IoT meets Agile

Rev. 2.0.0 – September 2024

AgileIoT

an AgileConstellation Star

Summary

Introduction

Welcome to the world of **AgileIoT** and its two operational frameworks **AgileIoT Ductile** and **AgileIoT Fiotto**.

AgileIoT looks at projects in *the* world of the Internet of Things *(IoT) in an agile* way, proposing a series of tools for their governance and execution.

Both AgileIoT Ductile and AgileIoT Fiotto embrace the *Philosophy, Principles and Practices* of the **AgileConstellation Manifesto**[1], providing the Team with two frameworks to work in a *Continuous Value* and *Continuous Deployment* key, supported by the "pull" logic.

The starting point is the successes achieved in the Agile and Lean fields in recent decades, but adopting a critical eye that takes into account what have been the main problems, especially in attempts to apply the same concepts to the world of hardware.

In particular, *AgileIoT Ductile* provides a holistic solution, suggesting an approach in which the different elements are considered part of a single integrated process. It is therefore not an over-process framework that tries to cadence the activities of teams specialized in software and hardware development, in order to create a *schedule* for the moments of *big-bang integration*.

Dually, *AgileIoT Fiotto* provides a *Visual Management tool* that takes into account the specificities of IoT projects and allows you to have a quick and complete view of the progress of the project and the related problems. The use of AgileIoT Fiotto allows you to obtain strong benefits in the overall management of the project, having readiness of the overall aspects of implementation and those specific to the individual elements of Value.

In both solutions, the relative *Working Team* is a "real" team that works closely together on a daily basis and has common and shared goals. The goal is to develop end-to-end solutions, focusing on the *Value* to be created for stakeholders.

[1] www.agileconstellation.info

1. Internet of Things

1.1 What is the Internet of Things

The term **Internet of Things (IoT)** means the capability, for two or more objects or real places to be connected, using the existing network infrastructure. This possibility makes them able to communicate and adapt their behavior on the basis of events and the evolution of the context.

In the IoT universe, there are two main elements: **Smart Things** ("T", smart devices) connected and managed through the **Cloud** ("I", Internet) also used to compute and analyze the Information. With the term Cloud we mean both meta-architectural approaches: *public* Cloud and *private* Cloud.

Although the idea is recent, the idea of getting devices connected to a network to have a constant flow of data could be dated back to 1982, when the ***Carniege Mellon University*** modified an automatic soda distributor to connect it to the network and finally get the amount of cans remained, in real time.

The first formalization of the **Internet of Things** is back to 1991, when *Mark Weiser* wrote an article titled "The Computer for the 21st Century" where he images a future with "*hardware and software, connected with cables, radio waves and infrared, will be so ubiquitous nobody will be aware of them anymore"*

It's a 30-year-old idea, now reality thanks to the evolution of the technology, the Internet broadband, the Cloud and the penetration of the IT in every part of our lives. *Gartner* estimates that, 26 billions of devices will be connected within 2020; *Pew Research Center* highlights (survey 2014) 83% of IT experts agrees on the fact that, within 2025 the Internet perception will completely be bound to the IoT thanks to the explosion of various market sectors like *domotic, robotic, wearables and automobiles*.

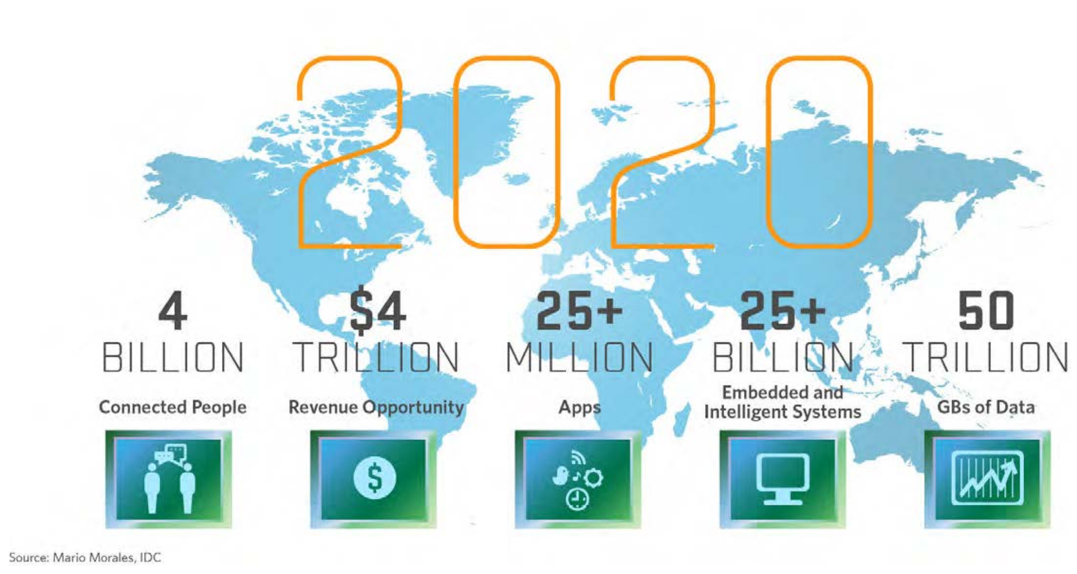


Figura 1 - IoT in the 2020, IDC Research

1.2 Main application fields

Smart Car, *Smart Home* and *Smart City* are main areas of growth for the IoT, highlighting the vocational multi-disciplinary of the IoT itself.
Let's think about the South Corea case: in *Songdo*, 65km away from Seoul, it was born the first *Smart City* in the world, a completely connected city, continuously gathering data regarding tens of areas, constantly computing them, without the need of human intervention.

The integrated city project in *Santander*, Spain, is interesting: about the 10% of the population use a Mobile App; the App gathers data from different sensors distributed along the city, providing real-time information regarding traffic, pollution, parking lots and other events.

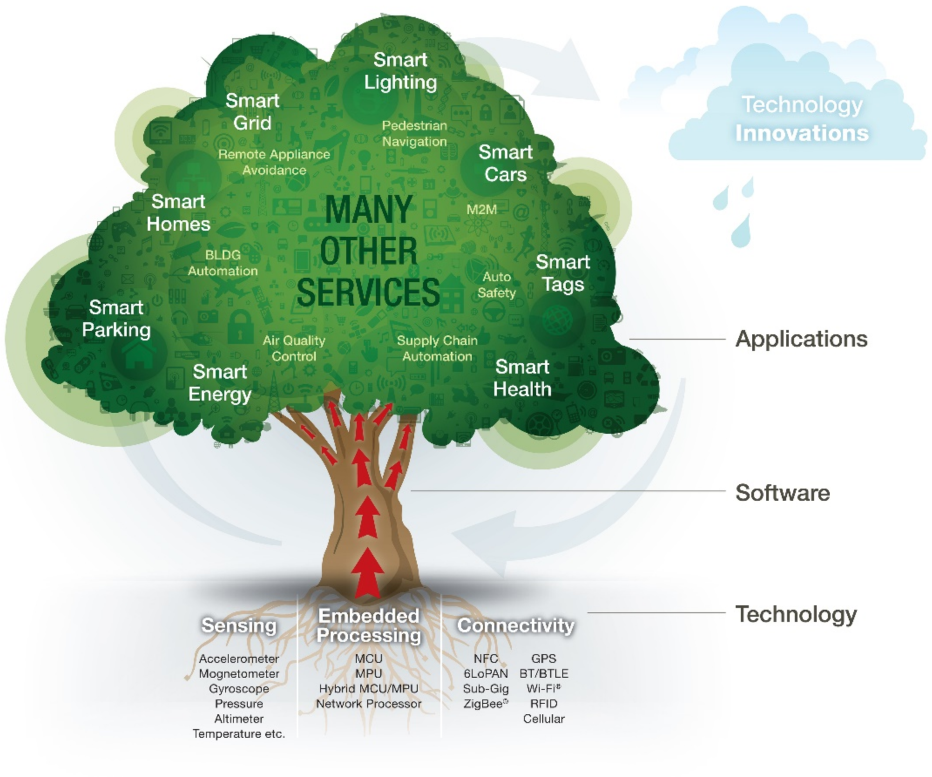


Figura 2 - Fields of Application

1.3 Challenges: privacy, security and energy saving

Privacy is one of the big problems related to IoT: more devices connected means more personal data traveling over the network. This requires to focus immediately on the implications on privacy that IoT solutions can bring.

The **data security** related to the equipment control and management is strictly linked to the privacy as it affects and impact the *real world*: an attack to in-car equipments, for example, could lead to real disasters.

Although these two topics are already analyzed and challenged, the **energy saving** topic is, on the other hand, less obvious since we tend to take for granted that the devices don't have energy problems.

What happens, though, if the device is, for example, on a marine platform where energy is limited and optimization goes to few watt-per-hour level?

It is clear that energy can become a driver for the design and development of the entire solution IoT.

2. IoT Project Governance

2.1 The multi-disciplinary scope

An IoT solution belongs to **different technological domains**, each of which, with their own rules and specifications. Thus, the project (or projects) should be, regularly, checked against different aspects: the *Hardware* – with strong and predictable processes – the Software, where the complexity and the change is always present, *the Cloud and Big Data* with its own rules and risks. Today, typically, hybrid *software* and *hardware* solutions are developed in a parallel way with determined synchronization meetings useful to start the integration process and, subsequently, to the test process.
This is a typical *Big Bang Integration* approach, with different critical issues:

- *Development efficiency is heavily penalized as different Teams don't constantly communicate with each other, without sharing their knowledge*
- *Issues are discovered in the late solution development phase, with high fixing problems, mostly due to the hardware fixing.*
- *As of the need of shortening / respect the Time-to-market, the final product, often, has a compromised solution to problems (based on software workaround, wherever components cannot perfectly integrate with each other), thus lowering the global product quality.*

The following sections will detail the primary aspects of the different domains.

2.2 The hardware aspect

The development of **Hardware** typically follows an approach waterfall-like, with a linear process optimized over the years and quite rigid compared to changes during construction. Roles and skills are better defined, including, at least:

- *the hardware designer,* which generally has the task of creating the schematic, establishing, for example, that the processor is connected to the RAM and FLASH, that there needs to be a level translator, a resistor here ... a capacitor there, etc.
- *the "sorter",* which transforms the circuit diagram (for which there are no "physical" positions on a Board) in corresponding physical schema, in which, on the contrary, the components are placed on the actual circuit by placing the connectors matching the holes of the mechanical designer. His task includes the definition of "slopes" that connect the components, etc. The *Bill of* Materials is created starting from actual electronic components;

- the *mechanical designer*, defines *mechanical* parts of the devices. By example, he/she defines the box containing the device, holes positions for connectors, sizes etc.
- *The software developer* (*driver e Board Support Package*), is in charge of developing drivers for devices. In the case a real Operating System is used on the developed device, he/she is also in charge of generating the customized OS Image. This is mostly true for systems based on *Windows Embedded Compact* and *Linux*. We must emphasize that, typically, the developer has profound hardware and firmware knowledge as it also plays the role of test driver of the hardware product. With a series of tests, he validates the proper connection of the wires and the work of the sorter and/or designer to the extent that they have done something wrong during the wiring (for example, placing a pin of the processor I/O memory power pin);
- *The software developer* for the application part of the firmware, making the firmware interact with devices (through software drivers), analyzing signals, sending/receiving data, etc.

2.3 The software aspect

Today, the development process of the **Software Component** is typically governed by Values and Agile principles [2]. Teams use a mix of practices to maximize the solution value. The approach is *iterative* and *incremental*, in order to quickly get feedback from an even more complete versions of the product; thus, activities are re-align on those bases, Team maturity and the progress of project activities.

Another aspect to highlight is the cross-functional vocation of Agile Team members: developers can work on the entire project, leaving it free from specific roles or persons. Further roles can support developers: one can be the *Scrum Master*, in the *Scrum* process; He / She plays the role of facilitator and culture-spreading among the whole company context.

Typical Agile roles are:

- the *Product Owner*, who is responsible for enhancing what has been achieved by the Team according to the objectives of value required by the customer;
- the *Scrum Master*, who is responsible for disseminating the Agile / Lean Culture within the specific team and the general company environment;

the *developer*, who has the task of implementing the project effectively

[2] Agile Manifesto, agilemanifesto.org

2.4 The cloud aspect

Together with the classic Hardware/Software dualism, an IoT project adds the specificity of the **Cloud Component**. It's not as simple as "*publishing own services and database on a remote VM*", but a totally new development paradigm, able to achieve and make own solutions available as commodities: Do I need it? Then I'll pay for that, otherwise I turn them off and I don't.

In order to achieve this objective, it is fundamental to think "for the Cloud" since the beginning: an example is the one related to the development of "elastic" services. They scale horizontally which means computing power is increased by increasing the number of machines and then balancing the computing load among all of them. This architectural approach is totally opposite to the typical "vertical scaling" wherever the increased computing power is obtained just by upgrading the (single) machine hardware (with consequent physical limits).

The same approach is the one related to the *Big Data* that needs to be managed with extreme efficiency and modern solutions (think of the storage in a NoSQL Repository), in order to make data readily available to be (eventually) transformed before consuming them.

The Cloud component can be approached using Agile methodologies but platform aspects (or, System Engineering aspects) need to be considered; this is the reason behind we often refer to *DevOps* with both meaning of software development methodology and a role with competencies between Developer and System Engineer.

2.5 The governance complexity

As from previous descriptions, the **governance complexity** of an IoT project is not an easy aspect: it needs proper tools in order to harmonize all activities and the workflow of different roles involved.

In order to do so, we can't keep *wall*s between development processes of main components; it's needed to create a uniform process based on the daily sharing of the project status and the quick integration of all components with the final goal of releasing a working solution.

As from previous considerations, it is evident it's not realistic to approach the governance and the development of an IoT solution by "lending" methodologies from software, hardware or Cloud, but it is needed to approach the problem in a holistic way.

Using such an approach, the software development and the hardware development are going to merge into a completely new disruptive context with specific rules and mechanisms supporting the specificity of the IoT. Think of the terminology, suddenly overlooked, but fundamental for the relation with stakeholders and the customer itself.

3. AgileIoT

AgileIoT starts from those considerations and is created on the *Philosophy*, *Principles* and *Values* of the **AgileConstellation Manifesto**[3]. It proposes a consistent methodology for the creation of Values in the Internet of Things world.

It helps to approach in an explicit manner the governance of the creation of a new IoT solutions, building it on the experience gained in the various disciplines involved.

The aim of AgileIoT is not trying to sketch adaptations from others discipline because they may be inadequate. This because typically the idea is to translate specific aspects of a single domain in a context in which the multidisciplinary is a carrier.

Rather, AgileIoT it suggests adopting a new approach that looks at the Internet of Things in a holistic manner, suggesting an approach in which the various elements are considered part of a single integrated process.

[3] www.agileconstellation.org

3.1 AgileIoT: an AgileConstellation Star

AgileIoT is constitutionally based on the Agile and Lean mindset, boasting the declination developed within the AgileConstellation project which looks at domains other than software and digital in general.

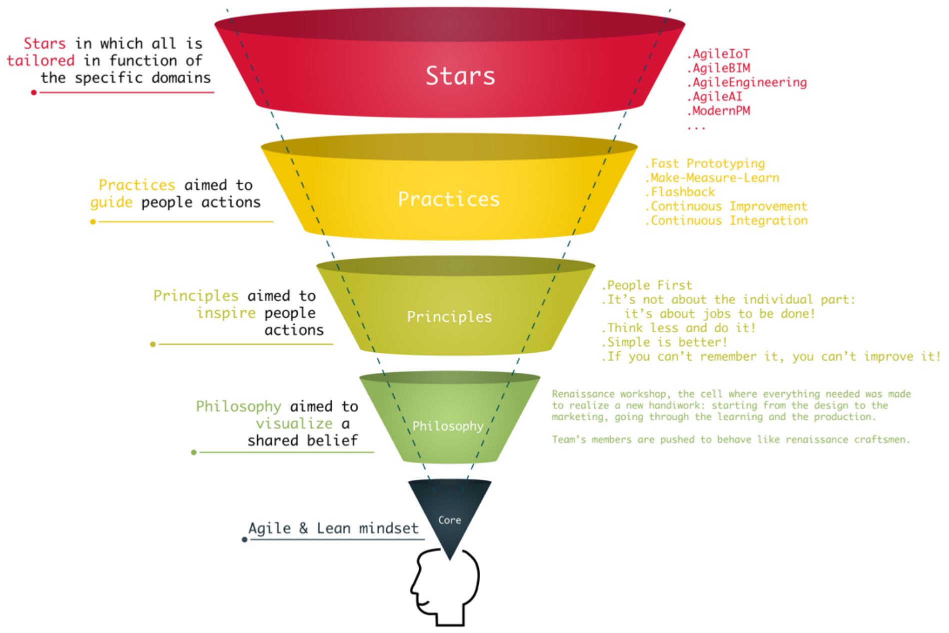

Stars in which all is tailored in function of the specific domains
Stars
.AgileIoT
.AgileBIM
.AgileEngineering
.AgileAI
.ModernPM
...
Practices aimed to guide people actions
Practices
.Fast Prototyping
.Make-Measure-Learn
.Flashback
.Continuous Improvement
.Continuous Integration
Principles aimed to inspire people actions
Principles
.People First
.It's not about the individual part: it's about jobs to be done!
.Think less and do it!
.Simple is better!
.If you can't remember it, you can't improve it!
Philosophy aimed to visualize a shared belief
Philosophy
Renaissance workshop, the cell where everything needed was made to realize a new handiwork: starting from the design to the marketing, going through the learning and the production.
Team's members are pushed to behave like renaissance craftsmen.
Agile & Lean mindset
Core

Figura 3 - AgileConstellation funnel

AgileIoT inherits the mindset, philosophy, principles (core) and practices (core) of AgileConstellation, declining and extending, in particular, these last two aspects in the specific domain of the IoT. So, we have:

- **Philosophy**, inspired by the **renaissance workshop,** where the *cell* does everything needed was made to realize a new handiwork: starting from the design to the marketing, going through the learning and the production.
- **Principles (core):**
    - It's not about the individual part: it's about jobs to be done!
    - Think less and do it!
    - Simple is better!
    - If you can't remember it, you can't improve it!
- **Practices (core):**

- *Fast Prototyping*, validate the sustainability of the solution
- *Make-Measure-Learn*, quickly experiment with different hypotheses and assumptions
- *Flashback*, *rapid alignment where is the observer to be responsible of it*
- *Continuous Improvement*, constantly improve every aspect
- *Continuous Integration*, constantly integrate the different souls of the solution

3.2 AgileIoT Fast Prototyping

According with the modular approach of the AgileConstellation Manifesto, the *AgileIoT Star*[4] added 6 new bubbles domains to the Fast Prototyping practices to validate the solution sustainability:

- **Security**: focused on the verification of the security aspects, which affect the realization of the solution;
- **Energy**: focused on the energetic-based aspects as a function of the needs of the operational continuity of smart devices;
- **Hardware**: focused on the validation of the hardware through one or more *Evaluation Kits* (EVK). The EVK is evaluated through CPU/MCU, and then with prototyping the other components, like RAM, USB, and so on;
- **Code**: focused on the prototyping of the firmware of the devices and the services made for acquiring the main data/events. In this phase, useful frameworks and productive IDEs can really cut the development time;
- **DataFlow**: focused on the aspects related to the gathering, cleaning-up and managing of the *Raw Data* that comes from the devices, with implementing the data transfer and serialization with the right protocols and the right format. It is very important to estimate the data volumes, the ways for analyzing them, in order to get the best approach possible, following the *Polyglot Persistence* pattern, which is fundamental for a big-data scenario.
- **Cloud**: focused on the Cloud aspects of the solution, as a data/event management platform.

[4] www.agileconstellation.org

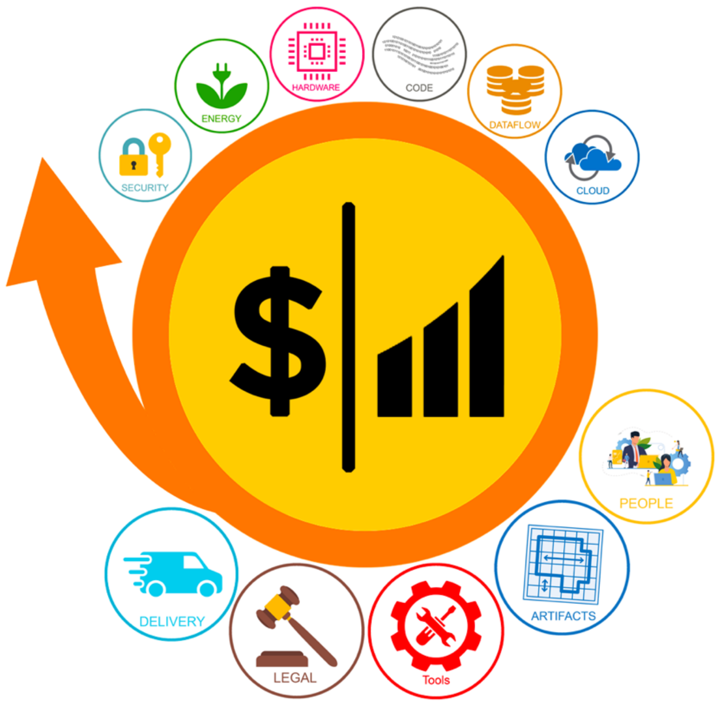


Figura 4 - The new 6 bubbles added by the AgileIoT domain

The *Product Backlog* can be designed deeply now, due to the prototyping phase outcomes. The *Product Backlog* will be created through the **Product Backlog Planning** (2), which is managed by the Product Owner (or by the Chief Product Owner, if there is more than one team). In this ceremony, the (Chief) PO, key stakeholders and the team will define the **Solution Big Picture** of the solution, together with the epics and their priority.

It is important to note that this phase also impacts on the company organization that must adequately support the development of the solution.

3.3 Frameworks

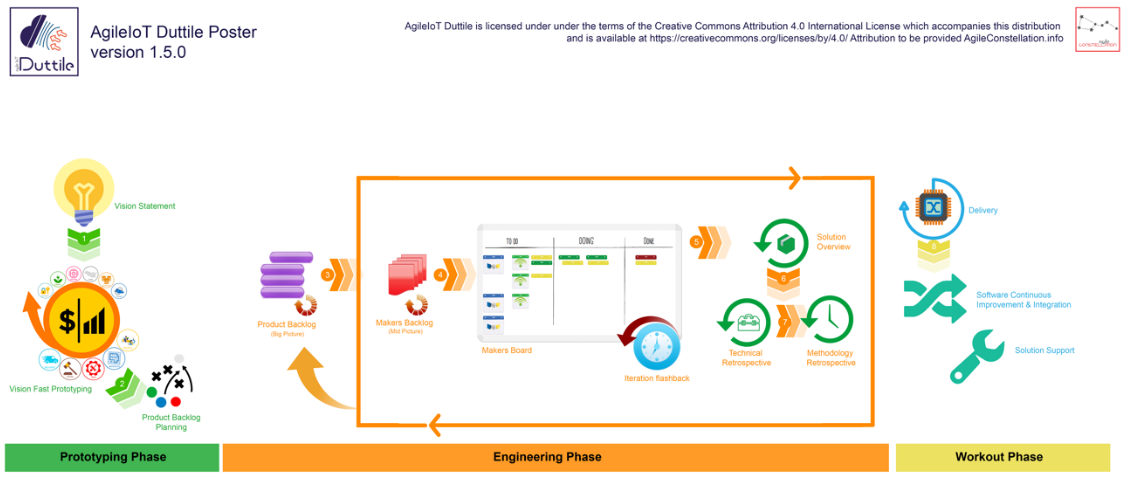
You can choice to implement AgileIoT through two different frameworks: **AgileIoT Duttile** and **AgileIoT Fiotto.**

In the following, we will go into detail about the two frameworks.

4 AgileIoT Duttile Poster

AgileIoT Duttile defines a **rich** process to create solutions bound to the Internet of Things world, with an emphasis on end-to-end solutions and their Value. In particular, the creation of a specific solution comes through three different well-defined steps:

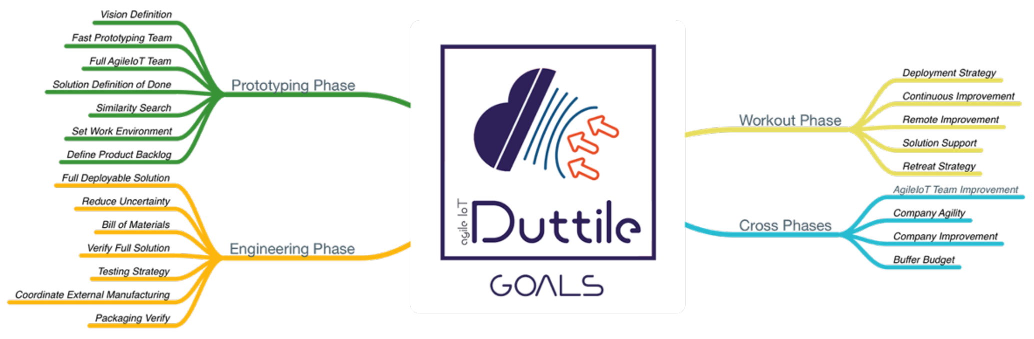
- **Prototype Phase** *(timing: typically, 2-4 weeks):* It's the first phase of the process. There's the Vision Statement, Fast Prototypization and Product Backlog definition using a specific planning phase;
- **Engineering Phase** *(timing: time needed for reaching the Value):* In this step, the solution is engineerized and developed. It's, as one could deduct, the most important and complex phase of the whole process;
- **Workout Phase** *(timing: Typically, 1-2 weeks):* It's the last phase, focalized onto the Deployment, Support and the Continuous Improvement of the product.



AgileIoT Duttile Poster
version 1.5.0
AgileIoT Duttile is licensed under under the terms of the Creative Commons Attribution 4.0 International License which accompanies this distribution and is available at https://creativecommons.org/licenses/by/4.0/ Attribution to be provided AgileConstellation.info
Vision Statement
Vision Fast Prototyping
Product Backlog Planning
Product Backlog (Big Picture)
Makers Backlog (Mid Picture)
TO DO
DOING
DONE
Makers Board
Iteration flashback
Solution Overview
Technical Retrospective
Methodology Retrospective
Delivery
Software Continuous Improvement & Integration
Solution Support
Prototyping Phase
Engineering Phase
Workout Phase

Figure 5 - AgileIoT Duttile Poster

The intrinsic complexity assigns an important role to the **Solution Definition of Done (sDoD):** it is the way to explicitly define WHEN the various *Events* and *Artifacts* of AgileIoT Duttile can be marked as “completed”.



AgileIoT Duttile Goals, version 1.5.0

Figura 6 - AgileIoT Duttile Goals

Over the goals of the individual steps, that we are going to find out soon, AgileIoT Duttile has several business objectives of growth and management, defined **Cross Phases Goal**:

- ***Team Improvement***, *Agile IoT Team operates in such a way to improve itself constantly, both from a technical and methodological point of view;*
- ***Company Improvement***, *the development of the solution should bring to the company an enrichment of know-how in combination with the improvement of internal processes;*
- ***Company Agility***, *business processes must become flexible and break down bureaucracy barriers;*
- ***Buffer Budget***, *the importance of identifying all stages, especially on the hardware, leading to the need for a budget for the "daily expenses". Consider buying a small resistor to the EVK (Evaluation Kit) for the cost of a few euros: for this activity, it is unthinkable that we should complex procurement processes that lead to weeks of waiting.*

4.1 Prototyping Phase

The **Prototyping Phase** is the phase in which the company decide to create a new solution and, starting from an idea or an initial need, defines the **Vision Statement**. With the Vision Statement the company highlights the specific objectives related to the customer needs, going to define the **Solution Goals** and the *validation metrics*.

This phase is a "discovery" and "validation" phase and it is essential to address elements related to the solution functionality and company organization to support the delivery.
To do this, you need to validate the **hypothesis** underlying the *Vision Statement*, structuring *teams* and *corporate assets* to be sure that the product is *achievable*. All this actions are made with the *make-measure-learn loop*.

Solution Big Picture and the reference *Product Backlog* are the primary artifacts.

In detail, for the prototyping phase, you have the following Goal:

- ***Vision Statement Definition,*** *here we define the Vision of the solution, identifying needs to satisfy and the reference market;*
- ***Fast Prototyping Team,*** *the Fast Prototyping Team is created, getting needed roles externally;*
- ***Full Working Team,*** *the complete* Working Team *is created;*
- ***Solution Definition of Done****, creation of the initial reference document for the Definition of Done;*
- ***Similarity Search,*** *as consequence of the Vision statement and the solution coming from it, the market is analyzed to verify if equivalent proposals are already present. In the case some others exist, we proceed by qualitative comparison to understand how our product could be positioned in the market to differentiate from them. It's a matter of understanding if we're in a Red Ocean (more players, rules already defined)* or *Blue Ocean (first player and rules to be defined);*
- ***Hypothesis Validation****, hypothesis is validated using the Fast Prototyping Events***;**
- ***Set Work Environment,*** *the work area is arranged or created. This aspect is relevant for the development of an IoT solution as the need of hardware prototyping requires properly equipped dedicated spaces;*
- ***Define Product Backlog****, the Product Backlog is defined on the base of prototyping results.*

These goals, owned by the *Fast Prototyping Temporary Team*, allows to achieve the **Vision Statement**, and to create the *Intentional Architecture*, to start the **Vision Fast Prototyping** (1).

4.2 Engineering Phase

With the **Engineering Phase** the implementation of the solution goes live. At this stage, most of the emphasis is placed on product development, but an attention is put on support aspects that must be supported by all the company.

This phase start with the definition of the **Mid Picture** (3) by the team (if there is more than one team, each team defines its Mid Picture). The Mid Picture defines the goal of the iteration in relation to the reference Goals:

- ***Full Deployable Solution****, the objective is the creation of a ready-to-use solution for the final customer;*
- ***Reduce uncertainty****, reduce to the bare minimum the uncertainty, particularly those related to the hardware and the firmware of Smart Things, services and Cloud aspects;*
- ***Bill of Materials (BOM)****, Smart Thing Bill of Materials is defined;*
- ***Verify Full Solution****, establish and validate objective elements to assess the actual adherence to stakeholders specific expectations;*
- ***Testing Strategy****, identify and implement appropriate testing strategies, functional and qualitative ones;*
- ***Coordinate External Manufacturing****, managing the relationship with the external manufacturing team responsible for implementing the Smart Things;*
- ***Packaging Verify****, implement packaging validation actions of the Smart Things and the whole solution.*

The Mid Picture is based on a set of *Epics* from which the **Makers Stories** are derived, during the **Makers Backlog Planning** owned by the PO. The **makers** define the **Tasks** of each Makers Story and locates them into the **Makers Board** (4).

The Makers Story with the lowest number of **Tasks** defines the **Rhythm** of the Mid Picture. During the Development Phase, rhythm dictate the frequency of the **flashback** Events.

When all the Makers Stories are completed, it means that every Task is completed too. The **Solution Overview** (5), managed by the Product Owner, will take place and the customers can view the running product.
In case of multiple teams, the Solution Overview can be delayed waiting for all the teams have completed individual tasks.

Then (6), it's the turn of the **Technical Retrospective**, useful for discussing about the chosen technologies and for making an important decision: *persevere* or *pivot*. In the end, it's the time of the **Methodology Retrospective**, useful for discussing about the team organization and methodologies.

Before starting a new iteration, Working Team (or teams) keep working with the grooming of the Product Backlog.

An important aspect is the *creation of the hardware component* of the Smart Thing. The EVK, which may be different than the one in the previous stage, will be progressively replaced by the final hardware.

We can identify three common scenarios:

- *Custom Hardware;*
- *Market Hardware;*
- *Hybrid Hardware, based on a mix between the Market and Custom ones.*

The first is the most common scenario. Everything is made specifically for the solution and evolves during the engineering phase. Typically, the work starts in the first iteration with the definition of the *Smart Things Bill of Materials* (BOM), using the EVK until the first unit of Smart Thing is ready and for the whole adjustment period. This is due to the problems related to the prototypes, especially the first models, which often have troubles highlighted only when control software is deployed on them.

The second scenario is more unusual. A market solution is taken and the control software is applied to it. This scenario drastically reduces the efficiency of the whole solution and every implementing activity of the solution becomes software-centric (it does not require any hardware design).

The third scenario is becoming more interesting every day. The CORE hardware (CPU and Memory) is supplied (already assembled) on a specific card. This allows to avoid any design/manufacture related problems such as, for example, tolerance to noise and the proximity of the slopes. The rest was designed and manufactured specifically according to the needs of the solution. In this case, the development cycle follows the custom solution steps.

In the two most common scenarios (first and third), the team will take care of producing the *Smart Thing BOM*, which evolves throughout the engineering phase, thanks to a special test suite specifically designed for test operation (testing both the firmware and software aspects).
Although it will be clearly of a different type, the test suite will be made also in the Hardware Market, as it's oriented to the verification of the functional and qualitative parameters.

Before going on, it is mandatory to clarify the Smart Things **manufacturing production** concept, when hardware is specifically made.

From AgileIoT Duttile point of view, *manufacturing* is always an external task, even if the product is made on the same company of the team. However, it is necessary to manage it according to deployment goal. The operative management is assigned to the *Prime Maker* (or the *Chief Prime Maker* if there are more teams), with the *Product Owner* (or the *Chief Product Owner* if there are more teams), which helps for management/administration aspects. These two roles must cooperate and create a trusted relationship between the different production units.

The (Chief) Prime Maker must support directly the hardware development team, using the BOM as a reference, in order to clarify the doubts and to respect the timelines on the engineering process.

Once you reach the desired level of stability, defined and proven by automated testing suite, the *Smart Thing BOM* takes the final form and should no longer be necessary to make further changes, because the costs would not be sustainable, especially when production is started. Here too, the (Chief) Prime Maker will have to manage the development phase of the Smart Thing in the final version.

4.3 Workout Phase

The last phase which affects the solution is the **Workout Phase**. In this phase, the product is put into operation (deployment) and all the services needed to support the customer are activated.

In detail, you have the following phase Goals:

- ***Deployment Strategy**, it is defined the deployment strategy for the whole solution;*
- ***Solution Continuous Improvement**, after the release, the solution is constantly improved, mainly with continuous firmware update, services update and cloud aspects update;*
- ***Remote Improvement**, implement technical and technological solutions that allow to update as much as possible directly from remote;*
- ***Solution Support**, it is produced all that is necessary to drive the use of the solution (for example, user manuals) and optionally activated a customer-care support;*
- ***Retreat Strategy**, choose the appropriate actions to retreat the solution at the end of its functional life or in case of serious malfunctions.*

When the solution is working, it's time (8) to support the customers and to constantly improve the solution itself, implementing updates on the firmware, services and cloud with **continuous improvement & integration** practices.

The hardware items will be update rarely (especially due to high costs), unless a new release is needed. It's mandatory to apply a set of tools for the **remote deployment**, especially when the number of the Smart Things is very large and geographically distributed.

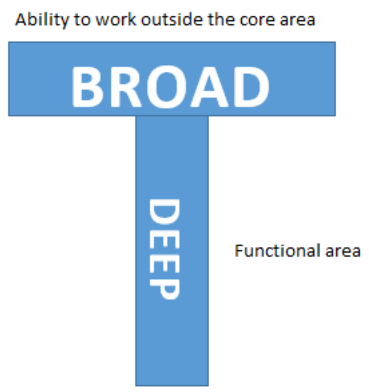
It must be remembered that the solution, unlike exclusively software or hardware solutions, has **two interaction points** with the customer:

- **cloud endpoint;**
- **hardware endpoint.**

In practice, the customer will have to work with the Smart Thing and the Cloud tools. These aspects require an efficient support enabled by on-site training, workshops, etc.

4.4 Working Team

AgileIoT Duttile provides a team that embrace the **T-Shaped** philosophy: team members often have vertical (*Deep*) skills in a certain area, but they can support the rest of the team on every activities (*Broad*).



The **Working Team** is formed by a variable number of **4:8 members**, which are *Primary Roles*:

- *Maker: 2, 4, 6*
- *Prime Maker: 1*
- *Product Owner: 1*

also, there are **4** additional *Specialist Roles*:

- *Cloud Closer: 1*
- *Data Flow Closer: 1*
- *Security Closer: 1*
- *Energy Closer: 1*
- *Legal Closer:1*

Totally, a Working Team can be composed of **8:13** total members.

The difference between *primary* and *specialist roles* is about their commitment on the solution. The primary roles are engaged for the whole development, and the specialist roles are focused on the prototyping phase. They support also the engineering phase. We strongly suggest getting both the roles, to get the highest possible quality of the solution. While the primary roles are involved full time, the secondary roles can be applied depending on the activities that are managed by the primary ones.

4.4.1 Primary Roles

T-Shaped Makers

The **T-Shaped Makers** (simply **makers**) are the *makers* that implement the *Makers Stories*, such the unity of work described later on this document.

Each Working Team provides an *even number of makers*, which are focused on 2 kind of primary skills: *Software Makers Oriented* and *Hardware Makers Oriented.*
Having “I” and “T” related knowledge in an Internet of Things solution, allows the team to face all the troubles in the best way. Software and hardware teams are not isolated, since they need to share the know-how and, eventually, to switch people temporarily.

Makers have the governance of the Makers Board.

Prime Maker

The **Prime Maker** (**PM**, also *Prime*) is the *Technological and Methodological Team Lead of the* Working Team. He can take care of both topics, as needed.
The Prime Maker helps the team on methodology, taking care of high-level technical aspects and holding the role of Solution Architect. He can replace a Maker when necessary, but only when explicitly requested by the other Makers. Then, the Prime Maker must create a “reference” solution, to provide technical stuff to the whole group. He is like a *technical evangelist*.

If there is more than one Prime Maker (there could be more than one team), it is necessary to form the *Prime Maker Board*, to align every Prime with each other, and also to identify the *Chief Prime Maker*.

The Prime Maker (or, if present, *Chief Prime Maker*) also manages the hardware supplying, for the Evaluation Kits and related components. Additionally, he takes care of the integration processes and milestone definitions for realizing externally the hardware, both for prototype and final release.

Product Owner

The **Product Owner** (**PO**) is the **interface** between stakeholders and the team. His primary role is to maximize the Value of the solution made by the team, based on the requirements requested by the customer and by the key stakeholders.

If there is more than one Product Owner (there could be more than one team), it is necessary to form the *Product Owner Board*, in order to align every PO with each other, and also to identify the *Chief Prime Owner*.

The PO (or, if present, *Chief PO Maker*) has the *Product Backlog* exclusive governance and the responsibility of the grooming tasks. In the same way, the specific PO (or the ones), has the ownership of the Makers Backlog. The (chief) PO actively participates to the Solution Overview and he can be involved in all other process phases, when the team ask for him.

Last but not least, the (Chief) PO supports the realization phase of the Smart Things (followed by the [Chief] Prime Maker) from an administrative perspective.

4.4.2 Closer Roles

Energy Closer

The **Energy Closer** is the expert of the energy aspects. He is a considerable role, dedicated to the *Smart Thing operations*, that is the capability of the Smart Things to do efficiently their work in the defined time box. Thus, components like *accumulators* become important as well as the *charge source*, especially whether the Smart Things must run on zones with no electric coverage. In this case, for example, solar power is needed. The role of the Energy Closer is help team to be steadily focused on *energy* aspects.

Data Flow Closer

The **Data Flow Closer** is the specialist who is responsible of the set of technologies and tools involved in "**harvesting**", "**cleaning**" and "**managing**" the *Raw Data* sent by the devices. The Data Flow Closer starts from the raw data (which is the output of the device), then choose the right transfer protocols, the serialization mode, the clean-up tools and the best storage solution for implementing the *Polyglot Persistence*.

Cloud Closer

The **Cloud Closer** is the architect of the Cloud aspects of the solution, related to the data/event analysis and to the response actions. We are speaking about **BigData** and **Event Driven Actions** where it is mandatory to have Cloud solutions ("I") which can manage such topics in a reliable way.

The Cloud Closer role is help team to be steadily focused on the Cloud aspects.

Security Closer

The **Security Closer** is the architect of the Security aspects of the solution, which are fundamental due to the importance of the data, of the events and of the related actions. This role is very important, since the IoT field (a brand new field) could give its side to reliability and security issues, amplified by the *headless & connected* usage. The role of the Security Closer is help team to be steadily focused on *security* aspects.

Legal Closer

The **Legal Closer** is a non-technical specialist whose role is to support the team in the analysis of the Law and Regulations which must be observed for the overall Solution. For example, consider the different types of electronic certifications or the unauthorized wireless frequencies or the compliance with the regulations related to the data collected by the different devices. The scenario becomes more complicated in an international context where regulations can be very different in function of the country where the solution must be marketed.

Unlike other Closers, the Legal Closer will make an intense activity in the prototyping phase, and then became a consultant for the Team during all subsequent phases.

4.4.3 Temporary Teams

Depending on the specific phases of the process, Working Team members give life to **Temporary Teams** (T2) that absolve the specific needs of the context. The outcome of its activities is shared with the rest of the members during the retrospectives.

Fast Prototyping Temporary Team

The **Fast Prototyping Temporary Team** (**FPT2**) is the temporary team that deals with the Fast Prototyping, designed during the Prototype phase. It is formed usually by a Prime Maker, two T-Shaped Makers and, possibly, by the Closers, forming the core of the future **Working Team**.

Story Temporary Team

The **Story Temporary Team** (**ST2**) is the *temporary team* that deals with the single makers story development.

ST2 people can be *Cloud oriented* and *Smart Thing oriented*, depending on the activity, and they cover both "I" and "T". Indeed, two makers with different skills form ST2. This is a new way of looking at the *pair-programing* (and not just programming) practice, where ST2 becomes the "basic unit" for the whole processes.

4.5 Artifacts

Artifacts are the process items which focus the efforts of the team for getting the progress of the solution. Each artifact has its own governance that belongs to one or more Working Team members. Artifacts are steadily under review, which make it synchronized to the current project status and know-how of the solution.

Work Items

Epic

An **Epic** emphasizes the Value depending on the Customer, considering the different impacts on the solution development.

Here is an example: let's suppose to implement a traffic monitor in a certain street. We need to consider *the number of car in a timeline, the quality of the air, the proper functioning of the traffic lights, and so on*. Each of these factors is an **Epic**.

Each Epic has a set of *Acceptance Criteria*, that is an explicit definition of the expected behaviors in the intended operating conditions. They impact on both technical aspects, for instance functional tests, and support aspects, like the specifications that will be added in a datasheet.

Makers Story

The **Makers Story** (or Story) is the core element of the Maker activity, especially of the ST2. It's based on input/output of the devices IoT. This activity can be separated in two categories (related to "I" and "T"):

- ***Device Story*** are the Makers Stories related to the Smart Things development. They can be separated in:
  - *Measure Story*: these are the Makers Stories that measure the external parameters as environmental sensors. In this case, the input is the sensor data and the output is the value read (and eventually normalized);
  - *Act Story*: these are Makers Stories with actuators which uses the parameters for applying a function. In this case, the input is the control sensor data and the output is the outcome of the action (based on the applied function).
- ***Cloud Story*** are the Makers Stories focused on computing, analysis and processing on the Cloud:
  - *Data Story*: these are the Makers Stories related to the data computing. They are the core item of the compute;
  - *Event Story*: there are the Makers Stories related to an event/alert, not focused on specific data;

Using the same traffic monitoring example, we can get the "number of cars in a timeline". We can suppose the following Makers Stories:

- *Smart Things that counts the number of cars (device story);*
- *Sending of the data to the gathering and management system (device story);*
- *Analysis and management of the data in an aggregated format (cloud story);*

Since the Makers Stories represent business needs, such a distinction can be omitted. This is not true for the Tasks, described below.

Task

The **Task** is the unity of work, specialized to a single application area: *hardware* [*task*], *firmware* [*task*], *service* [*task*] and *cloud* [*task*]. It's important to use a set of different card colors, in order to get a clear representation.

Specifying the task type is very useful. For instance, making a hardware modification deeper in the project could be risky and expensive. Thus, can lead to a solution based on the firmware modification. Using the right task types, the adopted solution could be less efficient, but it will prevent the risk of high costs and time.

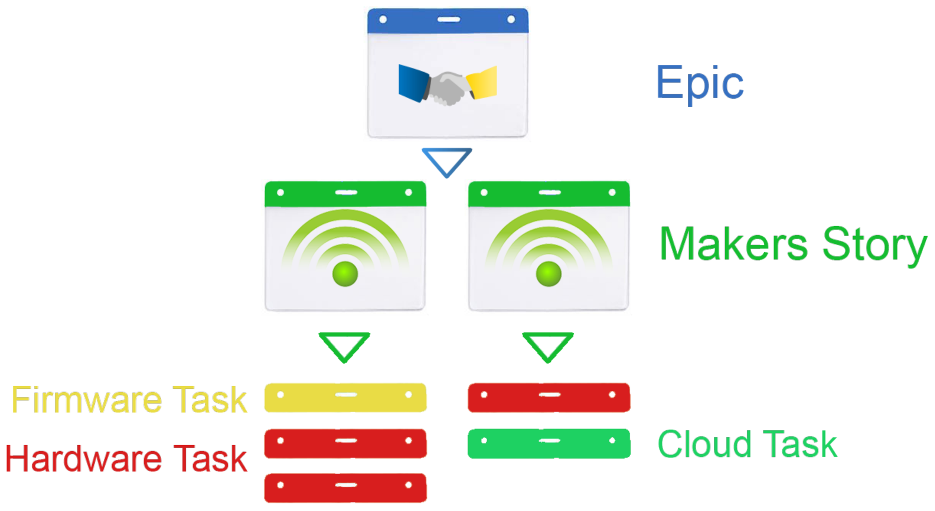


Figure 7 - Epics, Stories and Tasks

4.6 Visual Management

Product Backlog

The **Product Backlog** contains the **Epics** of the solution, priority sorted based on the customer. The Product Backlog is a simple stack of items, priority sorted, following a top-down pattern: the top item has the highest relative value.

The *Product Owner* (or the *Chief Product Owner* helped by the others PO) has the governance of the Product Backlog and of the related primary activities:

- *Product Backlog planning*: this is the planning of the solution high level Vision. During the Product Backlog planning, the Epics to be developed will be created, based on the needs of the (key) stakeholders;
- *Product Backlog grooming*: this is the Product Backlog governance activity, assigned to the Product (or the *Chief Product Owner* helped by the others PO). The Product Backlog grooming takes place after the Solution Review, when replying to the feedback gathered by the PO.

Makers Backlog

The **Makers Backlog** contains all the elements needed for create the Solution, starting from the Makers Stories, priority sorted based on the risk (explicitly) and the value (implicitly, related to the already sorted Epics).

The Makers Backlog is under the governance of the specific *PO*, who create it, with the makers, through the specific *Makers Backlog Planning*, the planning for the next iteration.

Makers Board

With the **Makers Board** it's possible to manage the Makers Story, and the others work items, development from a process point of view. Basically, it is separated in three columns:

- *To Do*, with the selected Makers Stories and the related Tasks;
- *Doing*, with the Tasks which are in development;
- *Completed*, with the completed Tasks.

Each *Temporary Team* can work on a single Makers Story and can have just a Task in "Doing" status. Thus, the ST2 will not waste resources in *Cross*- Makers Stories and *Cross-Tasks* activities (multi tasking) and it focuses on a precise objective instead.

The "Doing" and the "Completed" columns, have certain *Definition of Done*. The Definition of done of the "Doing" items are related to the Makers Story qualitative aspects whereas the ones of the “Completed” items are related to the operative aspects (the end user aspects).

Makers Boards is under *makers* governance.

Technical Debt & Risk Board

The **Technical Debt & Risk Board** is a tool for highlighting both the *Technical Debt* and the *Risks*. The first is a term used for indicating the potential problems related to hurry design of a complex solution, and the second indicates the risks related to mission-critical choices.

Every project has a certain percentage of risk and a technical debt, which constantly increases during the solution implementation. Tracing each aspect allows to act during the development, in order to leave the solution quality, the highest as possible.

Burndown Chart

The **Burndown Chart** allows the Working Team to monitor the trend of the tasks in development, correlating the Tasks (or the Epics) to "time" factor. It allows the team to act quickly to resolve the problems that create a deviation from the ideal performance.

4.7 Delivery Items

Smart Thing Bill Of Materials

The **Smart Thing Bill Of Materials (BOM)** is the list of all components, subassemblies, raw materials and semi-finished products required to realize the hardware.

Smart Thing

The **Smart Thing** is the IoT core intelligent component. The hardware component can be bought from a producer (if it’s already on the market) or built on demand for the specific purposes.

Solution to Delivery

The **Solution to Delivery** (**StD**) is the working and incremental solution. It must be shown during the *Solution Overview*, when the *Mid Picture* is achieved.

Generally, it includes:

- *One or more Smart Things;*
- *Cloud Solutions;*
- *Components of Delivery.*

The StD must be compliant with the *Solution Definition of Done*.

Solution Definition of Done Document

The **Solution Definition of Done Document** (**sDoD Document**) is a *live-document* that gathers the key items for considering the solution as completed. The goal is to clarify when the solution reflects the Value objectives and the process can move from Engineering to Workout phase.

The SDoD affects directly the Acceptance Criteria related to the Epics and Makers Stories, tracking the evolution of the specifications of the solution itself. In that way when an item is no longer functional to overall objective, it will not be deleted but marked as obsolete instead.

In general, the sDoD Document looks like a short list, drawn up in the first instance during prototyping.

4.8 Events

The **Events** are important moments in which all the planning and review actions are concentrated.

Vision Fast Prototyping

The **Vision Fast Prototyping**, that is based on the practice of the same name declined for the IoT domain[5], is the ceremony which is used for the quick prototyping as a function of its Vision.

It checks the *economical sustainability* of the solution and creates the initial steps for the *Product Backlog Planni*ng.

The *Vision Fast Prototyping* ceremony is responsibility of the *Fast Prototyping Team*.

The *Fast Prototyping* takes from **2** to **4 weeks** (average)

Planning

Backlog Planning

AgileIoT Duttile supports two principal **planning** Events: the *Product Backlog Planning* and the *Makers Backlog Planning*.

The *Product Backlog planning* purpose is to add items into the Product Backlog in order to achieve the Big Picture, that is the set of the valuable needs of the (key) stakeholders. In a similar way, the *Makers Backlog planning* purpose is to add items into the Makers Backlog with the Makers Stories based on the Epics, and the Tasks for achieving the Mid Picture.

The *Product Backlog planning* governance is responsibility of the Product Owner (or the Chief Product Owner if there is more than one team). The *Makers Backlog planning* is managed by the specific, or the unique, Product Owner.

[5] Take a look at the AgileIoT paper on www.agileconstellation.info for the details

The *Product Backlog planning* takes **16 hours** (average), grouped in **2** work sessions, while the *Makers Backlog planning* takes **4 hours** when the duration of the iteration is 2 weeks long (8 hours whether the iteration takes 4 weeks).

Refinement

The **refinement** is the activity for "cleaning" and "fixing" the *Product Backlog* and the *Makers Backlog*, based on the feedbacks and the evolution of the solution.

The *Product Backlog grooming* governance is responsibility of the Product Owner (or the Chief Product Owner with the assistant of the others PO), while the *Makers Backlog grooming* it's managed by the specific PO helped by the specific team makers.

The *Product Backlog grooming* could take **2** to **4 hours** for a 2 weeks iteration, based on the feedbacks. The *Makers Backlog grooming* can take max **1 hour**.

Iteration Flashback

The **Iteration Flashback** is the moment when the team shares its know-how and the task status of each implementation. Based on the *flashback* practice, it has a *Rhythm* (see below) and allows ST2 teams to meet with each other in order to ask a maximum of two aimed questions per ST2.

This **quick line-up** ceremony has been designed for the IoT world, where the solution is formed by hardware components, which cannot be repeatedly moved. Thus, the *observer* team will check the work of another team directly, replacing the classic Stand Up meeting.

The flashback ceremony, forces the teams to go deeper on some aspects of the solution, which can lead to a *Technical Review Blitz* in order to clarifying the doubts.

The flashback is managed by the **makers**, with the *Prime Maker support* when requested.

Each ST2 takes **3 minutes** for "observing" the others *temporary team* work. The feedback frequency is scheduled by the Rhythm.

Retrospectives

Solution Overview

The **Solution Overview** allows to show the STD to the stakeholders. During the Solution Overview, the Product Owner (or the Chief Product Owner helped by the others PO) will gather the feedbacks for implementing the Product Backlog grooming.

The solution overview is managed by the Product Owner (or by the Chief PO), with the Prime Maker (or the Chief PM) and the whole Working Team, when requested.

The average duration is **3 hours** for an iteration of 2 weeks.

Technical Retrospective

The **Technical Retrospective** purpose is to discuss about the chosen technologies and tools, based on the outcome results using them. Thus, each choice can be validated and new ones can be done: **persevere** or **pivot**.

Compared to the classic Agile approach, where the retrospective is typically only related to the process, AgileIoT Duttile explicitly cover a ceremony to talk about the technical aspects, especially in relation to the problems and solutions related to the hardware.

In the case of multiple teams, the Technical Retrospective is held individually by each team and, subsequently, the specific Prime aligns others during a Prime Maker Board meeting.

The *technical retrospective* is managed by the **Prime Maker**, (as facilitator) and the attendees are the whole Working Team.

It takes **2 hours** for a 2 weeks' iteration. 1/4 of the time of the data-related aspects.

Methodology Retrospective

The **Methodology Retrospective** takes places for discussing about the methodologies and the team organization. Its purpose is to achieve the continuous improvement pattern.

In the case of multiple teams, the Methodology Retrospective is held individually by each team and, subsequently, the specific Prime aligns others during a Prime Maker Board meeting.

The methodology retrospective is managed by the Prime Maker (as facilitator) and the attendees are the whole Working Team.

it takes **3 hours** (average) for a 2 weeks' iteration.

4.9 Metrics

The **Metrics** allows the Working Team to monitor the tasks trend and to design the Events and the related actions.

Rhythm

The **Rhythm** is the frequency for the flashback. Its value is based on the "smaller" Makers Storiy, that is the Makers Story with the lowest number of tasks assigned to it:

- *Story with 1 - 2 Tasks: Daily Rhythm, daily flashback;*
- *Story with 4 - 8 Tasks: b-Daily Rhythm, flashback every two days;*

- *Story with 12+ Tasks: t-Daily Rhythm, flashback every three days;*

Lead Time & Throughput

The **Lead Time** is the average time taken by a Makers Story in order to be completed (from the “To Do” status to the “Completed” one).

The **Throughput** is the estimation of the time involved for completing the Makers Backlog:

- *Throughput = All Makers Stories * Lead Time.*

5 AgileIoT Fiotto

AgileIoT Fiotto is the solution based on the "*Continuous Value*" and the "*Continuous Deployment*" pattern, for the IoT projects, following the *AgileConstellation Manifesto.*

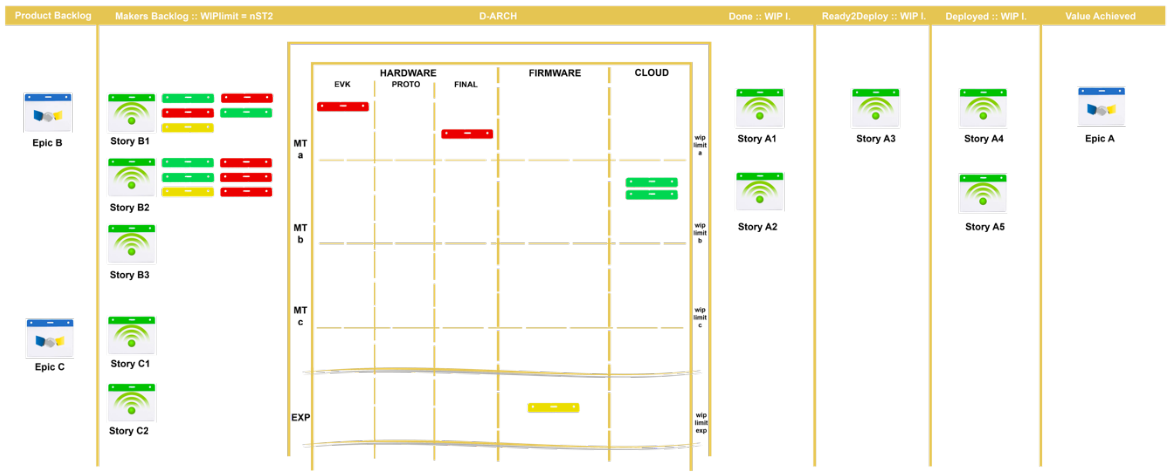


Figure 8 - AgileIoT Fiotto

It takes advantage of the **WorkPivot**, a transformation tool which allows to pass from the tasks of the whole *Fiotto Team* (vertical) to the specific Makers Team ones (horizontal).

This transformation is visible due to the **D-ARCH** (*Do it for… Achieve Rapidly Customer Hopes*), in which Tasks are represented by their particularity (*Hardware*, *Firmware* and *Cloud* – Y axis), by the Makers Team (X axis) and the related *Work in Progress limit* (WIP-limit / WIP-l).

5.1 Work Items

The **Work Items** are items on which the *Fiotto Team*, and the *Makers Team*, focus their actions. They help to manage and view the solution deployment status.

Epics

An **Epic** emphasizes the Value depending on the Customer, considering the different impacts on the solution development.

Here is an example: let's suppose to implement a traffic monitor in a certain street. We need to consider *the number of car in a timeline, the quality of the air, the proper functioning of the traffic lights, and so on*. Each of these factors is a Value Story.

Makers Stories

The **Makers Story** is the core element of the Makers Team. It's based on input/output of the devices IoT and can be separated in two main categories:

- ***Device Stories***, related to the Smart Things development. They can be separated in:
  - *Measure Stories*, that measure the external parameters as environmental sensors. In this case, the input is the sensor data and the output is the value read (and eventually normalized);
  - *Act Stories*: with actuators which uses the parameters for applying a function. In this case, the input is the control sensor data and the output is the outcome of the action (based on the applied function).
- ***Cloud Stories*** are focused on computing, analysis and processing on the Cloud:
  - *Data Stories*, related to the data computing. They are the core item of the compute;
  - *Event Stories*, related to an event/alert, not focused on specific data;

Using the same traffic monitoring example, we can get the "number of cars in a timeline". We can suppose the following Makers Stories:

- *Smart Things that counts the number of cars (device story);*
- *Sending of the data to the gathering and management system (device story);*
- *Analysis and management of the data in an aggregated format (cloud story);*

Each Makers Story has a set of *Acceptance Criteria*, that is an explicit definition of the expected behaviors in the intended operating conditions. They impact on both technical aspects, for instance functional tests, and support aspects, like the specifications that will be added in a datasheet.

Since the Maker Story represent business needs, such a distinction can be omitted. This is not true for the Tasks, described below.

Tasks

The **Task** is the unity of work, specialized to a single application area: *hardware* [*task*], *firmware* [*task*], *service* [*task*] and *cloud* [*task*]. It's important to use a set of different card colors, to get a clear representation.

Specifying the Task type is very useful. For instance, making a hardware modification deeper in the project could be risky and expensive. Thus, can lead to a solution based on the firmware modification. Using the right Task types, the adopted solution could be less efficient, but it will prevent the risk of high costs and time.

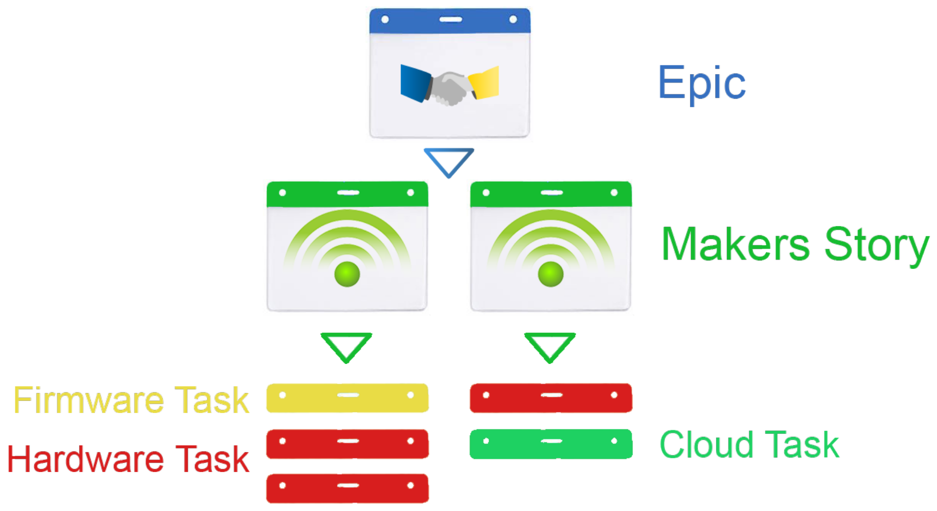


Figure 9 - Epics, Maker Stories and Tasks

5.2 Fiotto Team

Fiotto Teams embrace the **T-Shape** philosophy: team members often have vertical (*Deep*) skills in a certain area, but they can support the rest of the team on every activities (*Broad*).

The full team split themselves in the **Makers Team** (MT), a temporary team, engaged on a single Makers Story. The MT can be *Cloud oriented* and *Smart Thing oriented*, depending on the activity, and it is always composed by two makers with different skills. This is a new way of looking at the pair-programing (and not just programming) practice, where Makers Team becomes the “basic unit” for the whole processes.

5.3 Areas

Fiotto is separated in 7 main areas (columns):

- **Product Backlog**, contains the Epics of the product, sorted by the relative Value;
- **Makers Backlog**, contains the Makers Stories and the related Tasks, which are part of the Epics (the ones which are ready to be implemented). In this area, the WIP Limit is the same of the number of the MT. Thus, the MT, will focus on just one goal at a time. This limit can be changed depending on the needs;

- **D-ARCH**, is the working area for the Tasks. It will be discussed later;
- **Done**, when a Makers Story reach this area, the related Tasks are completed and it is time to prepare the deployment tasks. For example, Customer Service and Operations stuff can be enabled, to support the new production items;
- **Ready to Deploy**, the Makers Stories here should be deployed. For example, the deployment date and time should be defined;
- **Deployed**, The Maker Story is deployed actually;
- **Value Achieved**, when an Epics reach this area, the related Maker Stories are deployed and the solution is working. All the *Solution Definition of Done* (sDOD) criteria must be satisfied.

5.4 WorkPivot

WorkPivot allows to pass from the overall Tasks of the solution, which interest the whole Fiotto Team, to the single Makers Team ones.

Fiotto allows us to move gracefully between the two visions, and this is due to the D-ARCH introduction.

5.5 D-ARCH

The **D-ARCH** is the beating heart of Fiotto. It points out the "pull" items that give the rhythm. D-ARCH is an acronym of **Do it for… ACHIEVE RAPIDLY CUSTOMER HOPES**.

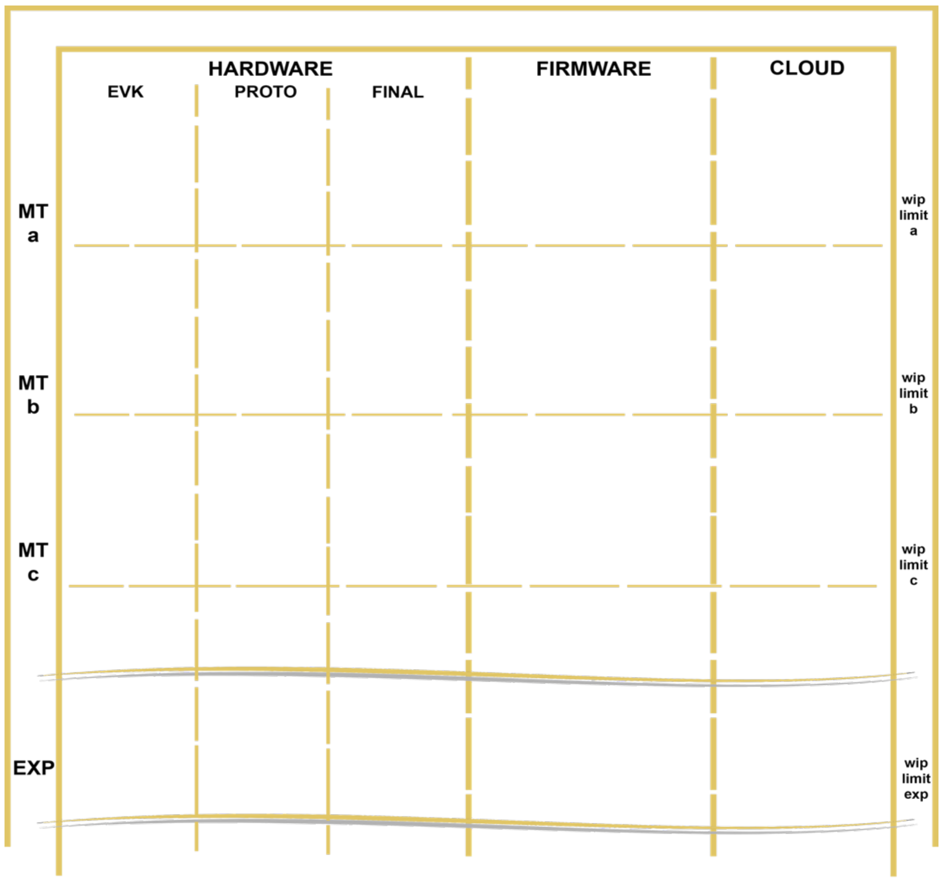


Figure 10 - D-ARCH

The D-ARCH reverses the overview (using *WorkPivot*) moving the focus on the tasks of the Makers Team. It's separated in *Vertical* and *Horizontal* dimensions:

- *Vertical Dimension:* the aspects of the Fiotto Team and the overall solution;
- *Horizontal Dimension:* the aspects of each Makers Team.

The pillars of the arc identify the *Makers Team* and the related *WIP limit*, while the rest of the canvas is separated in columns which identify the *Task* type: **hardware**, **firmware** and **cloud**.

The Hardware Task Area is separated in: **EVK**, **Smart Thing Proto** and **Smart Thing Final**. This highlights the different phases when implementing the hardware related to the Smart Thing.

Next to the Makers Team sub-areas, there is another sub-section for the **Experiments (EXP)**, where it is possible to add Tasks for verifying assumptions, risk assessment, know-how needs and so on.

Notice that the EXP sub-section, delimited by two swim lanes, is a horizontal dimension, not related to the MT. Thus, an **Experimental Team** (ET) establishment is encouraged by the framework itself. The ET should be formed by a couple of makers, to share the know-how derived by the experiments.

5.6 D-ARCH Metrics

An interesting aspect of the **D-ARCH** is the ability to associate different metrics to both the horizontal and vertical dimensions. This allows to gather important information when making decisions about the *Continuous Improvement* of the Fiotto Team and the project management itself.

The principal metrics are depicted in the following picture:

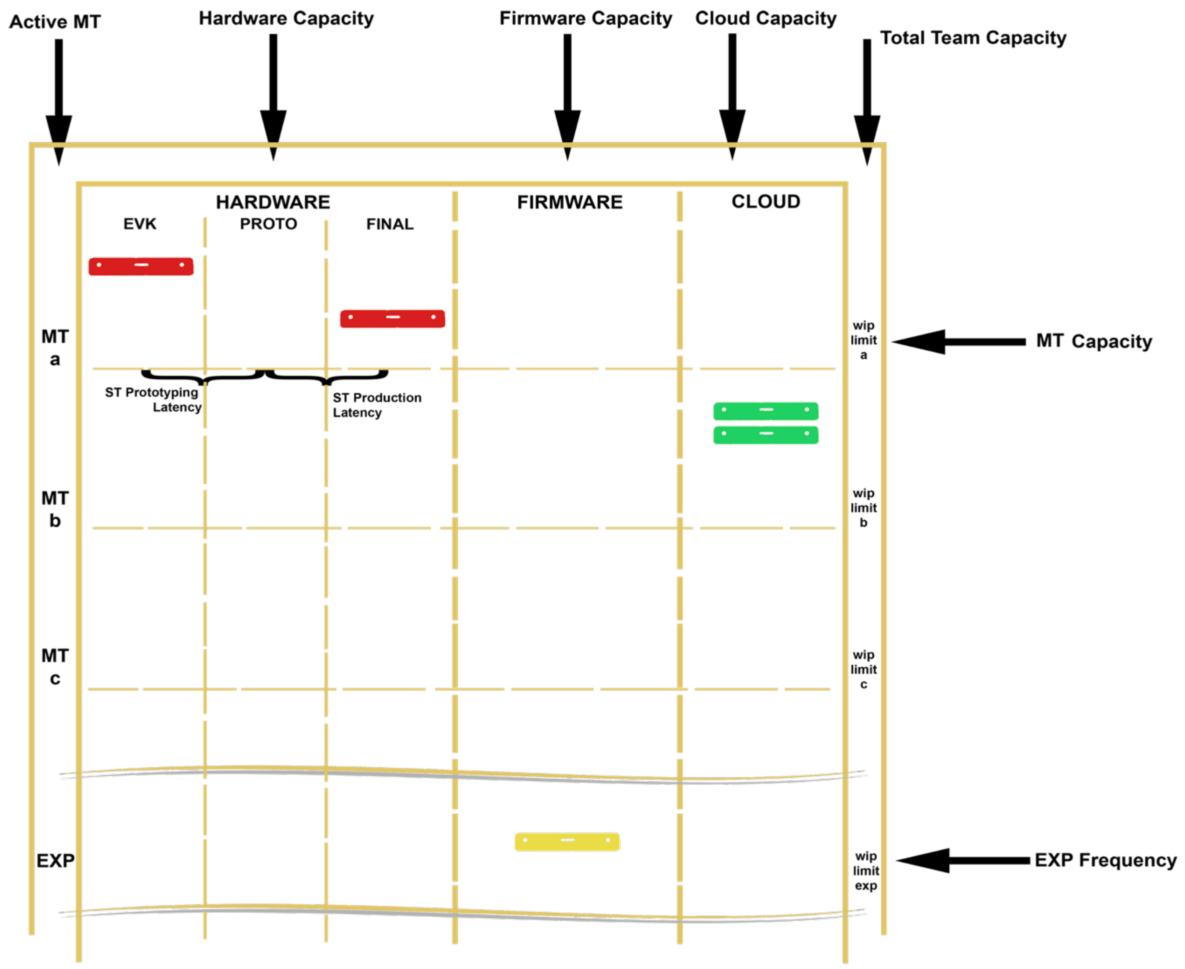


Figure 11 - D-ARCH Metrics

Going deeper, metrics can be categorized depending on the related dimension:

- *Horizontal Dimension, related to the Makers Teams:*

    - **MT Capacity**: ability to achieve the goals for the MT;
    - **EXP Frequency**: frequency for experimenting for the MT/ET team.
- *Vertical Dimension, related to the overall Fiotto Team:*
    - **Total Team Capacity**: ability to achieve the goals for the Fiotto Team;
    - **Hardware, Firmware and Cloud Capacity:** overall capacity of the Fiotto Team for achieving the goal related to the IoT aspects. The sum of the three metrics is the **Team Skills**.
    - **Balanced View**, that is the measure of the Fiotto Team overall skill balance;
    - **Active MT**: number of the active MT on the project;
    - **ST (Smart Thing) Prototyping Latency**: latency between the EVK usage to the custom Smart Thing prototype;
    - **ST (Smart Thing) Production Latency**: latency between the custom Smart Thing prototype and the final version.

Speaking about *Capacity* and *Frequency*, the unit is the Task.

Focus on the "latency" metrics, S*T Prototyping Latency* and *ST Production Latency*: both, with the related card and the start and completed time, allows to point on the gap between EVK usage and the custom Smart Thing implementation.

These informations, allows to manage at the best the definition of the *Bill of Materials* (BOM) and the production, which is commonly an external activity.

6 Conclusions

No work is perfect, especially in the world of software and hardware development, where there is always time for improvement and evolution.

Agile and Lean have taught us the *inspect-and-adapt* mantra, pushing us to improve continuously. AgileIoT Duttile and AgileIoT Fiotto are not an exception, and they are constantly evolving, thanks to the direct experimentation and your valuable feedback.

Author

Felice Pescatore (@felicepescatore) deals with Agile at enterprise level, helping companies to make the necessary Cultural changes to make them more responsive and ready to the modern market challenges.

Co-Authors

I'd like to thank for their patience and support: **Alessandro Alpi** (@suxstellino), **Matteo Emili** (@MattVSTS)**, Giuseppe Famiglietti** (@giufamiglietti), **Paolo Patierno** (@ppatierno), **Gian Maria Ricci** (@alkampfer), **Igor Antonacci** (@bitbreaker) and **Vincent Thavonekham** (@vincentthavo).

Thanks to **Andrea Palumbo** (@ndrea_palumbo) for the support in analyzing legal aspects, specially related to the privacy.