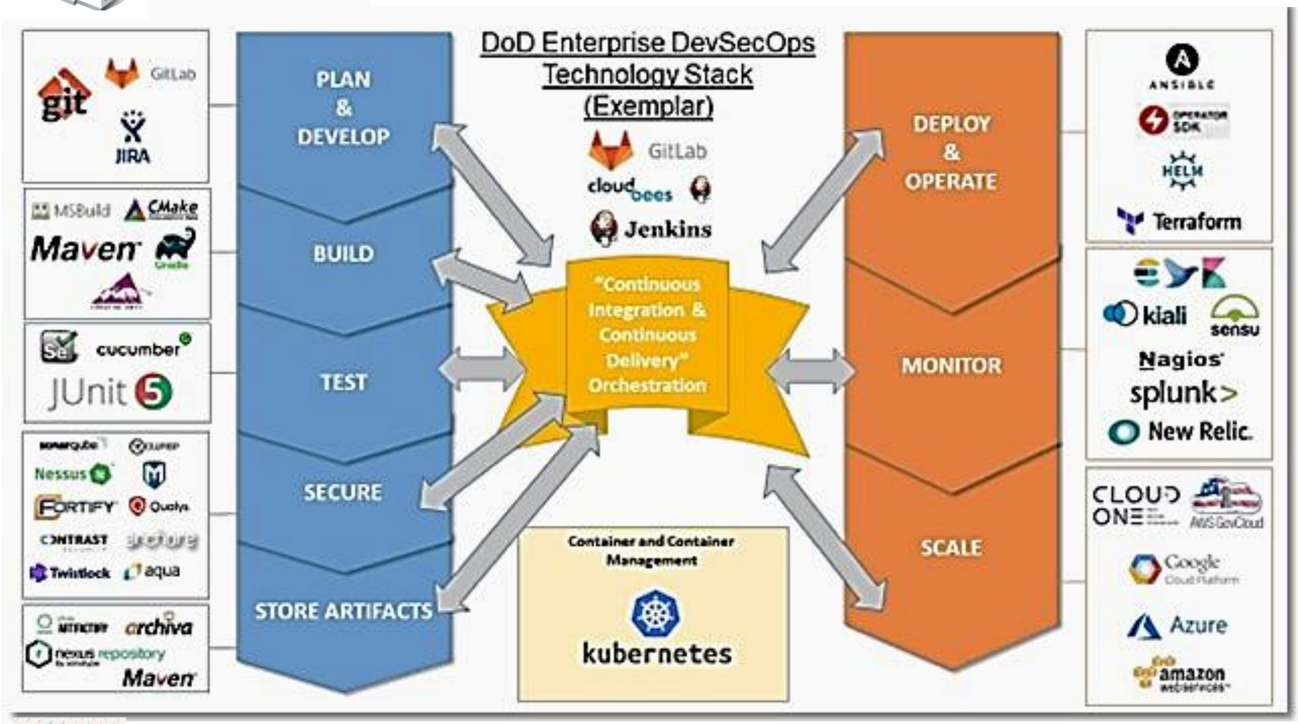
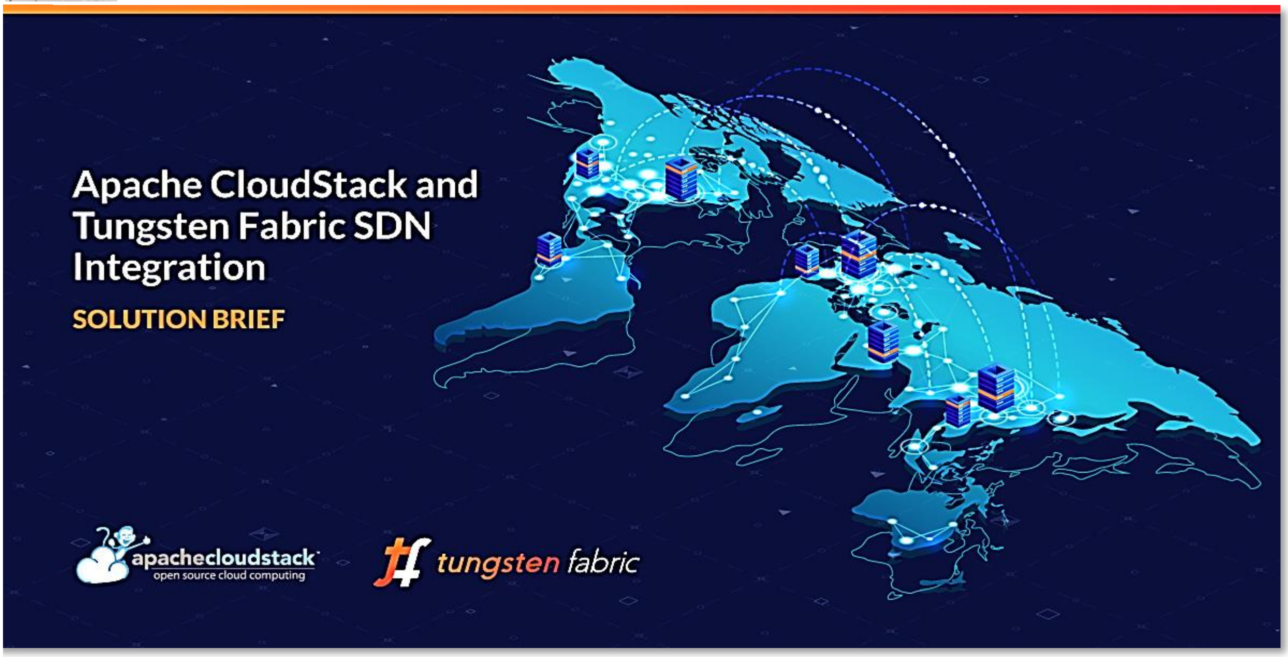
FACTORY – APPLICATION DESIGN



DATA-CENTER OPERATOR – EXECUTION ENVIRONMENT



Some Tools
in
Applications Factories
(DevSecOps)

INDEX

1.- TRANSITION TO DEVSECOPS.

1.1.- WHY DEVSECOPS?

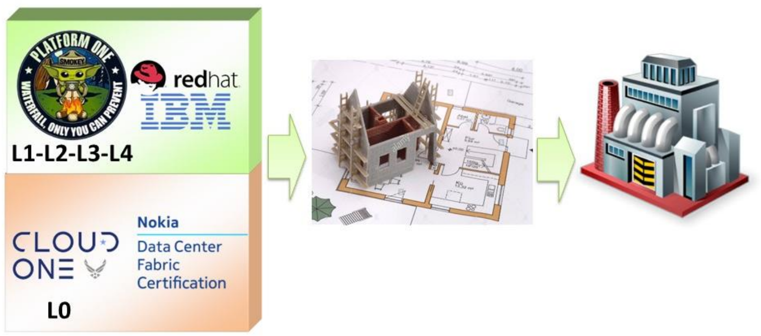
Four pillars are the backbone of motivations that led the United States Defense Air Forces to carry out a radical modernization of its entire application production system, which we are trying to identify here:

 **Cibersecurity**	Shielding the world's largest weapons system within the context of the imminent Internet of Things by adopting the following principles: a. <u>**USER** – Eliminate spoofing</u>, by using a credential system based in SIM cards instead of passwords (the very same way to be used in the imminent Internet of Things), similar to a phone line nominal assignment, evolving towards an electronic ID... with regulations in constant evolution. b. <u>**USER** – Without ease of use, security is not possible</u>: using SIM card makes many cumbersome measures used to prevent spoofing unnecessary (memorization of many and complex passwords, frequent renewal of those passwords, an associated device to authorize transactions, etc.). c. <u>**PLATAFORM** - Closed application execution environment</u>: DevSecOps platforms are much more controllable systems as they are fully managed via software. d. <u>**PLATAFORM** – Micro-segmentation</u> greatly reduces the surface area and exposure time of the data plane. Whitelist policies per service control visibility between services, minimizing the exposed data surface. In addition, each front gives access to a fragment of that data surface, and each refresh of those fronts renews authentication properties, reducing data exposure time. e. <u>**FACTORY** - Continuous analysis of applications behavior</u>: thanks to security specialists who constantly monitor and correct the behavior of applications produced by factories.
 **Cost Reduction**	<u>Software Defined Datacenter</u> (Software Defined {Network & Storage & Compute}): software-controlled data centers reduce machinery maintenance costs by up to 60%. It is also called "*Data-Center Operating System*" such OpenStack, CloudStack to handle virtual machine meshes or Fabrics to handle physical hosts meshes (OpenFabrics Alliance, Juniper Apstra, Cisco ACI, Arista CloudVision, Nokia NOS); both types of meshes sectorized in clusters (ie.: Kubernetes).
 **Continuous Delivery**	<u>Release Speed</u>: integrating and automating all factory processes provides the ability to quickly adapt to all challenges that this new information society brings.
 **Future Evolution**	a. <u>Guarantees a future evolution</u> in each component across the anatomy of the platform, being able to evolve at the pace of these cloud technologies. b. <u>Discard obsolete systems</u>, reducing maintenance costs of all the legacy that is piling up in data centers (ie: abandoning virtual machines whose management complexity means high maintenance costs to the point of preventing applications to scale; replacing them with containers).

1.2.- REQUIREMENTS: TO RENEW THE ENTIRE DATA-CENTER NETWORK.

Transition to a DevSecOps methodology mandates to renew data centers, for three main reasons:

1. **Central Control of the Network of Datacenters**[1]: increasing security criteria and adapting to the imminent internet of things... lead to centralized application distribution systems and user authentication across the enterprise cloud, just as mobile phone operators do with their network resources nationwide.
2. **Specific design of each Data-Center to get an actual '*Software Defined Datacenter*'[2] that guarantees future evolution**: the transition to software-controlled computing cannot be done without specific data center design (either virtual machine meshes over old machinery or new machinery fabric with interoperability certification[3])… eventually integrating, in the process, the mobile phone authentication systems into the infrastructure interface (as automobile industry is currently adopting[4]). Mobile phone operators expose their user base to network service providers through an interface (OSA=Open Services Access), eventually this mechanism can be used in the L0 layer of data centers, and thus maintain a single centralized SIM-based credential system for the entire computing ecosystem.
3. **Certification of each Data-Center[5] before putting into operations**: since it is a compact structure (a platform with all parts integrated into a coherent whole), the required synergies are essential to achieve an effective integration testing scaffold capable of evaluating and versioning the evolution of the platform.



Obtaining return to such investment implies diversifying the results. In other words, *certify platforms for all possible scenarios* (real time in Telco Clouds, persistence for Banking, etc.). Therefore, it is vital to standardize the interfaces of each layer of the platform in order to admit any internal implementation… *for being able to build the same architecture with different technological combinations, according to the strategy to be followed in each scenario*.

[1] **Lt. Gen. Jack Shanahan** *(director of Defense Department's Joint AI Center), "the lack of Enterprise cloud"* https://fcw.com/it-modernization/2020/05/pentagons-ai-chief-lack-of-enterprise-cloud-has-slowed-us-down/196057/

[2] **ETSI, OSM Hackfest 9,** *"OSM Architecture and Installation, the Software Defined Datacenter"***:** https://osm.etsi.org/wikipub/index.php/OSM9_Hackfest

[3] **OpenFabrics Alliance,** *"Interoperability certification"*: https://www.iol.unh.edu/testing/hpc/ofa

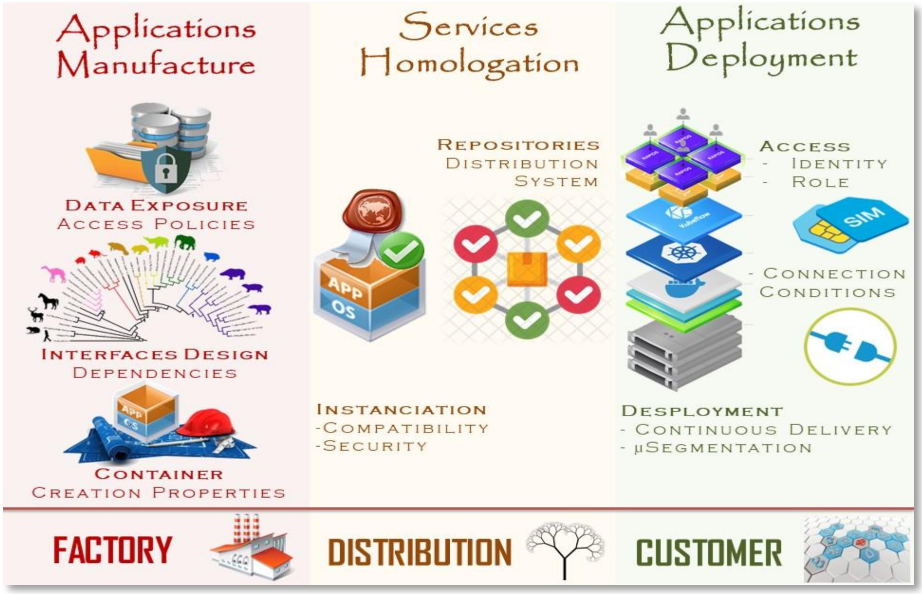
[4] **Cibersecurity,** *"iSIM, eSIM, XDR"*: https://www.nokia.com/networks/cyber-security/cybersecurity-tech-talk/

[5] **OPNFV,** *"Certification Testing for Telco Clouds"*: https://www.opnfv.org/

2.- CIBERSECURITY: CONTROLLING THE APPLICATIONS SUPPLY CHAIN.

2.1.- RESPONSIBILITIES IN THE SUPPLY CHAIN.

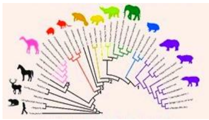
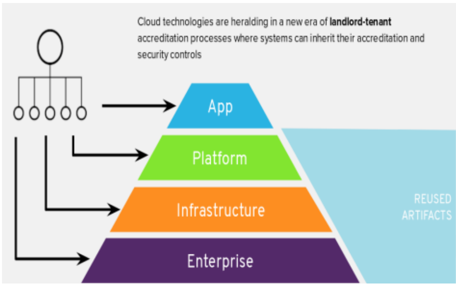
Security in Information Technologies cannot be addressed without a holistic approach that involves all agents in the application supply chain. This chapter aims to define the responsibilities of each agent in this chain. In the next chapter, some proposals for tools to meet these responsibilities:



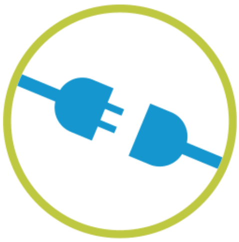
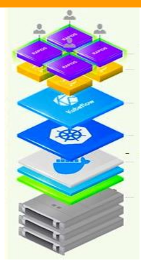
- **Access, application deployment**: network of datacenters, the services run-time platform and identity system to access the application ecosystem, involving end to end network automation with associated access policies.
- **Distribution, services homologation**: guarantee the deployment conditions of each service that, like lego pieces, are combined in the creation of final applications... being supplied and updated, continuously, through a system of repositories.
- **Production, application factories**: applications manufacture under DevSecOps methodology that guarantees standards of stability and security in end products supplied, which means:
  - *Data: Data Surface Exposure Design* – access policies to the data associated to each API call.
  - *Logic: Interfaces Design* - visualize the system of dependencies between services, to keep stable contracts of functionalities offered by each service.
  - *Communications: Microsegmentation* - whitelisting policies between services from which each application is made of.
  - *Container: Encapsulation Design* - manage the correct encapsulation of services in containers for their subsequent distribution.
  - *Artifacts Certification: Delivery* - a system of authorization gates through the supply chain to speed up the delivery time to production.

2.2.- TOOLS FOR EACH RESPONSIBILITY.

PRODUCTION – Application Factory

 DATA EXPOSURE	 INTERFACES DESIGN (DEPENDENCIES)	 CONTAINER CONSTRUCTION	 CERTIFICATION BEFORE PRODUCTION
 DATA ACCESS POLICIES (VISIBILITY FE -> BE)	SERVICES PHYLOGENETICS	NSA & CISA METHODOLOGY	 CONTINUOUS AUTHORIZATION TO OPERATE

DEPLOYMENT – Data-Center Operator

 SIM CARD	 CONNECTION CONDITIONS	 µSEGMENTATION

IP MULTIMEDIA SUBSYSTEM (AAA for SIM Cards)	 EXTENDED DETECTION AND RESPONSE (XDR) (Constant access scanning)	SECURE ACCESS SERVICE EDGE (SASE) (End-to-End Network Automation for centrally control and authorize access to each resource across the enterprise cloud)	SERVICE MESH MANIFEST (Sidecar Container, Platform monitorization)	WHITELISTING LANGUAGE PER SERVICE (Handling Exposed data Surface)

NSA &CISA Methodology, *"Kubernetes Hardening Guide"* *https://www.nsa.gov/Press-Room/News-Highlights/Article/Article/2716980/nsa-cisa-release-kubernetes-hardening-guidance/*

United States Department of Defense cATO, *"Continuous Authorization to Operate",* *https://media.defense.gov/2022/Feb/03/2002932852/-1/-1/0/CONTINUOUS-AUTHORIZATION-TO-OPERATE.PDF*

3.- OPERATOR: THE ENTERPRISE CLOUD.

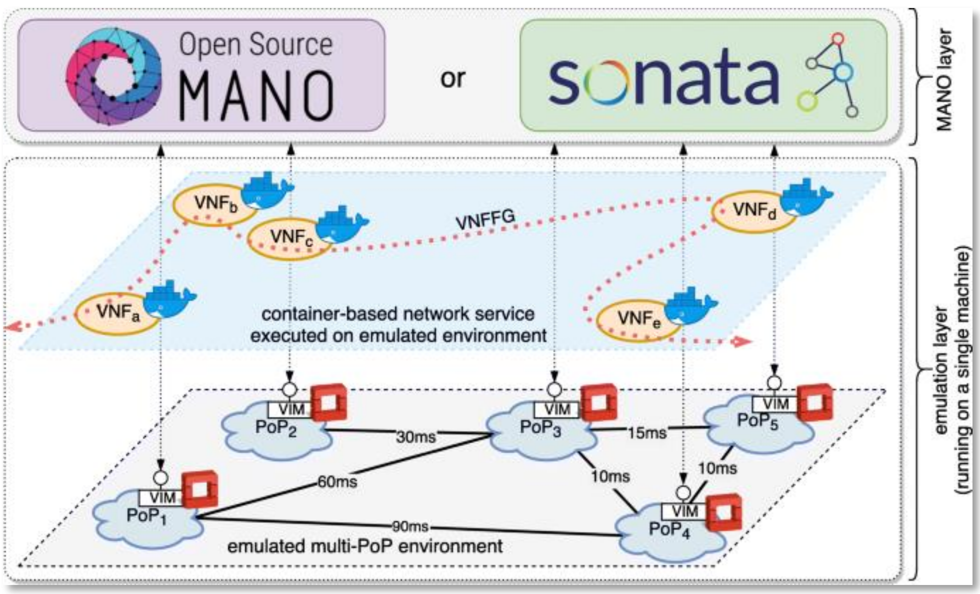
3.1.- A SECURE EXECUTION ENVIRONMENT.

As shown in last page, cybersecurity relies on two factors: applications secured by design and a secured execution environment:

- **Factory – Application Secured by Design**, four elements should be handled:
  - Design of the Logic Plane: APIs and dependencies.
  - Design f the Data Plane: access policies.
  - Design of application internal communications: whitelisting policies between services
  - Artifacts Encapsulation, instantiation conditions and delivery mechanisms.
- **Data-Center Operator – Application Execution Environment**, operators should deploy an Enterprise Cloud, meaning, the ability to centrally control all assets (logical and physical) across the network of data-centers, with associated access policies.

3.2.- ARCHITECTURE OF AN ENTERPRISE CLOUD.

In the picture how telecom operators simulate a centrally controlled network of five data-centers[6] for smoke testing of network services in a single computer. An enterprise cloud is the required organizational scheme (or architecture) to have a secure execution environment able to evolve adding new cybersecurity features, such as SIM authentication or extended detection and response as well as moving towards federations of applications that creates distributed applications to reduce data fragmentation, since content based routing required meta-data design for visibility and data growth.



The architecture of the picture is as follows:

[6] ***Enterprise Cloud Simulation :*** *https://jwcn-eurasipjournals.springeropen.com/articles/10.1186/s13638-019-1493-2*

- **Emulation Layer**: Operation infrastructure of the data center network. Two totally decoupled layers appear:
  - Layer L0: NetOps - Software Defined Data-Center... in Telco Cloud this layer is called VIM (Virtual Infrastructure Manager), each data center of the network is simulated as OpenStack inside a virtual machine. In a computing environment, each data center would consist of a network of kubernetes clusters, over a mesh of physical[7] machines (OpenFabrics Alliance, Arista CloudVision, Juniper Apstra, Cisco ACI, Nokia NOS, etc.) or virtual machines (OpenStack, CloudStack).
  - Layer L1-L2-L3-L4: GitOps – Continuous Delivery Platform... the simulation just deploys the virtualized network functions directly on Docker, without any platform involved. In a compute environment, this would consist of cluster configurations, a continuous delivery system (such as Jenkins), and a service mesh system (such as Istio)
- **MANO layer**: Articulation of the data center network. Two main elements:
  - MANO Layer: controller that distributes virtualized network functions throughout the data center network. In computing, there is no equivalent, each application factory must design a manager that allows the distribution of its applications to all the nodes of its business cloud from a single control center.
  - VIM Interface: API used by MANO to deploy applications on each data center (represented by a white dot on each VIM). In a computing environment, this is a service area controller capable of managing the network of clusters on each data center. These service areas are federated and controlled from a main header: the Universal Networking Fabric[8] (UNF).

In the picture *Nokia Nuage Networks*[9] UNF where the federation of service area controllers can be clearly appreciated as well as the main control header[10] from which access policies to each resource of the network is centrally setup.

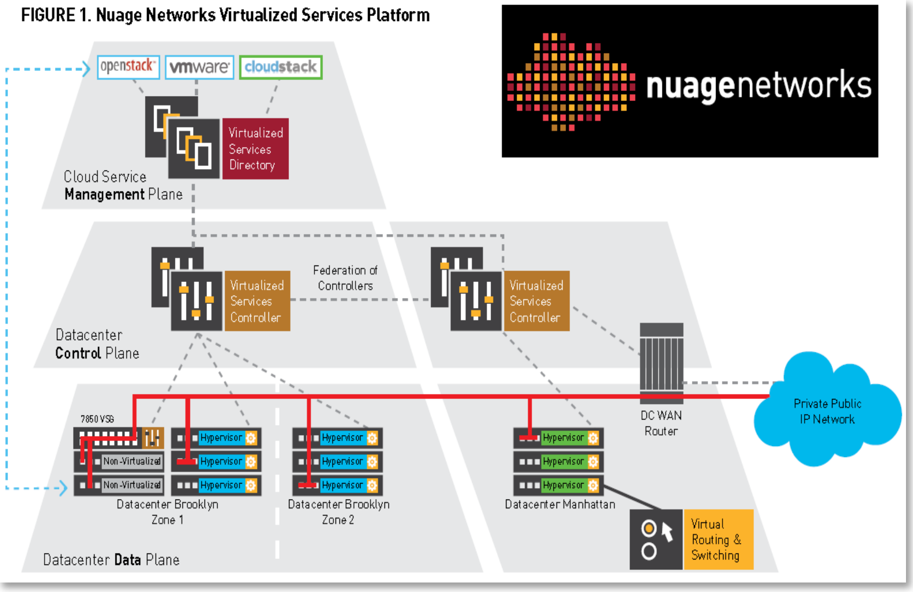
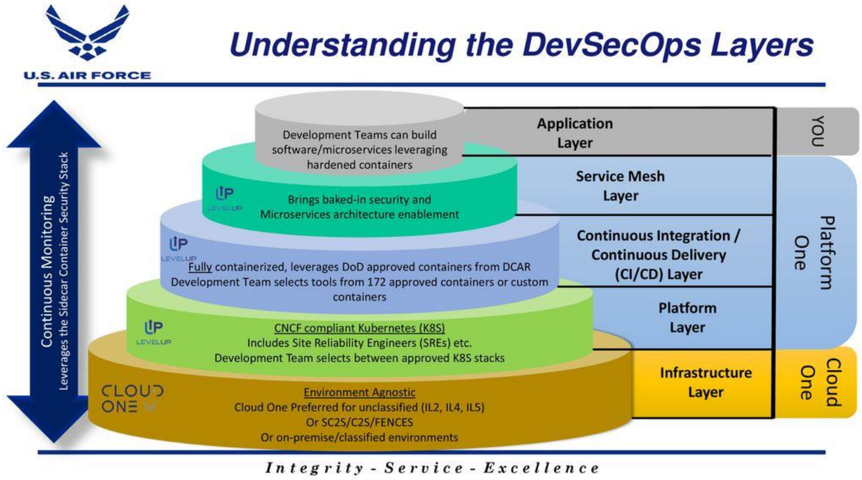


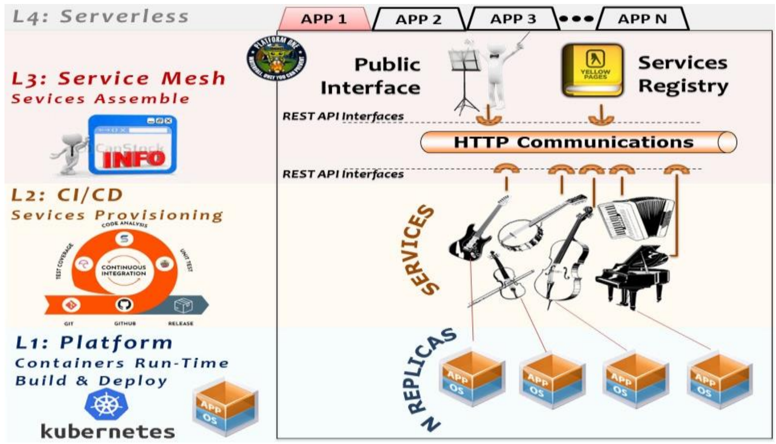
FIGURE 1. Nuage Networks Virtualized Services Platform

[7] *OpenFabrics Alliance:* https://en.wikipedia.org/wiki/OpenFabrics_Alliance
[8] *UNF, SDN Controllers:* https://en.wikipedia.org/wiki/List_of_SDN_controller_software
[9] *Nokia, The Universal Networking Fabric:* https://onestore.nokia.com/asset/212701
[10] *OVH Installs Nuage SDN for OpenStack as a Service,* https://convergedigest.com/ovh-installs-nuage-sdn-for-openstack-as/

3.3.- ARCHITECTURE OF A DATA-CENTER.

3.3.1.- OPERATION LAYERS: APPLICATION DEPLOYMENT

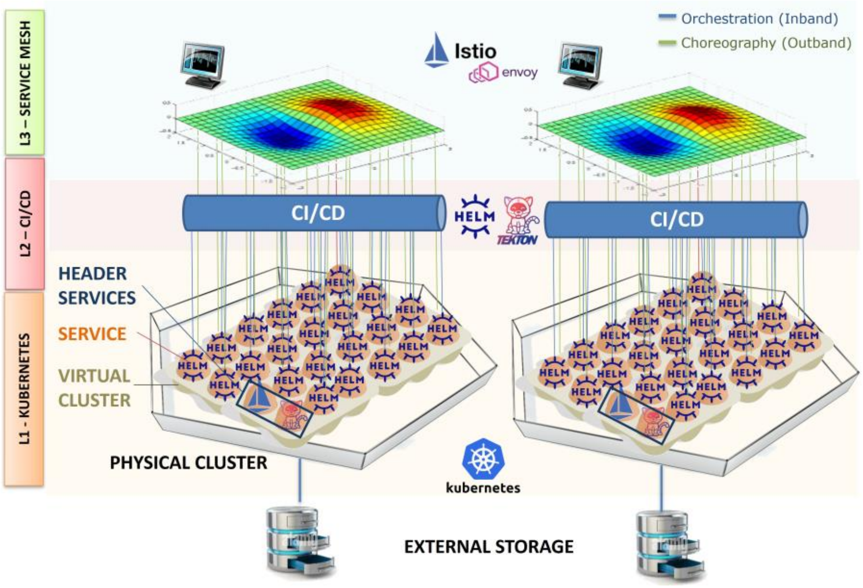


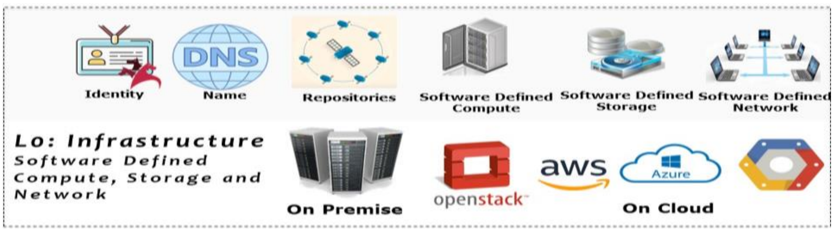


CONTINUOUS DELIVERY PLATFORM

PLATFORM ONE
WATERFALL, ONLY YOU CAN PREVENT

https://p1.dso.mil





SOFTWARE DEFINED DATA-CENTER (UNF)

CLOUD ONE
FAST
SECURE
STREAMLINED

https://www.cloud.mil/

LAYER	GOAL	TECHNOLOGIES
L0 **Infrastructure (IaaS)**	• **Physical Hosts**: deploy and control of federations of clusters of computers from a central header over a mesh of physical or virtual hosts. L0 methodologies usually called NetOps that produce *"Infrastructure as Code"*.	• OpenFabrics Alliance. • Cisco Application Centric Infrastructure (ACI) • Juniper Apstra • Arista CloudVision • Nokia Data-Center Fabric • OpenStack, CloudStack
L1 **Plataform (PaaS)**	• **Logical End Points**: instantiate pods (with associated containers) of a service over a cluster provided by L0 infrastructure layer.	• RedHat OpenShift • Novell Rancher • Canonical Charmed Kubernetes • VM Ware Tanzu
L2 **CI/CD**	• **Services**: continuous provisioning and update of services deployed over logical end points (usually several front-ends and one back-end) provided by L1 platform.	• Helm Chart • RedHat OpenShift Pipelines • Tekton • Jenkins, Jenkins X • ArgoCD, GitLab
L3 **Service Mesh**	• **Application**: automate the deployment of all services that compose an application (with his six main strategies: recreate, ramped, blue/green, shadow, canary, a/b testing) with monitoring and log handling, in other words, to assemble all services provided by L2 continuous delivery system.	• RedHat OpenShift Service Mesh • Istio • Traffik
L4 **Serverless (FaaS)**	• **Applications Ecosystem**: system of contexts to create integration models for application design, meaning, create the environment to easily create ecosystems of applications, just as application servers does.	• RedHat OpenShift Serverless • Knative

3.3.2.- ARTICULATION LAYERS: LOGICAL RESOURCES MONITORING AND CONTROL.

Articulation layers centrally monitor and control the ecosystem of services-oriented applications. In latter page picture, it is represented by a double blue arrow labelled as *"Continuous Monitoring"*, meaning that these layers are transversals across operation layers, in other words, they coordinate operations across all layers of the structure to easily manage the ecosystem of services. While the physical resources are handled by *"OpenFabrics Mangement Framework"* on each data-center, the logical ones are controlled by an application controller with following responsibilities:

LAYERS	GOAL	TECNOLOGIES
A0 **Coreography** Ecosystem of Services (Outband)	• **CMP – Continuous Monitoring Platform**: central control of a federation of services meshes across the data-center. Monitoring is based on side-car container, which integrates logs information with HTTP monitoring tools (such Jaeger).	• Dynatrace • Datalog • SolarWinds • IBM Instana • Sidecar Container Security Stack
A1 **Orchestration** Service Life Cycle (Inband)	• **SDP – Service Deployment Platform**: bootstrap sequencing of the continuous delivery platform and centrally control deployments: 1) creation and 2) initialization of the network of clusters, 3) assign pipelines for artifacts deployment to different clusters across the network; 4) start the continuous monitoring platform.	• RedHat Advanced Cluster Manager • Open Cluster Management • D2IQ

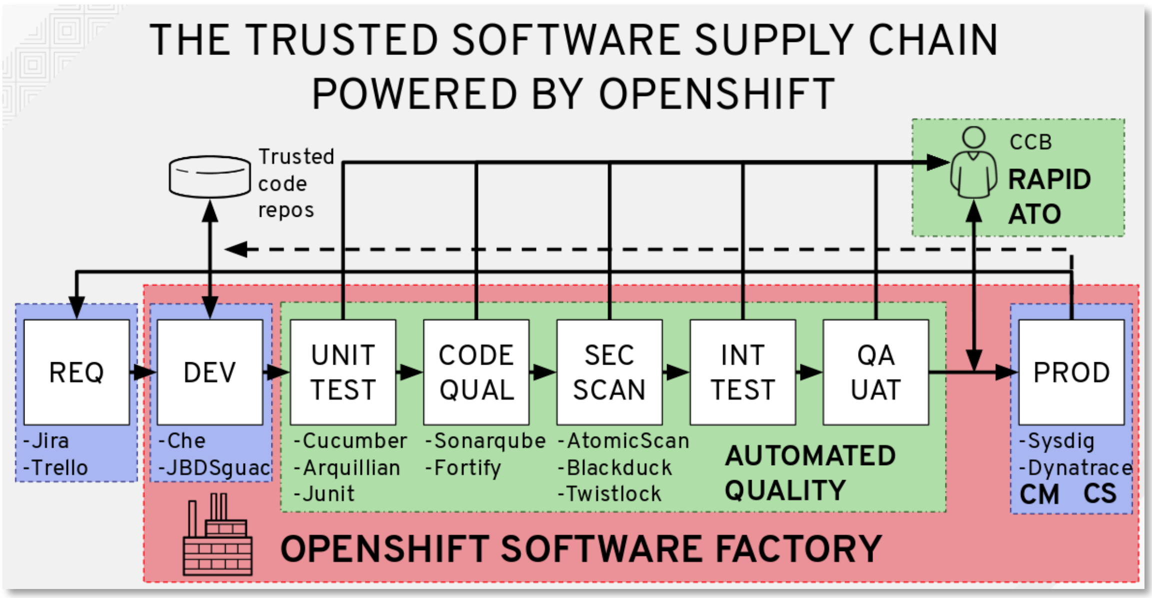
4.- *FACTORY*: AUTOMATING THE APPLICATIONS DEVELOPMENT.

4.1.- ARCHITECTURE OF PROCESSES IN APPLICATION DEVELOPMENT.

The starting point would be to **standardize the structure of processes involved in a DevSecOps factory**[11] **through European institutions** such ETSI.

From a well-defined structure of responsibilities, the tools[12] that each process needs to perform its duties successfully emerge. *Depending on the type of applications produced by each factory, a different toolbox would be required.*

In the picture, a summary of the most common tools in each stage of the application production lifecycle.



Security[13] must be present in every stage of the DevOps life cycle applied by software factories, however, since an holistic approach involving the entire software supply chain is required; both decision-making on measures to be applied at each stage by the different factories, and performance evaluation of these security measures with the associated corrective tasks, are carried out by a process in parallel to the production one... specialized in improving the computer security of each application independently, as well as together within the ecosystem of applications where it will be integrated.

---

[11] *IBM RedHat Secure Software Factory:* http://redhatgov.io/workshops/secure_software_factory/
[12] *Michael Bryzek, Design Microservices the Right Way:* https://youtu.be/j6ow-UemzBc
[13] ***Nokia Berlin Security Centre**, application security analysis and continuous improvement:* https://youtu.be/JIEoRChIus8

4.2.- TOOLS FOR EACH PROCESS: REDHAT CODE READY PORTFOLIO.

In the picture the integrated Application Development Suite for a DevOps methodology being developed by RedHat, whose trade name is RedHat Code Ready[14].

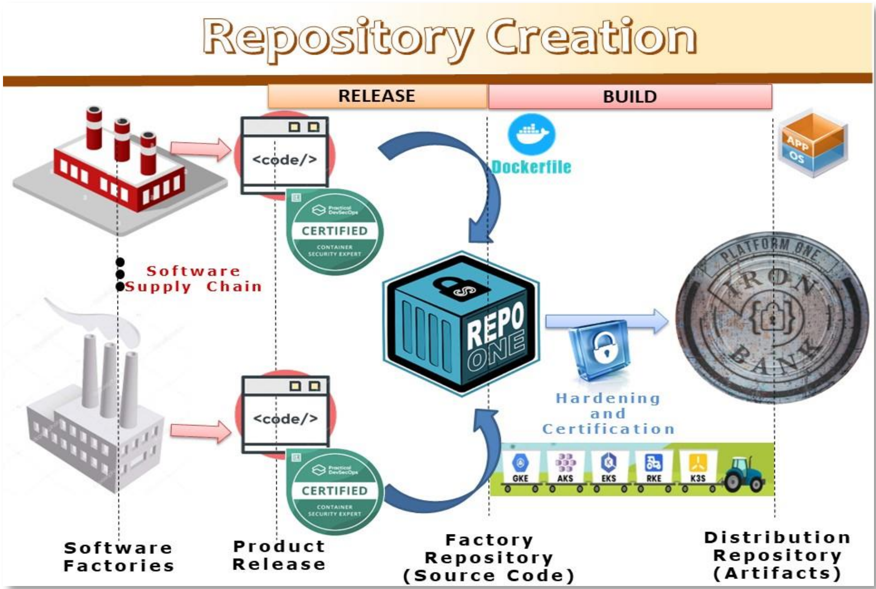
The suite is not complete, and needs to be extended with other tools, especially API validation[15], dependency analysis[16] and micro-segmentation. This implies a complex evaluation process until all these tools are successfully integrated into a final solution from which to create a single working methodology for the entire factory (similar to Metric v3[17] in Spanish State administrations)

- **Red Hat CodeReady Workspaces & Eclipse Che**: Eclipse based IDE to work with Kubernetes.
- **Red Hat CodeReady Containers**: laptop OpenShift cluster deployment.
- **Odo**: CLI to automate deployments abstracting all the technical aspects of Kubernetes. It can be integrated into Eclipse
- **Red Hat OpenShift developer console**.
- **OpenShift Pipelines and Tekton** for CI/CD.
- **OpenShift Serverless** and Knative.
- **VS Code / IntelliJ**: alternative IDEs.
- **Red Hat CodeReady analytics**: dependencies check.
- **Red Hat CodeReady toolchain**.

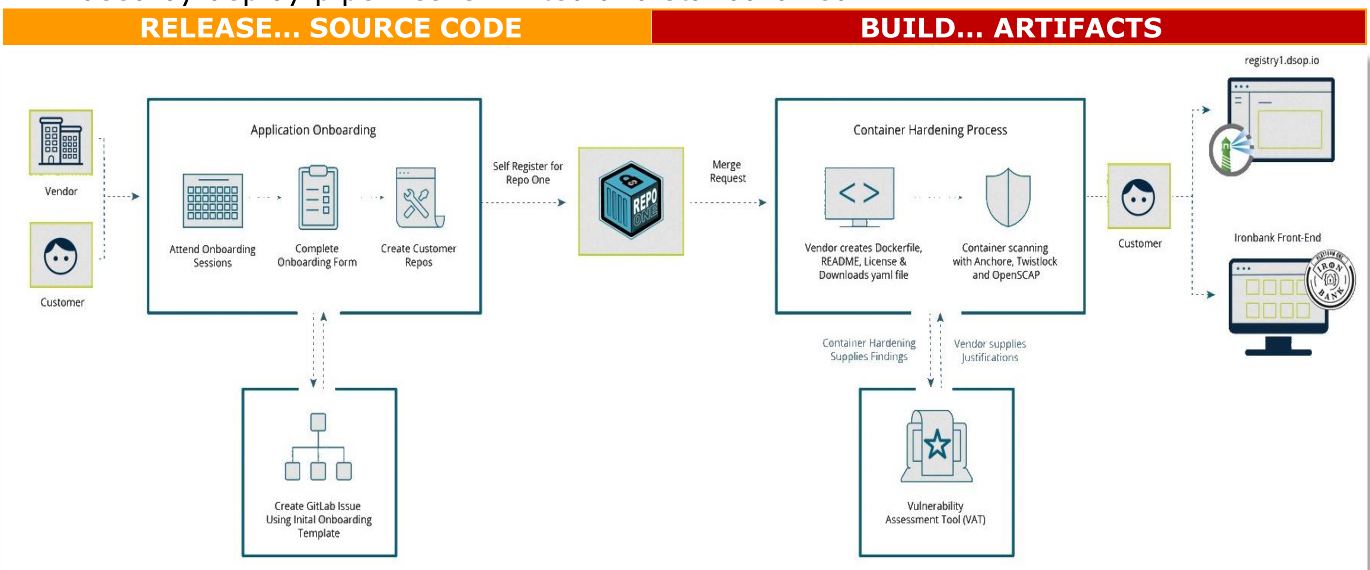
[14] ***Developer Tools, RedHat Code Ready Roadmap:*** https://developers.redhat.com/summit/2020/developer-tools-codeready-roadmap
[15] ***API Builder:*** https://www.apibuilder.io/
[16] ***Endor Labs, dependencies monitoring:*** https://www.endorlabs.com/
[17] ***Metrics v3:*** https://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Metrica_v3.html

4.3.- RELEASING APPLICATIONS: CENTRALIZED ARTIFACTS REPOSITORY.

In the picture how US Department of Defense distribute services across his ecosystem of software factories through the repositories RepoOne[18] source code repository and IronBank[19] artifacts repository.



Factories release a source code verified by a continuous authorization system. Then, a certification process (image below) builds, using the source code, the artifacts to be distributed and deployed in clusters. In development environments, there is no certification, instead the process is automated: a build CI/CD pipeline (which transforms the source code into artifacts) is linked to a deploy GitOps pipeline (that automatically instantiate artifacts throughout different clusters). In order to automate the process, the possible artifacts used by deploy pipelines is limited and standardized.



[18] ***Repo One,*** *DoD Centralized Source Code Repository (DCCSCR):* *https://repo1.dso.mil/dsop/dccscr*

[19] ***Iron Bank,*** *DoD Centralized Artifacts Repository (DCAR):* *https://docs-ironbank.dso.mil/overview/*

5.- SUPPLYING MEANS FOR PRODUCTION.

5.1.- CONTINUOUS DELIVERY PLATFORM.

The factories of all industries require sophisticated machinery for being able to produce what they must supply to society. In case of application factories, these are continuous delivery platforms that allow service-oriented applications to be deployed.

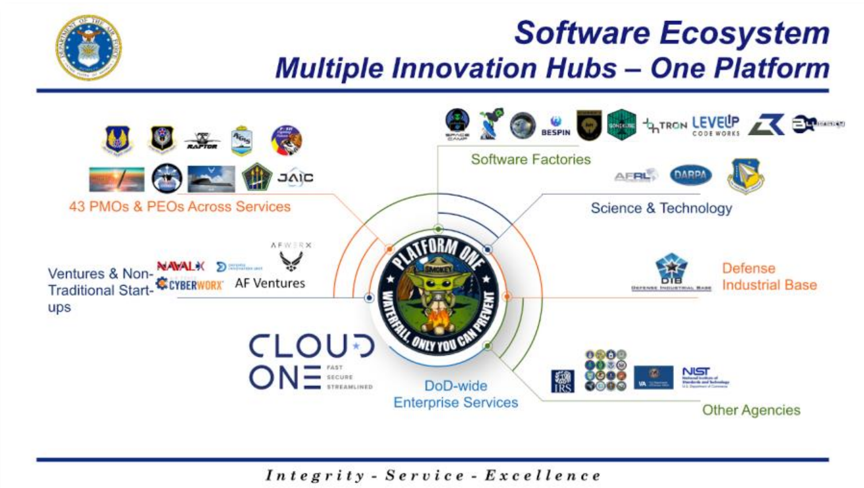
In computing, there is an anomaly consisting on application factories having the daunting task of assembling their own DevSecOps platforms, in other words, they need to manufacture, not only the application, but also the platforms in which these applications run. A task that they tackle without any guidance and based on millions of different pieces provided by open source. Both data center operators and application factories have two possibilities: either subscribe to large capacity platforms (such as Amazon); or build their own proprietary platforms with low performance and doubtful future viability.

Renting computing shared by millions of users (such Amazon) to host critical business logic is not a safe practice. Therefore, to reduce costs, operators shuffle complex balances between what part is hosted on external servers (such as Amazon), and what part on a more secure private platform, but with few capabilities and high cost.

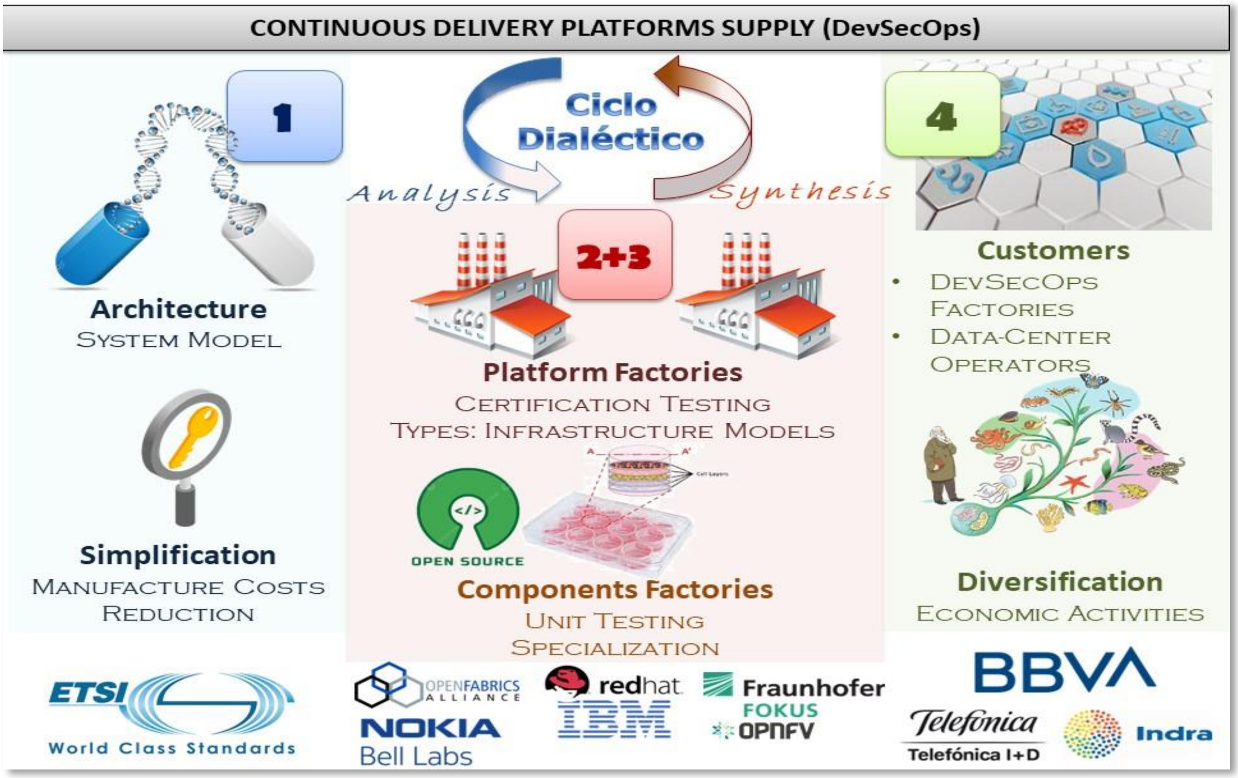
The end result of these hybrid structures, made up of scraps not designed to be integrated into a final structure (and often incompatible with each other and/or unfeasible in the long term) are platforms difficult to operate and maintain, with serious safety problems and exorbitant costs.

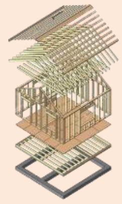
The need arises to establish a value chain capable of supplying this type of platforms, both to application factories and data center operators, avoiding all the security risks involved in renting shared computing capacity, in addition to simplifying the management of these platforms with specialized designs, greatly reducing operating and maintenance costs.

In the aeronautics sector, there is the exceptional condition of designing together both the factory as well as data center operator environments, which makes it privileged for the integration of an end solution capable of solving all cybersecurity issues at once; thus serving as a reference for a new software applications industrial fabric, the only way to address the dilemma of European digital sovereignty



5.2.- VALUE CHAIN STRUCTURE.



STAGE	GOAL	DESCRIPTION
1	 ARCHITECTURE	• **Architecture – System Model**: Standardization institutions, such as ETSI, coordinate the entire productive ecosystem thanks to a single system model for the platform, taking as a starting point the manufacturing specifications of the Cloud One and Platform One platforms of the United States Department of Defense, available online for the public.
2	 DESIGN	• **Design – Platform and Components Factories**: two decoupled pieces: ○ <u>L0 – NetOps - Software Defined Data-Center</u>: the physical infrastructure of these platforms. It can be a mesh of virtual machines through OpenStack or physical hosts through Fabric (OpenFabrics Alliance is the only open fabric solution). ○ <u>L1-L2-L3-L4 – GitOps - Continuous Delivery Platform</u>: there is only one solution on the market that contemplates the four layers of continuous delivery (Kubernetes, CI/CD, Service Mesh y Serverless): RedHat OpenShift.
3	 CERTIFICATION	• **Testing – Platform Homologation**: certification testing scaffolds to evaluate the different technological options, establishing infrastructure models for the different use cases, allowing versioning of each evolution path. OPNFV certifies 5G core networks over Telco Clouds, being Fraunhofer Institute its most prominent representative.
4	 DEPLOYMENT	• **Customer – System of Needs**: The evolution depends on the guidelines coming from the system of needs: the application factories and the data center operators of the different economic activities. The collaboration of strategic sectors is required, such as banking, telecommunications or aeronautics.

5.3.- RISK MITIGATION.

Individual companies that already tried to solve this challenge, such as *Sun Microsystems*, disappeared because of the high risk involved in such investment: the threshold for a commercially viable product is too high, it is easy to get stuck. Critical business data has a natural inertia to change.

Eventually, the reason why current investment is focus on establishing different Theme Parks where massive advertising provides quick return on investment, to the detriment of investments in the legitimate use of computing, which is nothing more than alleviating the administrative tasks.

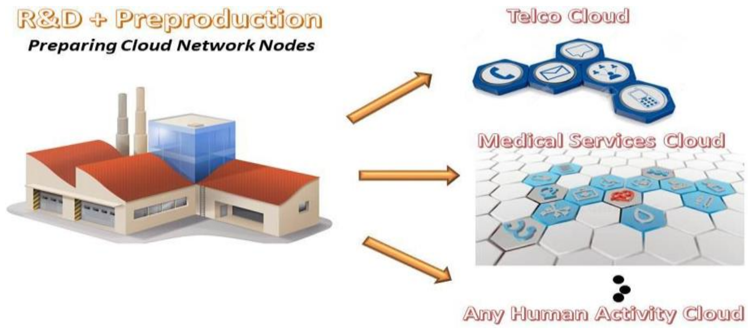
It becomes vital, then, to locate a methodology that overcomes all the difficulties involved in the production of this vital machinery. A risk similar to that assumed by IBM when it miniaturizes first computers, but resulting in a 90% market shared.

In this case, research starts from an already established base: the standards developed by the United States Department of Defense for all its application factories (Cloud One and Platform One). The research area is much smaller compared to the case of IBM and with some economic sectors forced to follow the very same path of Defense Air Force of United States for national security reasons.

Locating a methodology that mitigates risks means analyzing the point of view of each agent involved in this production process:

- **Data-Center Operators – The Needs**: as responsible for critical business data they will only invest in adopting new systems if they present very compelling advantages that worth the effort of adoption. Eventually an open process (similar to the *Java Community Process*) on a testing infrastructure, where the operators can evaluate the prototypes in addition to expressing their needs for their improvement, can speed up product acceptance times.
- **Manufacturing Ecosystem – The Interests**: computing is a recent sector, unconsolidated, compared to telecommunications or aeronautics. In other words, there is no tradition of coordination, there is no business model that guarantees greater benefits than working in competition. Just the economic sector that must update their application production environments can be the starting point for an ecosystem that will grow and diversify for a future miniaturization of these data centers, the only effective way for their democratization.
- **Standardization Institutions – The Costs**: data center operators suffer from certain symptoms. However, only an understanding of the entire production system is capable of accurately diagnosing the causes of these symptoms, which translates into minimizing the costs of resolving the needs raised, guaranteeing future viability of the entire production process. Public financing gives the necessary stability to this process of normalization of the structure, reducing the risks of a lack of government model.

6.- BIBLIOGRAPHY.



STATE OF ART	
IBM Secure Software Factory	http://redhatgov.io/workshops/secure_software_factory/
Thomal Erl, SOA: Analysis and Design for Services and Microservices	https://www.arcitura.com/books/
MuleSoft Microservices	https://youtu.be/SouNISAnXlo
Universal Networking Fabric, List of SDN Controllers	https://en.wikipedia.org/wiki/List_of_SDN_controller_software
Cloud Landscape	https://landscape.cncf.io/
IDC, Cloud Centric Infrastructures	https://info.idc.com/cloud-centric-digital-infrastructure-infographic.html
David Cheriton: Arista/Apstra OS	https://youtu.be/LA_LEdV8Cq4
Nokia, The Universal Networking Fabric	https://onestore.nokia.com/asset/212701
Dimitri Stiliadis, Nokia Nuage Networks architect	https://youtu.be/O7UrGrjnYV4?t=88
Microservices Architecture	https://youtu.be/j6ow-UemzBc

CHALLENGES	
Stanford, Cloud Strategies	http://web.stanford.edu/class/cs349d/
Stanford, Zero Trust Discussion	https://youtu.be/ooAPzzYkyaE?t=3593
Rawlinson Ribera, VM Ware, Data Fragmentation	https://youtu.be/dFySwm2bKTg?t=220

EUROPE, DIGITAL SOVEREIGNTY	
GAIA-X	https://www.data-infrastructure.eu/GAIAX/Navigation/EN/Home/home.html
Oliver Wyman	https://www.expansion.com/economia-digital/2020/11/22/5fba2e48e5fdea66688b458c.html

PROJETS OF REFERENCE	
OpenFabrics Alliance	https://en.wikipedia.org/wiki/OpenFabrics_Alliance
Platform One, Air Force	https://p1.dso.mil/#/
Karl Isenberg, D2IQ	https://www.youtube.com/watch?v=qku6ilFG5RM
Java Community Process	https://www.jcp.org/en/home/index
Data-Center OS	https://cs.stanford.edu/~matei/papers/2011/hotcloud_datacenter_os.pdf

FRAGMENTED PRODUCTION ECOSYSTEM	
Giuseppe Carella, FOKUS	https://youtu.be/nybxtzYY0NU?t=2271

TELCO CLOUD	
OSM ETSI	https://osm.etsi.org
OPNFV Pharos Lab	https://www.opnfv.org/community/projects/pharos
Enterprise Cloud Simulation	https://jwcn-eurasipjournals.springeropen.com/articles/10.1186/s13638-019-1493-2

RESEARCH LINES	
Single Unix Specification	https://es.wikipedia.org/wiki/Single_Unix_Specification
Constellation System	https://en.wikipedia.org/wiki/Sun_Constellation_System
INCOSE, International Council for Systems Engineering	https://www.incose.org/