energy costs, more than a Watt per Bit in the case of bitcoin.

Our proposition is different, we propose to delegate the the skin protection of the machine to a federation of subjects called the engineers. Assuming they manage this ecosystem through peer-to-peer cooperation and giving them all the necessary tool to contract and monetize their cooperation. Their contracts can be set on any economic parameter of the system, those concerning machine skin thickness to match its exposure of course, but also those concerning machine spreading and its use.

All these ecosystem management tools, skin protection, controlled spreading, use monitoring and control, came bunched in a single "*horizontal*" application called the "*Pilot*" hosted by the machine and accessible through plain and secure HTML5 interface.

Components of this application can be recycled in numerous other fields of activity, like supply-chain of commodities, drugs, industrial artefacts (OEM). These are expertise and market "*verticals*" of partners or affiliate of mezzònomy.

mezzònomy has but one vertical. offering to the market an alternate blockchain ecosystem management solution based on blockchain values : cooperation, negotiation, and monetization.

1.2 Object

This document state the principles and the investments needed to provide to the market a **Blockchain Hypertext Pilot**.

1.3 Glossary

- **Engineer**: subject allowed by its peers to pilot the ecosystem infrastructure.
- **Federation**: group of subject or machine related by peer-to-peer relationships
    1. the ecosystem is a federation of machine "*piloted*" by engineers.
    2. An intention is a federation of ideas leading to a collective input recorded in a blockchain.
    3. the machine is a federation of instances hosted by computers all over the world-wide managing an hypertext representation of the common intention of its users.
    4. the engineers is a federation of subjects in charge of the machine piloting
- **Subject**: any actor interacting over a machine.
- **Idea**: Any subject thought leading to an input.
- **Input**: elementary subject to machine interaction, interpreted as a learning step by the machine of the motive, the intention leading to it.
- **Machine**:
    1. any mechanical device.
    2. Without further notice, its a "*Blockchain Hypertext Machine*"
- **Instance**: Single virtual machine hosted on a computer mirroring hypertext applications by means of the blockchain.
- **Blockchain**:
    1. complex and multidisciplinary method and algorithm federation allowing to build eternal machines with solid past.
    2. Sequence of solid records – a difficulty with blockchain experience is that the method and the object share the same name.
- **Solid**: a solid record is a record which is the
    - same for all instances of the machine,
    - sealed in a chain,
    - sound, that is hyperlinked to multiple previous records
- **Application**: an document stating all economic, behavioural and technical parameter of a particular use of a blockchain. Application are developed in XSL.
- **XSL**: descriptive programming language, as any resources accessible to engineers, this a global standard issued by W3C
- **W3C**: The World Wide Web Consortium is an international community that develops open standards to ensure the long-term growth of the Web. To our eyes, the prototype of the engineer community.
- **Pilot**: application allowing engineers to pilot machines. mezzònomy product targeted by the fund raising in progress.

2 Partial learning sharing of a software application

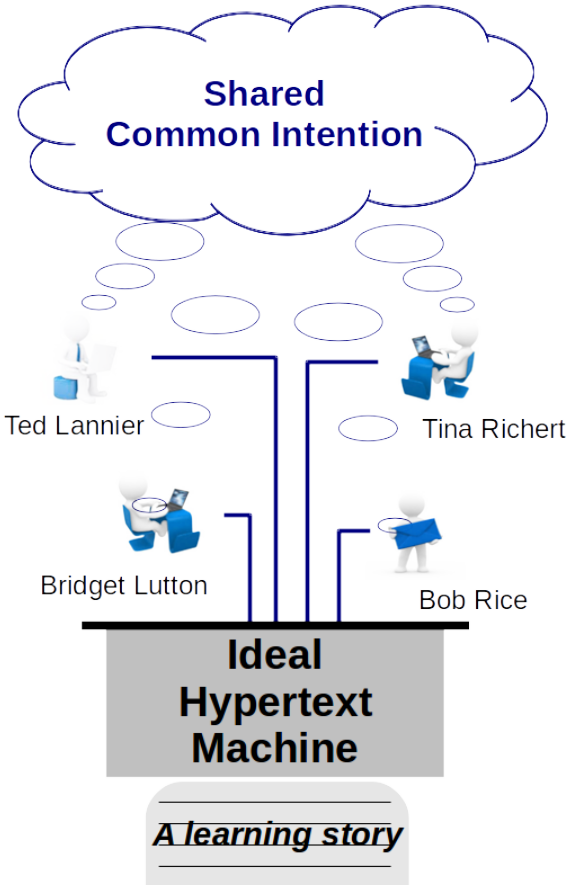
The complete title of mezzònomy patent referenced as EP2011/071770 and US 20130304678 is "*Method for partial learning sharing of a software application*". The patent being public since April 4th 2017[1], this part is a public statement explaining its principle and some of its use or implications.

2.1 Ideal Hypertext Machines

mezzònomy patent is expressed as a new communication pattern between "*machines*" but these machines are not common computers, these are "*ideal hypertext machines*" (IHM) close to the "*Xanadu servers*" envisioned by Theodore "*Ted*" Nelson in 1965.

IHM offer to each subject "*sharing a common intention*" a man-to-machine interface allowing him to learn this common intention to the machine.

$IHM_X(S)$ where S represents the "*current machine learning stage of the common intention*" is "*a composition of W3C regions*", that is, in common tongue, an HTML page.



The patent embodies "*current machine learning stage of the common intention*" as "*a story of inputs*":

- Any subject *X* sees $IHM_X(Story)$ as its view of the story
- If needed, a contributor subject *C* inputs a "*learning instruction to the machine*" to make $IHM_C(story; input_C)$ "*closer to his view*" of the common intention
- Any subject *X* sees $IHM_X\ (Story; input_C)$ as its view of the story
- If needed, a subject… *[ad libitum]*

Some subject may share the same appearance, for example in our toy example, it might be significant that $IHM_{BL}(S) = IHM_{TR}(S)$, *if Bridget Lutton* and *Tina Richert* belong indeed to the same "*community of views*".

As a contributor to the issue of mutual exclusion problems in distributed systems, the patent writer left aside, by design, the serialization of an interleaved story has an open issue.

1 This thrilling day where Sir Tim Berners-Lee was named recipient of the Association for Computing Machinery (ACM) A.M. Turing Award.

2.2 Deployment of a partial sharing

Some intentions can be hardly performed by one single machine, even an ideal one. This limitation embodies that over a certain level of complexity, the top common intention relies on subsidiary ones needed to be delegated by agreement or contract.

By complex shared intentions, through mezzònomy experience of it, our its founder's one, may lead to the following list :

- launching and maintaining a satellite constellation [THALES ALENIA SPACE | 1997 -2000],
- throw relativistic protons in a human body to cure cancer [IBA | 2001-2002]
- building a plane from sketches to the sky [AIRBUS | 2005-2010],
- delivering parcels or commodities world-wide with numerous actors [~~NDA~~ | 2011-...],
- insuring payment of mutual insurance in a national health service [~~NDA~~ | 2016-...],

All these complex shared intentions need a federation of IHM.

In this self-contained world depicted in the patent, communities have but one capital: *machine learning of their intention*. To preserve their economic situation and the integrity of their capital, federated subject can't allow uncontrolled access to their IHM.

The question addressed in the patent is how a pair of IHM can communicate and exchange value to perform complex shared intentions, without divulging their story.

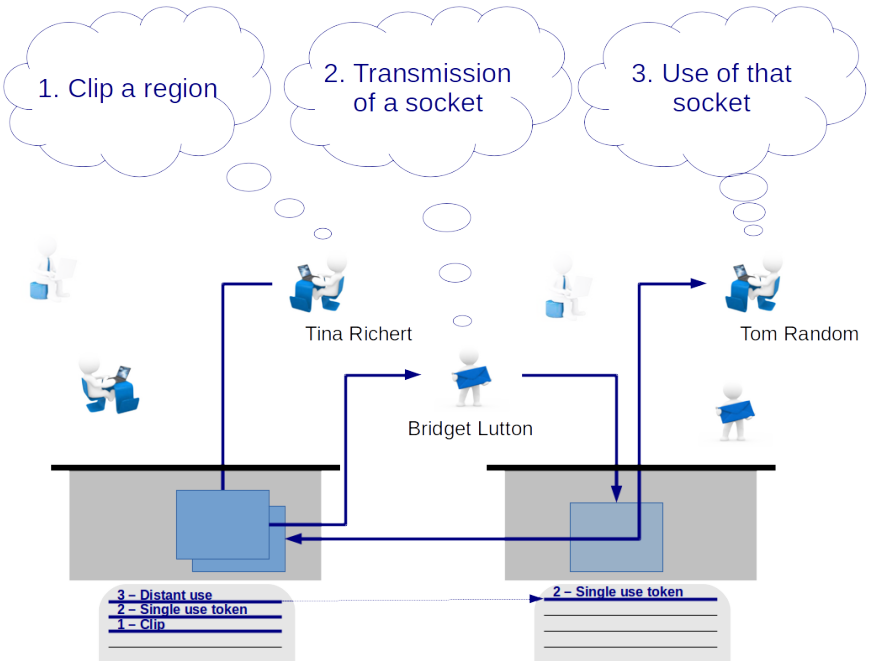


Figure 2: *Partial sharing deployment by clip socket transmission*

Assume that the shared intention around "*primary machine*" is to provide a quotation application for different kind of retailers[2]. Around "*secondary machine*", shared intention is to retail this kind of items. "*secondary subjects*" do not need to access "*primary machine*" only the application they need, and "*primary subjects*" do not want to see their client to intrude in their process or pillage their knowledge.

The method depicted in the patent is very intuitive, one actor defines a region on its view of the story of the "*primary machine*" and "*clips*" it. This operation has defined a "*primary region*" which does not concerns all the view of the story, but only the "*clipped part*". Once the perimeter is clipped, it acts as a "*partial sharing*" server.

Partial sharing server is not public: any actor of the primary machine deliver only single-use token to access it. That could be done on-demand in exchange of a of monetary transaction or tightly controlled, unit per unit, if needed.

Figure 2 expresses that controlled deployment process. Note that "*secondary users*" can clip and view their socket to address different quotation processes, but all the computing of input internal to any image of the "*primary clip*" will be done inside the primary machine.

The "*primary story*" is kept safe, only its resulting behaviour is transmitted.

2.3 Drag'n Drop composition

In fact, partial learning sharing by means of clip socket exchange is a graphical equivalent to the encapsulation of a function in a standard programming language.

Another question addressed in the patent is what kind of programming can be compliant with that form of programming. That's the purpose of the "*secondary process*"[3].

The hint is that "*drag'n drop*" works precisely as "*single use token*".

Therefore any IHM story composed of "*drag'n drop composition*" is compliant with partial sharing of the learning of a common intention. This is still a very powerful statement to allow any users to contribute to a programming environment.

We develop in 2011 a complete prototype of a "*drag'n drop composer*" called MIRZA, with the capacity to develop a simplified quotation application, fluid and intuitive.

At this stage, we had a powerful and expressive way of programming, with the side capacity to be decidable by construction – as any spreadsheet machine incidentally, and with an embedded solidity of intellectual property...

But to what intend?

2 in the "*secondary process*" of the patent, it is a "*bicycle retailer*" inside a supply chain involving a "*bicycle provider*" and a "*machine-welded frame*" industrial.

3 despite its obfuscated title due to the comparison with standard spreadsheet office applications.

3 Blockchain in a nutshell

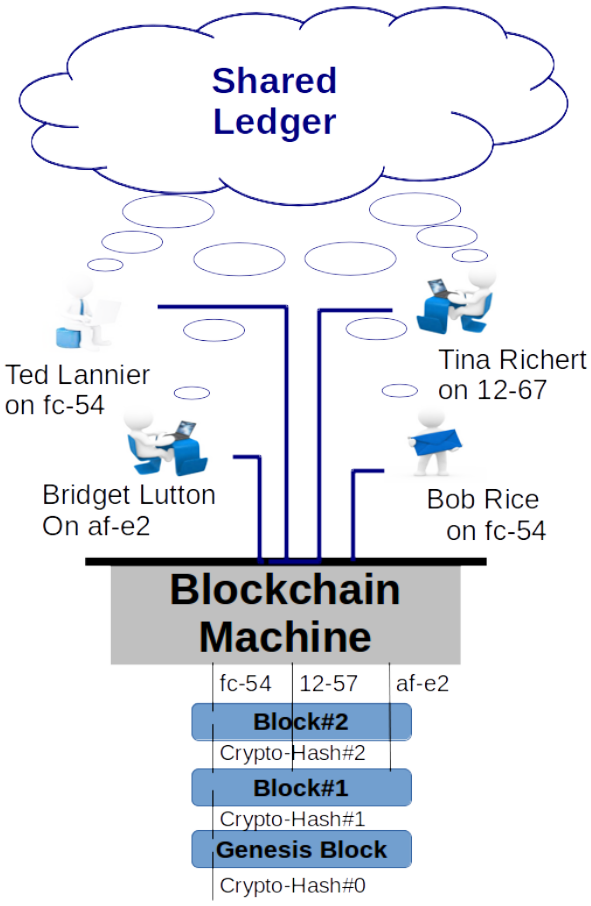
Blockchain, as bitcoin "*distributed infrastructure*", appears on our radar by mid-2015, while we were developing, after numerous development of the hypertext engine around digital factory, another composer dedicated to office document and W3C screening, but still with lack of a clear market to match with.

Bitcoin and its underlying blockchain technology erupts and, in a dramatic resonance with our own views, promote solid stories as the foundation of the ideal global computer to come:

- The story recorded by bitcoin blockchain was the first ever story recorded by mankind: *accountant ledgers*. Our primary assumption is there was room for other story categories known since.
- The solidity of the story is achieved by means of "*cryptographic quality hashing*", a deterministic computation issuing a long random sequence of bits which avoid the past to be changed.

3.1 Distributed infrastructure

The general scheme of a "*blockchain machine*" can be pictured on the right as a possible implementation for our *Ideal Hypertext Machine*.



*Example: four subjects share a distributed blockchain machine composed of three "instances" hosted on standard computer named **-**. Inputs of subjects linked to fc-54 were just sealed in the Block#2, and knowledge of this block has not been "verified" by the two other machines.*

No global authority defines the correctness of the chain, every instance of the machine does the "*same*" verification on the "*same*" data, at the "*same*" time, to reach the "*same*" conclusion (Cf. Blockchain Application). This heavy redundancy allows the blockchain machine to manage the loss and recovery of instances and as long as the machine exists, its story won't change, the only possible operation is to append new inputs.

The "*same*" can be breached in two circumstances : interleaving and conspiracy.

Interleaving came when two different instances of the same machine propose different blocks at the "*same*" time. Two "*forked*" versions of the same story may coexist in the network, one will be "*elected*", the other "*deprecated*". In bitcoin blockchain, the election is done by length : the longest is elected. This leads to the majority issue for executive control of the blockchain and the question of conspiracy.

3.2 Crypto-currency

Bitcoin came with more than blockchain. Bitcoin came with an economic model, relying on a disruptive conception of cooperation, value and money. In bitcoin blockchain, the assumption of common intention is not made and will see how this lead to its economical model. As there is no way to control the participants, the bitcoin blockchain can be attacked by conspiracies of servers.

We already saw that when alternate stories are promoted by different machines, the chosen story is the longest. Therefore in bitcoin blockchain ecosystem, conspirators can submerge the blockchain with an alternate story, a sequence of alternate and potentially fake inputs – in a blockchain translation of the classical "*Denial of Service Attack*" (DoS).

To embodies the risk of these attacks, bitcoin put it into an economic equation based on energy, monetised in the blockchain with the bitcoin currency. To do so, each block is associated to an energy cost called Proof of Work (PoW), a huge computing called "*mining*" is needed to add a block which requires an "*instant computing*" to be checked.

To mine is to finds a solution of a random puzzle of a given difficulty. Setting the "*mining difficulty*" is the key economic parameter of the bitcoin infrastructure : too little makes it vulnerable to a DoS attack, too big makes it unable to be used[4]. In bitcoin, mining difficulty is updated every 2016 blocks, based on the time it took to compute the previous sequence of 2016 blocks in order to set its desired duration to 14 days[5]. Again, no global authority sets the difficulty, every instance of the machine does the "*same*" computing at the "*same*" time on the "*same*" data, and reach the "*same*" conclusion.

The *instant computing*" check, is part of the computing done by any machine instance, but the mining can be done by any machine. The instance issued a puzzle, and delegates its solution to subsidiary grids called "*mining pools*". When companies offers you to "*mine bitcoins*", they actually manage for you a place in a mining pool. If mining is part of the bitcoin infrastructure, it is not a part of bitcoin machine instance.

In bitcoin infrastructure, bitcoin instances spend a small amount of time and energy to perform the puzzle checking all around the world, and mining pools consume huge amount of energy to feed them with on-demand puzzle solutions. Today, bitcoin distributed infrastructure consumes more

4 the maximum security level possible, 2^{224} operations, would require the whole energy of the sun for an hour to achieve a single block. Yet, it is far less than the effort needed to insert a fake message in the chain, 2^{256}.

5 This intends a constant targeted block rate around one per ten minutes.

than 250MW to maintain a less than 250MB ledger and 99% of this cost is mining. Thus, mining is the price to overcome (so far) conspiracy, what we call in the latter "*skin protection*".

But, mining has also a positive side effect on interleaving. No one can predict when someone will find the solution to the puzzle. Setting the average global time to 10 minutes makes very thin the probability (1/10000) that two miners found a solution to the "*same*" problem in the "*same internet time slice*". Thus, mining is also mitigate interleaving for the same price.

In bitcoin, mining is the price for sealing, for solidity and above all the price for "same".

That price is converted into currency by rewarding the miners : every block rewards its miner in bitcoin. Every 210.000 blocks, the reward is cut by half[6]. So, if bitcoin economic model sets the value in its infrastructure, its monetary representation called bitcoin is stored in the bitcoin blockchain.

3.3 Blockchain applications

In order to allow every instance to perform the "*same*" computing all around the world, bitcoin came with a "*virtual machine*" (VM) called Stack[7].

All computing concerned by the requirement of "*same*" are coded in Stack, mainly the "*next difficulty computing*" essential to the economic model of the infrastructure, and the "*verification computing*", essential to its intention.

The verification starts when a machine receives a block – *as an example, when 12-57 receives Block#2*. This verification is mainly composed of:

- **the proof check**: the "*puzzle checking*", the "*instant computing*" counter part of mining
- **for each input in the block:**
    1. the **signature** of the input – this is performed by an asymmetrical cryptographic algorithm, such as ECDSA. Subject privately signs the input, issuing a number, with this number anyone can publicly verify that the subject has signed it. Again, making a false signature requires unaffordable amounts of energy.
    2. the **right** of the signing subject to perform that input. In bitcoin application, the rule is crystal clear: "*bitcoin is not a bank*". No loans. Any account must be positive, any time.

Bitcoin has given birth many children, the most important of them is Ethereum. Ethereum is close to bitcoin but its design separates the "*ecosystem*", the "*blockchain*" and the "*application*" and that is a very important innovation.

This innovation has led the emergence of an ecosystem called the "*Decentralized Autonomous Organization*". The DAO raises important fund in may 2016 to allow different applications to use

6 This implies that the 210.000 first miners had mined the same number of bitcoin that all other miners in entire history.

7 Here is, at last, one main component of bitcoin which is not called bitcoin!

the Ethereum blockchain. But the ecosystem collapse in June 19th 2016. The cause of this collapse is due to a design flaw here discussed.

We saw that the kernel of the application is a virtual machine providing the "*same*" computational environment to the "*same*" distributed code. The flaw is in the language of that code. It exists three majors levels of programming complexity[8] :

1. **Lite programming**: the level used by "*Stack*", the language of **Bitcoin** – the name being a reference to the kind of machine needed to interpret it "*deterministic single stack automata*"

   ~~X **poor expressiveness**~~: you can evaluate formulas without any recursion. Sufficient to compute a compensation, but not to express a contract or a process.

   √ **complete decidability**: a *lite program* will always terminate.

2. **Descriptive programming**: the level used by "*Velocity*", the language of **Hyperledger** machine instances. The name expresses the incredible speed of this kind of machines called "*tree automata*". Without string* functions, XSL belongs to that category and thus the **mezzònomy** ecosystem.

   √ **strong expressiveness**: you can express almost everything you need for a blockchain application, but sometimes, you will have to "*engineer*" it right.

   √ **strong decidability**: a *descriptive program* will always terminate, but sometimes it will issue a "*loop error*" but before its first iteration.

3. **Full programming**: the level used by "*Javascript*", the language of **Ethereum** machine instances.

   √ **complete expressiveness**: you can express any computing, any routine you can imagine or perform is a program. Your language is [Alan] "*Turing-complete*". The original Turing machine was a "*deterministic double stack automata*".

   X ~~**poor decidability**~~: a full program may *never* terminate, and you have no way to prevent it.

And here is the Ethereum design flaw, a hacker succeeds in inserting a transaction in an "*infinite loop*". And on June 19th, Bingo! Of course, the machine manages to crash on a kind of "*loop error*" but after millions and millions of iterations: a quarter of the raised funds goes in hacker's virtual pocket in the blink of an eye, up to a quarter after the crowd-funding[9].

Blockchain ecosystem management is still an open issue.

8 We here use the taxonomy of the Ontology Web Language (OWL)

9 This dramatic fail had an important impact on our company. We were in the final stage to be chosen as provider of a proof of concept blockchain application and this event damaged the reputation of our single concurrent.

4 Manage Blockchain Hypertext Federations

mezzònomy was founded in 2008 by Pierre Gradit, PhD in Computer Science, granted by the submission of a thesis in 2000 over the *modelling of cooperative applications*. The conclusion of the thesis that there was, at that time, no proper model to describe a "*dynamic system of intricate knowledge owned by the people*", later called "*hypertext*":

- Some models offer simplicity, like *Petri nets* and the capacity to *model processes*. Some models offer *expressiveness*, like *term rewriting*, but then they can not *model networks*, as we state in part 1.
- Some models offer n*etwork modelling*, and specially loss/recover of instances but not *expressiveness*, even with the algebraic graph grammars context we elaborate in part 2.

In part 3, we state that Petri Nets could be seen as the common ancestor of both *term rewriting* and *graph grammars*, but we lack the *common descendant*. Part 4 was dedicated to the most complex example we can deal with term rewriting: *the article acceptance and the proceedings editing of a conference*. A process we had tested a lot during this work but with little success.

And in 2001, after a decisive journey to Japan and ten years spent to the service of France as any scholar of ENS Lyon, the founder left the public research and the academic world to work in the field.

4.1 Precursor "*Pilot*"

In 2001, the first case study to implement his vision was to re-engineer a complex federation of embedded systems built to monitor an control a gigantic ion-beam device located at Boston Hospital for proton therapy. The mission was to retail part of it to another identified client.

It rapidly appears that we need a platform to deal with all these intricate data, inputs and knowledges. At that stage, we chooses XML/XSLT to make it – a "*descriptive programming*" environment based on a world-wide standard suits our needs and intentions [10].

With this precursor "*pilot*", we implement a proof of the concept hidden in the design of the federation maintenance process: a genuine precursor of "*digital twins*" used there for testing purposes. Each system of the federation, and even recursively sub-systems when needed, were able to be isolated and fed with data captured in the field. Specially when problem occurs. There were seldom simple issues in a federation of fifty embedded softwares controlling an entire building dedicated to send relativist protons in a human body.

Reshaped for a simplified test case, the mission target, as "*integration test of a gas turbine embedded controller on an office PC*"[PTIV – SAFRAN | 2001-2010], the entire prototype, its complete documentation and all the needed test capacities were managed by a structured set of XML files and many XSL to manage the different outputs of it – source code, office documentation. As already

10 The argument given in the previous part about *strong expressiveness and decidability* was already in the balance.

noticed, *descriptive programming* is incredibly efficient and it appears easy to learn, specially for engineers of a foreign domain, such as automatics, mechanics, and even, *managers*.

And thus the "*pilot*" spreads. At the end of its life, the "*pilot*" was able to manage the entire process of developing the retro-propulsion flaps of the GE engine for the A380. Each iteration of the A380 project process need a complete code and documentation output. This was produces by the pilot in a blink of code generation, 24 hours of automated test on boards, 1 hour of document generation, and 24 hours for printing the seven linear meter of the complete documentation of the iteration. Five person, instead of more than 50, could manage it all. All cross-checked and sound.

Based on this success, the founder was called to join a team dedicated to specify, design, deploy and maintain a transnational and trans-disciplinary environment called ISAMI to size and qualify the structure of last-to-date civilian plane of AIRBUS: the A350XWB. But once the early stages passed, the design inputs of the founder were dismissed and his mission redefined to open most of the project technical jobs and then learn the incumbents how to manage it – through improvement of their personally skills by formation and by elaborating their collective methods with the help of a colleague. Buried in this project too big to fail, even fed by experience through the entire process of software editing and deployment, the "*pilot*" story was about to be ended. When, "*ta–da"!*

The common descendant of *graph grammars* and *term rewriting* was found, glued together by a glimpse of A. Grothendieck's work known as "*dessin d'enfant*". Originally called "*modal formulation*" to encompass its capacity to deliver multiple interactive views to a set of subjects, we call it now "*hypertext algebra*" which carry more technical meaning of its nature.

mezzònomy was founded in 2008 to develop the applications of that discovery under the social object of "*programmable software for engineers*". The first round of study, funded by love money in 2010, produces the results depicted in part 1. Since 2013, together with numerous prototypes in the digital factory trend already mentioned, we focused on the monetary implications of unique inputs around a "*Bank of Common*" project.

And blockchain strikes.

4.2 The Blockchain Pilot

When you are, as mezzònomy now is, a company dedicated to blockchain ecosystems, you have three main activities:

1. **Explain blockchain concepts**
2. **Prototype blockchain applications**
3. **Pilot their deployment and effective use**

Can we explain blockchain concepts? *We hope we did*, but this is no more a market. It might be a secondary income source, but not until the *real* market is found.

Can prototype blockchain applications? *Yes we can*. Our blockchain hypertext machine, even in a preliminary version, is already able to deliver all kind of ledgers in a more affordable and accessible way than its competitors. Forming people to develop XSL is not an issue, we made it in numerous occasions. The foundation set on *descriptive programming* prevents us from the DAO fail. But making *PoC*, and we did it for almost two decades, is not a market. *It is a hobby.*

Can we provide a generic BH application to develop BH Applications? *Yes we can*. The patent is around that issue, but there is a complexity gap. *Drag'n Drop programming* is able to manage *lite programming* issues, needed to build smart-contracts, but not the *descriptive programming* needed to build an application from the ground up. We prototyped in 2015 a Proof of Concept that states that any *descriptive programming* is in the grasp of our machine. But an hypertext application came with numerous file types, a complex structure. Developing it through standard git, with all the management tools provided by GitHub cloud solution for 7$ a month is pretty sound and effective. Struggling on that field is not in the grasp of our machine at this stage. *It might be, but not now.*

Can we provide a generic BH application to develop any kind of documents? *Yes we can*. It was the original intention of 2015 proof of concept dedicated to knowledge sharing and monetizing at research/industry interface. But a market study held in 2011 shows that market is hard to conquer, office solutions are well established. And let's face it, the ergonomic of a full-scale editing suite through blockchain hypertext can be a disturbing experience without a step-by-step learning process. And if we can state that any document produced that way is opposable as evidence in court, *who will believe it?*

There is one thing our in our grasp, congruent with the intimate ways of our machine. Something needed to achieve solid ecosystem in the field: a distributed BH application able to pilot a federation of BH applications linked by partial sharing: **Blockchain Hypertext Pilot (BHP).** This pilot provides tools from three area of engineering expertise of network building:

- **Blockchain Hypertext Pilot is a kind of supply chain manager with an embedded invoice mechanism.**

  Blockchain machine building is a serious construction game. Any brick of it, and not only transactions, but also instances, genesis blocks, partial sharing sockets, subject registration fee, can be delivered with a PoW and thus, with a prize. There is no constraint on any prize by design, transaction fee is not a necessity as the recording of all subject inputs make *sui generis* a strong protection against mechanical aggressions – "*human is the proof of work*". Once the machine code is no more proprietary[11], mezzònomy will not be the only provider of that mining effort but it can be a strong income during the "*golden time*". All the work done for the BHP can also be recycled as components for supply-chain process, both for commodities, and in OEM industrial issues.

  > *It is not mezzònomy core business to provide smart-contract applications, but it can be the work of partners or affiliates, and mezzònomy can partially share its knowledge to ease it.*

11 which is inevitable, either voluntary as a global standard, or involuntary by its hacking, both being indisputable marks of our success.

- **Blockchain Hypertext Pilot is a recursive and property preserving test manager.**

  All the work done for the precursor pilot can also be recycled as components for the pilot. A blockchain is a story, like a test scenario. Filtering, truncating, noising test cases to provide useful test case preserving the final user knowledge are long-practised techniques to mezzònomy. The last contribution of mezzònomy to the ISAMI project was an automated test suite for the entire platform. AUTOTEST was able, like in Boston Hospital, to isolate a faulty component in that intricate structure with proper inputs for correction. This kind of tools are decisive when a multi-hop application failed somewhere in the wilderness.

  All these techniques can be recycled with applications, for partners or affiliates, in the embedded systems industry, toward IoT solid applications.

- **Blockchain Hypertext Pilot build and deploy interface with existing systems**

  Based on the same design than the HTML interface for subjects operational since mid-2015, based on a PyQt precursor operational since 2012, the system comes with an integrated suite a tool dedicated to the integration of external sources of data, called oracles in the blockchain world. Based on a heuristic history reconstruction, these oracles allows blockchain hypertext machine to deal with with non-blockchain environment with a pretty sound liability.

  All these techniques can be recycled by partners or affiliates, for any blockchains.

4.3 What is to be done?

The evaluation made to reach that level of complexity through hypertext machines has not changed much since 2011: *800k€ of investments in R&D*. This sum remains constant trough time as we manage to develop or prototype part of it, but also discover new challenges to face. The new fact since 2017, is that we know what to offer to the market with this machine: *a tool to control the skin protection, the spreading and the use of the machine in a collaborative and incremental way.*

This "*software*" – which exact nature is a bit more complex than the standard meaning of this word, needs the following components to be fully operational at a global scale:

- **Hypertext XSLT engine**: this C++ library if the "*heart*" of the machine.

  Actual XSLT engines does not work on hypertext but on a precursor called DOM, a "*digital twin*" of an XML file. Even if a blockchain is still serialized in XML for fast recovery and testing purposes, the internal data structure have to be the hypertext. The loop pre-emptive alert, essential to the purpose, requires particular algebraic attention in this situation.

  Furthermore, to allow partial sharing of XSL, a bytecode obfuscated form of the *tree automaton* is needed. This form does not prevent from copy, but forbid the hacker to make any improvement on it. When the next version of an application is loaded, its copy will be useless and above all, detected as an attempt of treachery by the engineers.

And finally, we need to go further in reducing the main default of all XSLT engines: *traceability of error report*. We currently use *lxml* python binding of *libxslt* developed by Daniel Veillard, which is, along with its astonishing performances, the best on this specific issue. We need to improve it to facilitate the work of event isolation by an engineer. The isolation process need to be sure to keep the "*same*" event behaviour while rendering the test case with absolutely no glimpse of final user knowledge.

- **Subject recognition**: this C++ library is the "*skin*" of the machine. Its purpose is to compute from the subject input "*epitomics*" their private key. The standard library uses a *login/passwd classical engine*, but we plan to improve it with behavioural biometrics, and this in important design constraint for any *interactive representation*. The type of skin used may depend on the subject – or its role, or even its intention. Top protective skins may be reserved to deep applications structuring the value of the ecosystem backbone, while lighter skin allows a more viral spreading of the machine. Always remember that, together with skin protection, the spreading of the machine is a key purpose of the main application we deliver.
- **Blockchain manager**: this XSL/Python/C++ main is the glu of the machine. Its purpose is to use the blockchain content to manage the application behaviour, stating all economic parameter of any applications, as well as the network topology and the subject base. It comes with standard XSL library corresponding to the standard blockchain model, as expressed in part 2. But as this part is also "*programmable*", alternative models can be implemented, under the responsibility of their engineers. The genesis block of an application defines the core subjects, nodes and network of it, as well as cryptographic informations to secure the tunnelling of the exchanges.
- **Banking oracle**: this C++ library is the interface with the monetary world. For the hypertext machine, a fund transfer is a kind of Drag'n Drop – securing both side of the transaction offers a sufficient level of security by its redundant cross-ckecked processing.

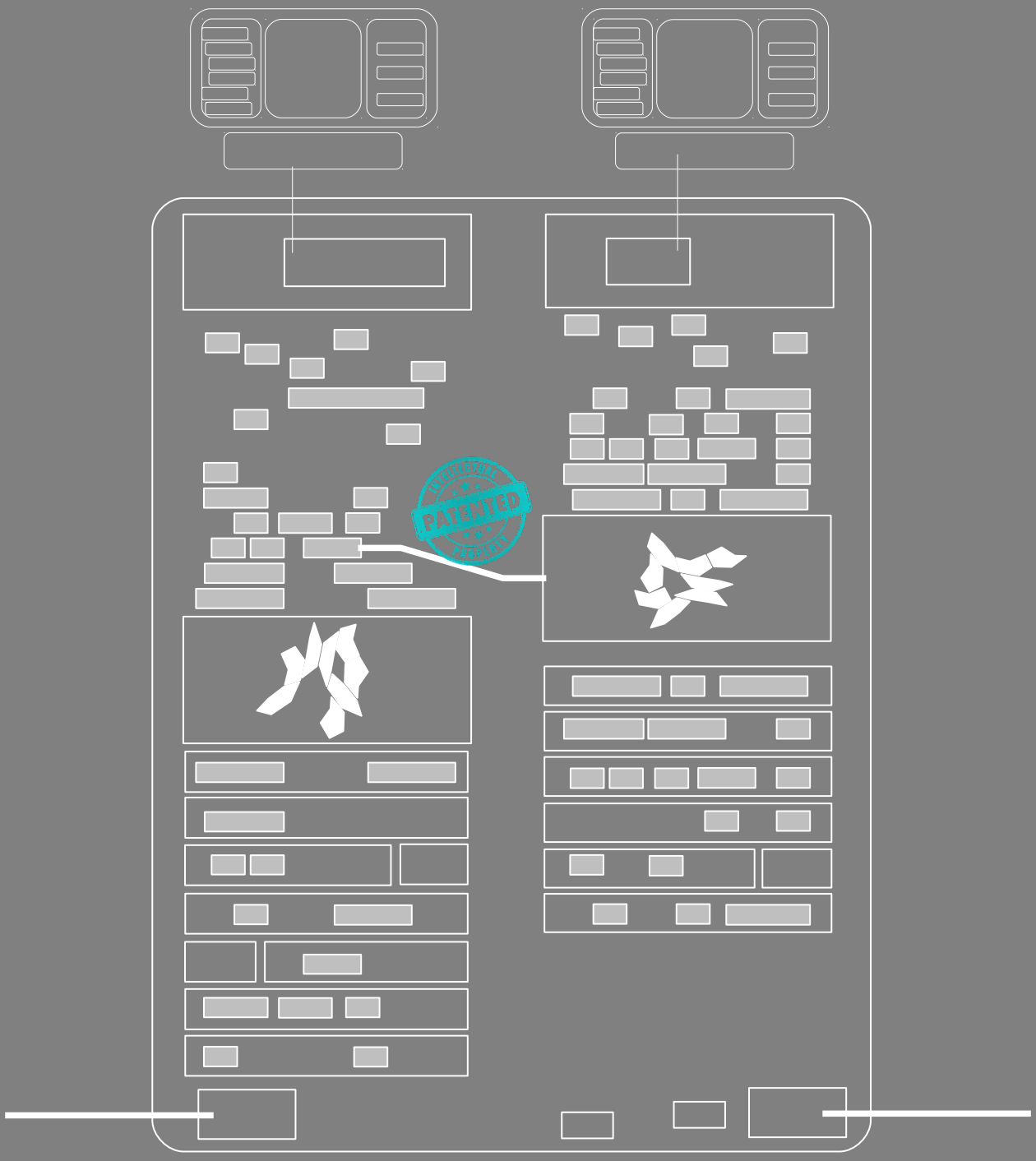
The other components are XSL-based, and serves to build the engineer interface to the pilot:

- **Hypertext Tetris**: this XSL/javascript library build a matrix *interactive representation* of the hypertext, it is needed for consensus over formulas of a given ecosystem. The SVG output, which is a kind of gimmick of all consensus phase in hypertext ecosystem may be the disturbing part advocated before. The case of formula, simpler than full text, is a necessary step for hypertext engineers. Once the use of the tetris is acquired, they will ask for its generalisation to all documents. It will start with the management of the need expression of a new application, glossaries and requirement management are intermediate objects for subjects between economic formulas and full-text. For hypertext, any of these items are formulas: some with a decidable evaluation, some without.
- **Hypertext Navigator**: this XSL/javascript library build a tree *interactive representation* of the hypertext, it is needed for structuring the environment of the engineers. This is the first

component completely describes in 2005. Its behaviour extends standard tree navigator by shading part of the hypertext already exposed elsewhere.

- **Hypertext Polygram**: this XSL/javascript component is the network *interactive representation* of the hypertext. The basic layout is the dual form of the mainstream graphical network representation[12]. Duality is a key concept of hypertext algebra. One of its application, *time/space duality*, is the corner stone of the possibility to link hypertext with blockchain and thus, the corner stone of the machine.
- **Hypertext Pilot**: the application has to be developed, but with all the components before, it is piece of cake for any engineer – somehow the examination level to become one.

12 a glimpse of a Polygram is given in the logo of the company, together with the movement of Tetris pieces fall, the general picture evoking the middle letter of the company name.


PATENTED