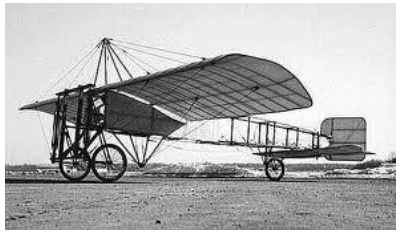
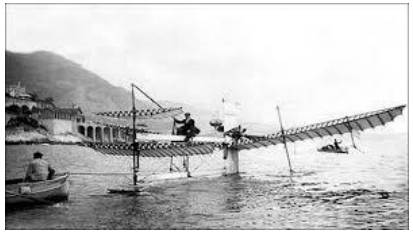
CentraleSupélec

# Elements of Complex System Engineering

Antoine B. Rauzy
Department of Mechanical and Production Engineering (MTP)
Norwegian Science and Technology University (NTNU)
and
Chaire Blériot-Fabre
Centrale-Supélec, SAFRAN


CentraleSupélec

# LECTURE 8.
# STOCHASTIC SIMULATION

Notions:

- Monte-Carlo Simulation
- Cellular Automata

CentraleSupélec

# LECTURE 8. PART 1. INTRODUCTION

CentraleSupélec

# Objective of this lecture

In many system operation situations, we face an **uncertain environment** and nevertheless we have to **make decisions**. A priori, this seems impossible: how to make a decision if we don't know what will happen?

In many cases however, we have at hand some **statistical data** about the system under study and its environment. We can use these data to design **stochastic models**, i.e. models in which **the system evolves at random**. This randomness is however controlled in two ways: first, trajectories (runs) of the system are **governed by the statistical data**; second they are **reproducible** – they are **pseudo-random evolutions**.

This type of models is widely used in many different areas: from epidemiologic studies to financial markets going through insurances, nuclear safety…

The objective of this course is to introduce **Monte-Carlo simulation** which is a fundamental tool of systems engineering.

# Case Study: Prevention of Forest Fires

Forest fires destroy regularly the most beautiful Mediterranean landscapes, especially during summer where the vegetation is dried.

To prevent such devastations, material and human means must be put in place. But to be efficient, we must understand how fires propagate.

The problem is that propagation of a forest fire is by no means deterministic. It depends (among other things):

- Of the configuration of the ground, in particular of the density of the vegetation;
- Of the wind force and direction.

We cannot know in advance what they will be. But we have at hand statistical records.

# What to Do?

The idea is to split the considered region into small square zones. The dimension of these zones is chosen so to be able to characterize the content of each zone: vegetation, rock, house , water (lake or sea), road…

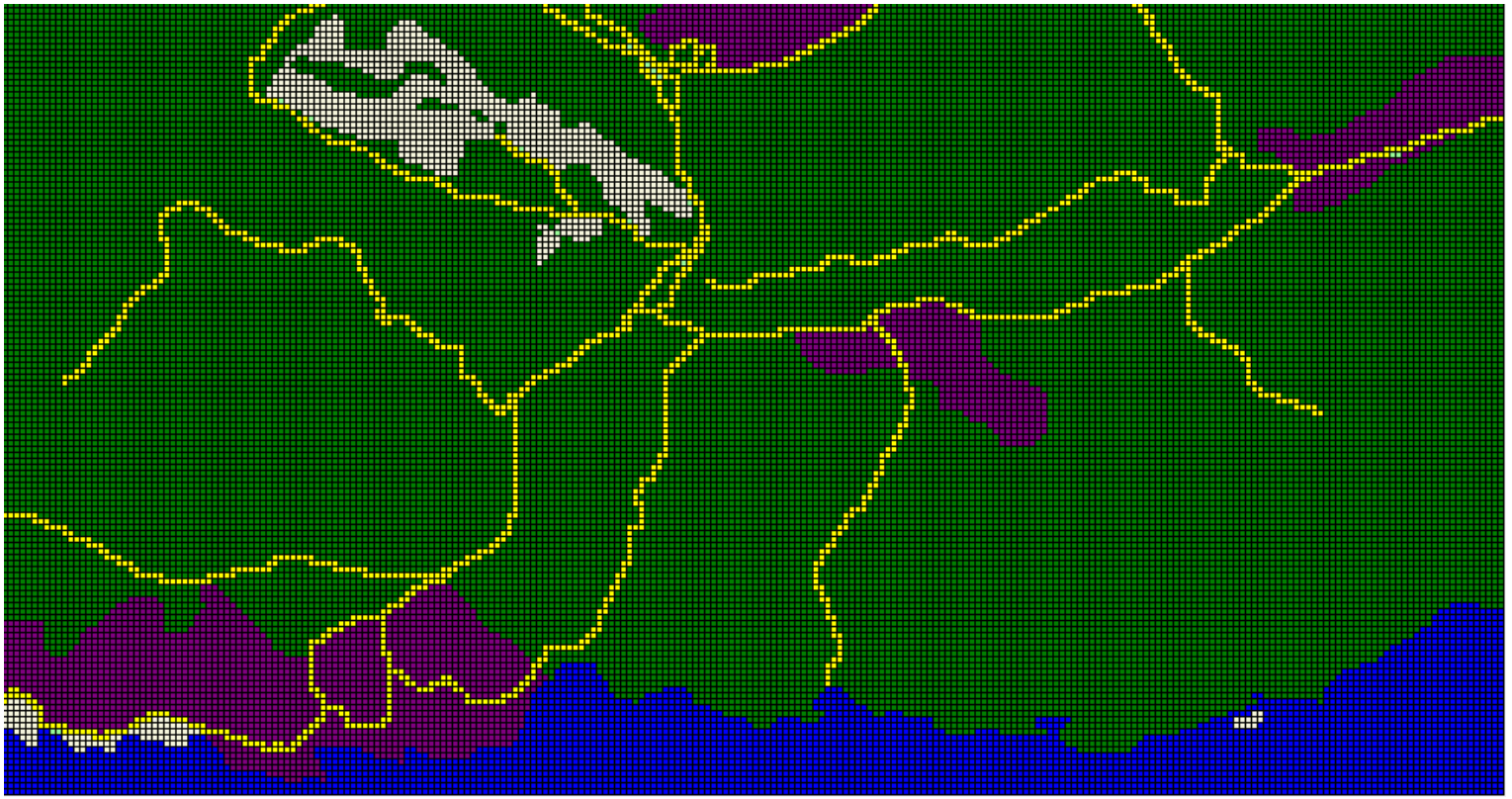
CentraleSupélec

# What to Do?

The idea is to split the considered region into small square zones. The dimension of these zones is chosen so to be able to characterize the content of each zone: vegetation, rock, habitation, water (lake or sea), road…

250

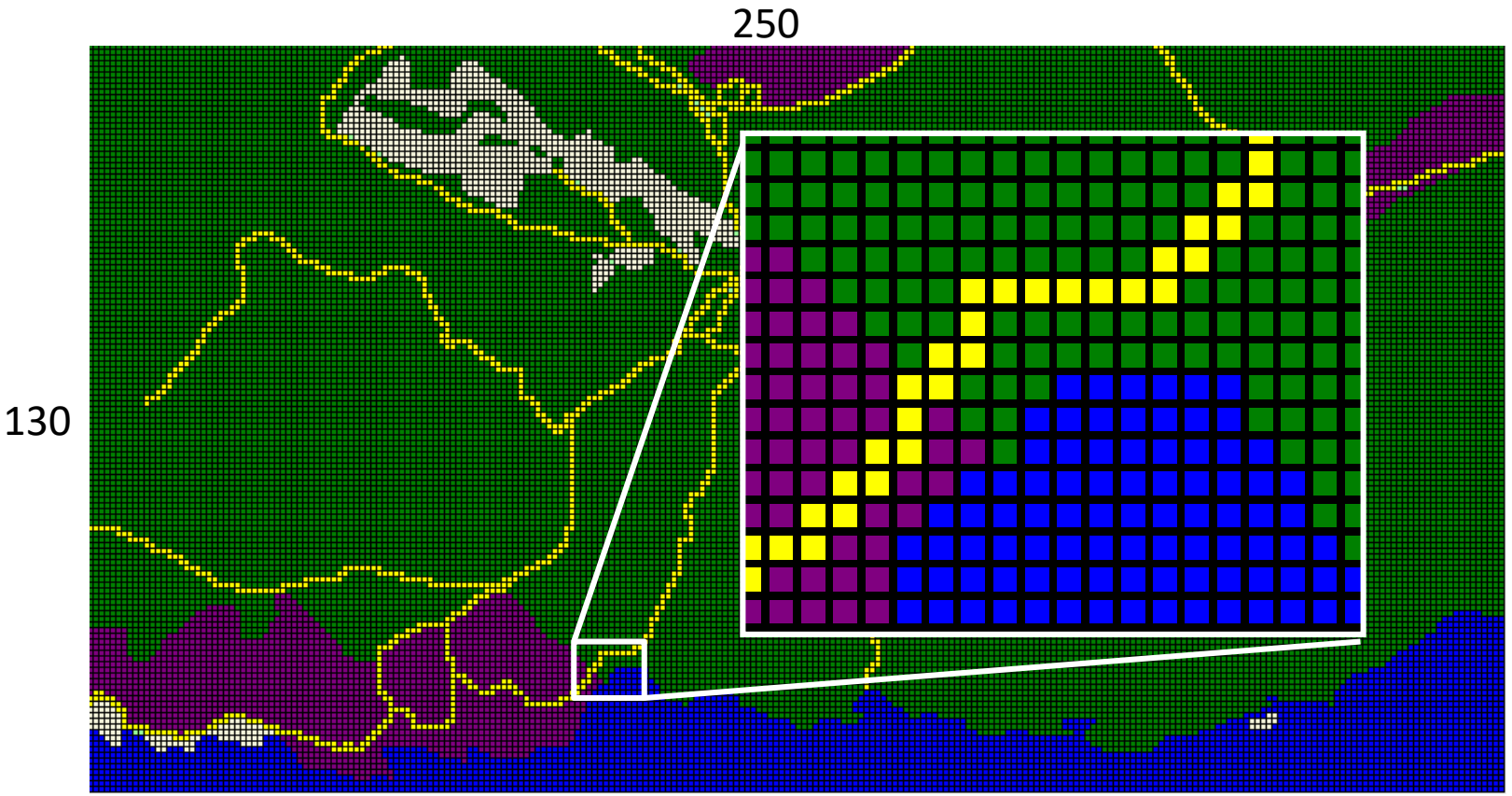
130



CentraleSupélec

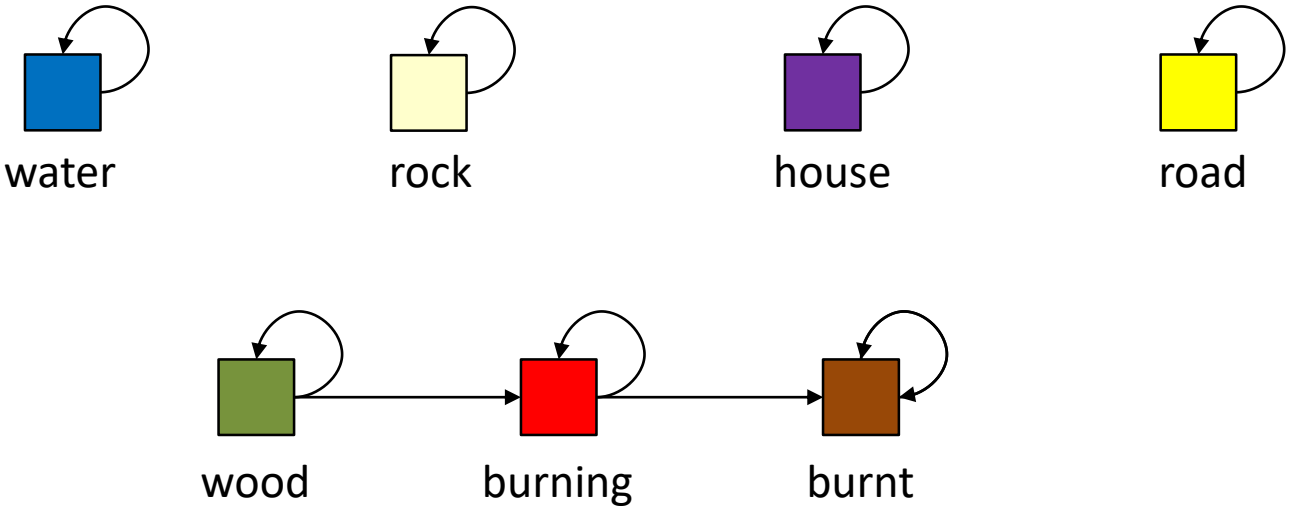
# What to Do?

The idea is to split the considered region into small square zones. The dimension of these zones is chosen so to be able to characterize the content of each zone: vegetation, rock, house, water (lake or sea), road…



# Cellular Stochastic Automata

The first step consists in building a stochastic model for the fire propagation. This stochastic model describes how each cell evolves in parallel with other cells depending on its content and the state of surrounding cells. This is the framework of **stochastic cellular automata**.




In our case study, evolutions of zones “wood” and “burning” are stochastic. Probabilities of transitions depend on the wind force and direction and the state of surrounding cells.

# Cellular Stochastic Automata

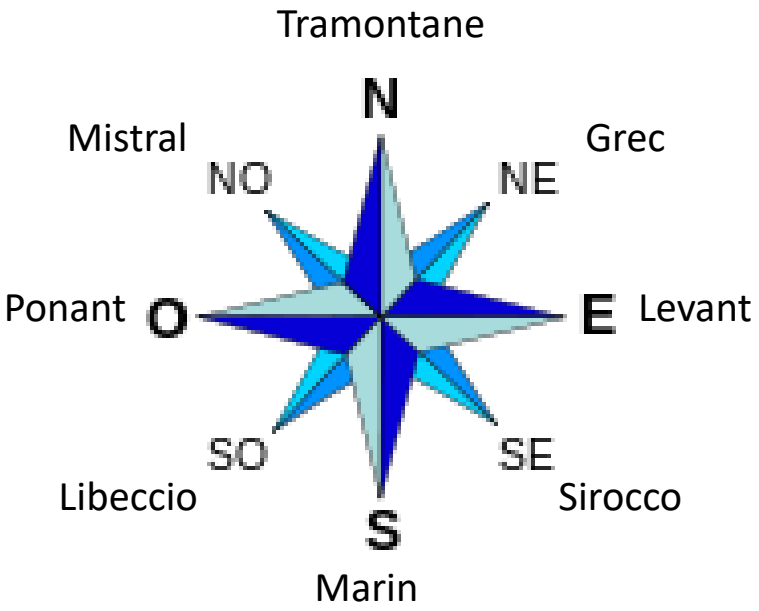
It is the possible use statistical data to describe how the fire propagates from one vegetation zone to an adjacent vegetation zone depending on the conditions.

| Wind | light (0-10 knots) | moderate (10-20 knots) | strong (> 20 knots) |
|---|---|---|---|
| West → | 0.08<br>0.08<br>0.08 | 0.10 0.06<br>0.10 0.06<br>0.10 0.06 | 0.12 0.08 0.04<br>0.12 0.08 0.04<br>0.12 0.08 0.04 |
| South-West ↗ | 0.08 0.08<br>0.08 | 006 0.06<br>0.10 0.10 0.06<br>0.10 | 0.04 0.04<br>0.08 0.08 0.04<br>0.12 0.12 0.08<br>0.12 |

The probability that a zone switches from state “wood” to state “burning” is the sum of the probabilities over its neighboring zones that of these zones propagates the fire.

# Statistical Data on Winds

Météo-France provides the following statistical data on winds in Marseilles region on the period June-August (these data have been slightly simplified).




| | 0-10 knots | 10-20 knots | 20-40 knots |
|---|---|---|---|
| N | | | 5% |
| NE | | | |
| E | | | |
| SE | 5% | 5% | |
| S | 10% | 5% | |
| SO | | 5% | |
| O | 10% | 15% | |
| NO | 5% | 20% | 15% |

In our model, we shall assume that the wind is changing periodically according to the above probability distribution.

CentraleSupélec

# Simulation(s)

It is then possible to simulate a fire outbreak. The location of this outbreak is chosen at random on the map. It propagates depending on ground configuration and wind force and direction.
In practice, firemen are overloaded when several fire outbreaks occur simultaneously. These simultaneous fire outbreaks are often created by pyromaniacs.

It is clear however that performing a small number of simulations by hand is not sufficient to get a good global picture of where to position men and materials.

CentraleSupélec

# LECTURE 8. PART 2.
# STOCHASTIC DISCRETE EVENT SYSTEMS

CentraleSupélec

# Discrete Event Systems

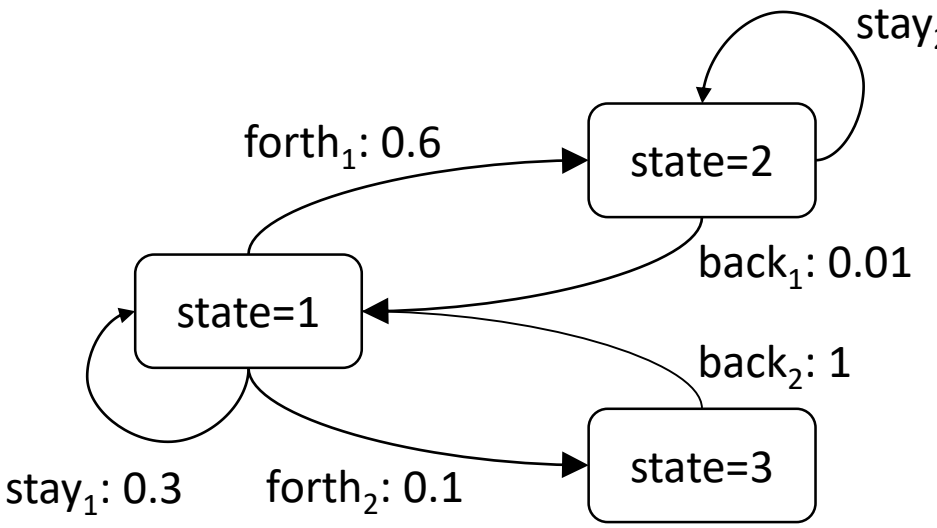
A **Discrete Event System** (DES) is 4-tuple $\langle V, E, T, i \rangle$ where :

- V is a set of **variables** describing the **state** of the system. In general, variables of V take their values into **finite domains**;
- E is a set of **events**;
- T is a set of **transitions**. A transition is a triple $\langle G, e, A \rangle$, denoted as $e: G \rightarrow A$, where:
  - G is a Boolean function over V, i.e. a mechanism that given a value of the variables returns true or false. G is called the **guard** of the transition.
  - e is an event of E.
  - A is a function from V to V, i.e. a mechanism that given a value of variables returns a new value for these variables. A is called the **action** of the transition.
- i is the **initial assignment** of variables.

A transition $e: G \rightarrow A$ is **fireable** in the state $\sigma$, i.e. for the variable assignment $\sigma$, if $G(\sigma)$ = true.

# Discrete Time Stochastic Transitions

There are two ways to introduce “randomness” in Discrete Event Systems. The first one consists in adding probabilities to the untimed model, leading to (implicit) **Discrete Time Markov Chains**.

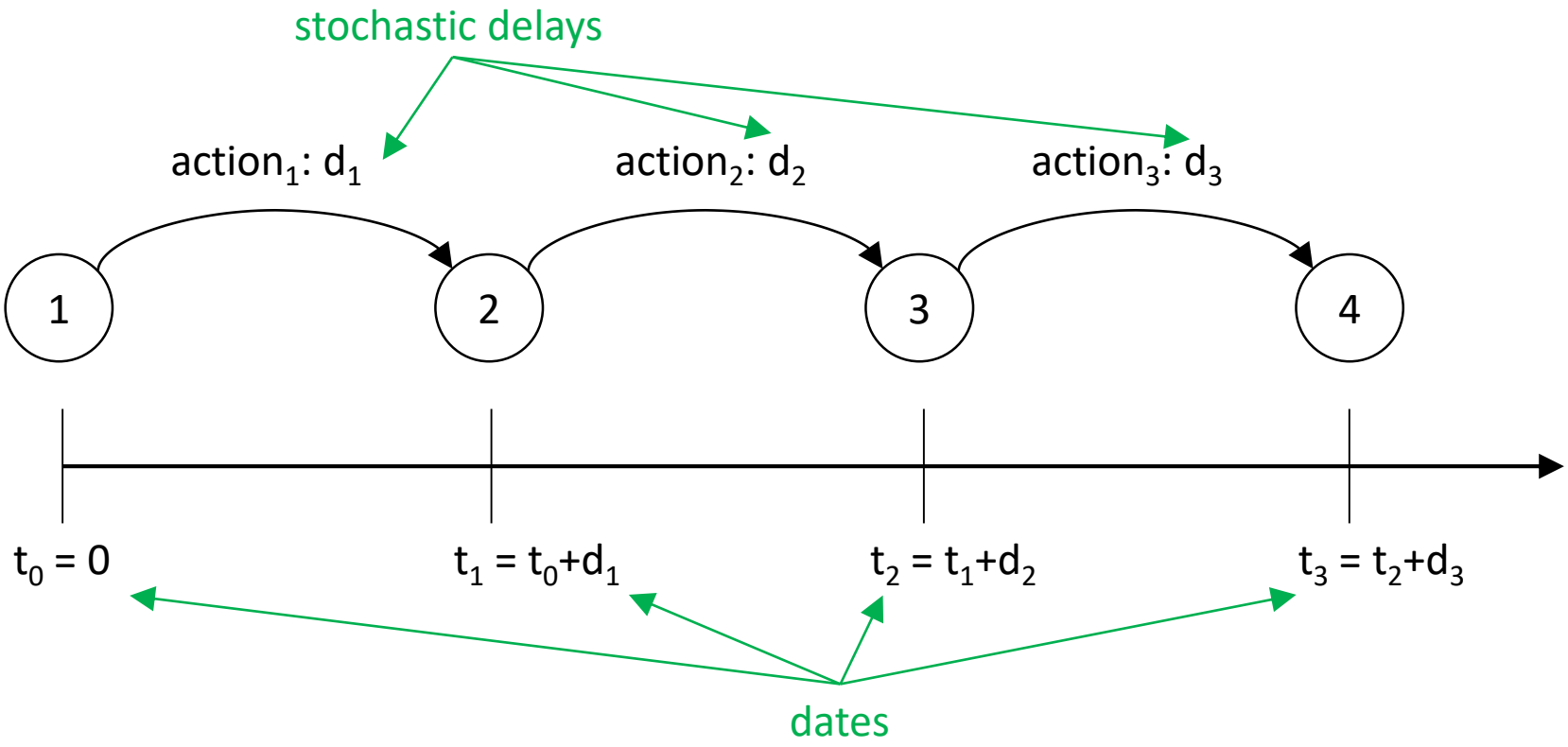



- In state 1, the probabilities of firing the transitions forth$_1$, forth$_2$ and stay$_1$ are respectively 0.6, 0.1 and 0.3.
- In state 2, the probabilities of firing the transitions back$_1$ and stay$_2$ are respectively 0.01 and 0.99.
- In state 3, the probability of firing the transition back$_2$ is 1.

The evolution of the system is then seen as a sequence of transitions. The probability to be in state s at step n is the sum of the probability of paths leading from state 1 to state s in exactly n steps.

**Our cellular automata simulator is based on this model.**

# Continuous Time Stochastic Transitions



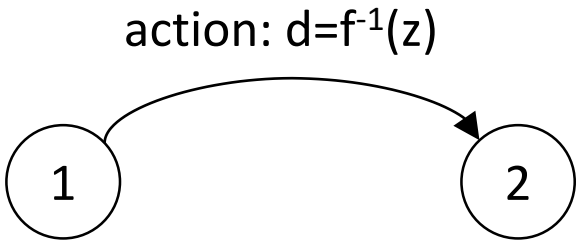
# Timed Runs

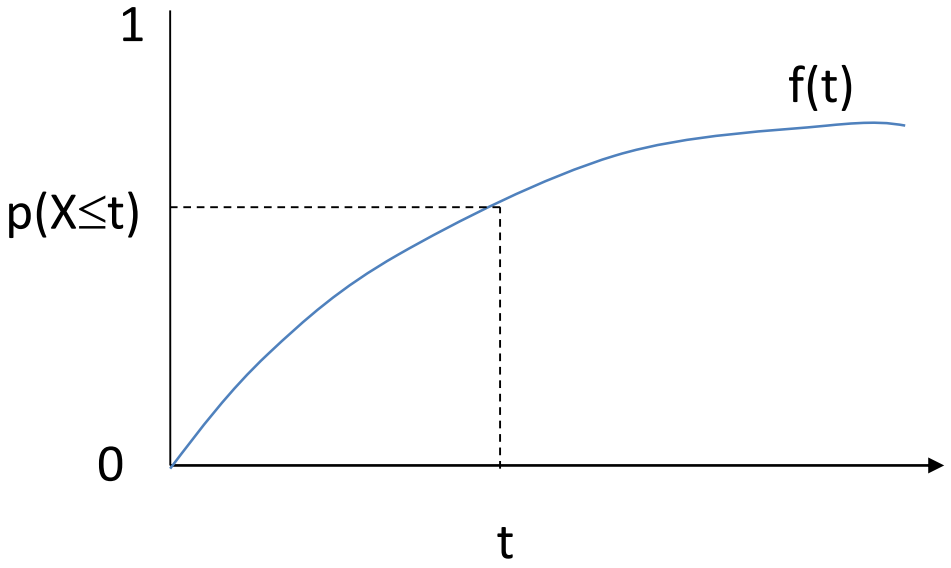
Let M: $\langle V, E, T, \iota \rangle$ be a timed discrete event system. A **timed run** of M is a finite alternated sequence $\langle(\sigma_{0,}, s_0), (e_{1,},d_1), (\sigma_1, s_1), (e_{2,},d_2),\ldots (\sigma_{n,}, s_n)\rangle$ where the $\sigma_i$'s are variable assignments, the $s_i$'s are schedules, the $e_i$'s are events and the $d_i$'s are delays.

The set $R_M$ of runs of M is the smallest set of runs such that:

- $\langle \iota, s_0\rangle$ belongs to $R_M$. It is the only empty run. It starts and ends in $\iota$. s0 is the schedule made of fireable transitions at t=0.
- If $\langle(\iota, s_0),\ldots , (\sigma_{n,}, s_n)\rangle$ is a run of $R_M$ ending by the state (variable assignment) $\sigma$ with the schedule s, and e: G $\rightarrow$ A of T is a transition of delay d and such that G($\sigma$) is true then $\langle(\iota, s_0),\ldots, (\sigma_{n,}, s_n), (e_{,}d), (\sigma_{n+1,}, s_{n+1})\rangle$ is a run of $R_M$, if:
  - $\sigma_{n+1} = A(\sigma_n)$
  - $s_{n+1}$ is the schedule obtained by updating $s_n$ with the firing of e: G $\rightarrow$ A

# Cumulative Probability Function




A **cumulative probability function** f is a monotone increasing function from positive real numbers to [0, 1]. It associates with each date t, the probability that the transition is fired before t. Its inverse can $f^{-1}$ can be used to generate a delay at random.

CentraleSupélec

# LECTURE 8. PART 3.
# STOCHASTIC SIMULATION: PRINCIPLE

# Monte-Carlo Simulation

Whether with the discrete time model or with the continuous time model, it is in general not possible to calculate the exact probability to be in state s at time (or step) t because it is not possible to build the underlying reachability graph.
The idea is thus to draw at random a number of runs and to make **statistics** on various **indicators** by observing these runs. This is the principle of **Monte-Carlo simulation**.

E.g. here follows the results of 1000 simulations (a fire is extinguished when there is no more “burning” zone):

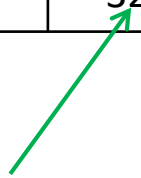
| Indicator | mean | standard-deviation | 95% confidence range |
|---|---|---|---|
| Number of steps before fire extinction | 142.19 | 169.82 | [131.66, 152.72] |
| Number of burnt zones | 625.62 | 1109.91 | [556.83, 694.41] |
| Number of houses threatened | 3.05 | 8.35 | [2.53, 3.57] |

# Quantiles

The standard-deviation and confidence ranges are not always very informative. To have a more precise idea about indicators, it is possible to calculate **distributions** and **quantiles**. E.g.

Quantiles of the number of steps before fire extinction

| | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|
| minimum | 2 | 2 | 4 | 10 | 32 | 75 | 127 | 191 | 278 | 388 |
| mean | 2.00 | 2.89 | 6.71 | 19.34 | 52.01 | 99.53 | 155.09 | 230.57 | 327.85 | 525.90 |
| maximum | 2 | 4 | 10 | 32 | 74 | 127 | 191 | 278 | 386 | 880 |



30% of the runs terminated in less than 10 steps
40% of the runs terminated in less than 32 steps

# What conclusion can we draw?

Our program is too simple to draw any valuable practical conclusion. To get realistic results, we would have to take into account the type and density of the vegetation, the topography of the region… and above all to check that the model describes in an accurate way the reality by matching its results with observed fire propagations. However, this program makes it possible to understand which conclusions could be drawn from such a study:

- Understand fire propagation so to position material and human means at the right place;
- Prevent fire propagation by selective cuts and/or by forbidding the access to especially sensitive zones;
- Allocate the right budget to the intervention forces;
- Adjust regulation regarding brushing;
- Define insurance mechanisms;
- …

CentraleSupélec

# Algorithm

The generic **Monte-Carlo simulation algorithm** is as follows.

```
StochasticSimulation ( numberOfHistories ):
    InitializeIndicators()
    for t in 1.. numberOfHistories:
        RunHistory()
        UpdateIndicators()
    MakeStatistics()
```

It is thus quite simple to implement. However, you must take care at:

- Selecting the right **probability distributions** and the right **performance indicators**;
- Performing **sufficiently many runs** (a **sufficiently large sample size**) regarding the probability of the **observed phenomenon**;
- Selecting a good **pseudo-random number generator**;
- To verify that the **model** meets the **reality.**

# When to use Monte-Carlo simulation?

Monte-Carlo simulation has several advantages over other methods:

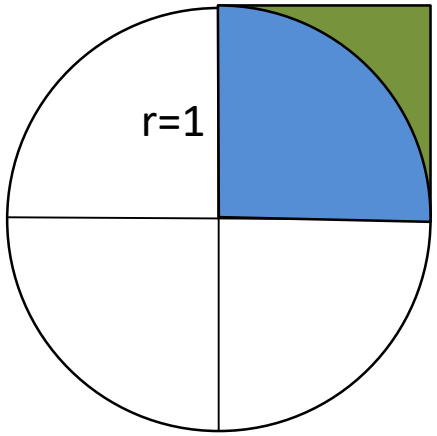
- It is much **cheaper** to realize that any physical experiment. Moreover it is **easy to reproduce**;
- It makes it possible to study phenomena for which **no analytical solution** exists or for which such a solution is much too hard to establish;
- It is an "**anytime**" algorithm: to more you draw runs, the better the result, but you can stop the simulation any time.

However, Monte-Carlo simulation is not an universal panacea:

- It may be very (too) **calculation resource consuming**;
- It gives only **approximated solutions**;
- But more fundamentally, it is often extremely difficult to **check models**;

# Estimation of $\pi$=3,141 592 653 589 793

In theory, Monte-Carlo simulation can be used to estimate $\pi$.




Idea: draw a large number N of pairs (x, y) uniformly in [0, 1]x[0, 1], and count the number K of pairs such that $x^2 + y^2 \leq 1$.
Then, $\pi \approx 4.K/N$

In practice however, only the first decimal of $\pi$ can be reached.

| N | K | $\pi$ |
|---|---|---|
| 1 000 000 | 784 802 | 3.139208 |
| 10 000 000 | 7 852 310 | 3.140924 |
| 50 000 000 | 39 264 640 | 3.141171 |
| 100 000 000 | 78 532 441 | 3.141298 |

… but it may used to test your random number generator…

CentraleSupélec

# Integral Calculation

Assume we have to calculate a complex integral in the form:

$$\int_a^b f(x)dx$$

The value of this integral can be calculated as (b-a) times the mean value of the function f in the interval [a, b].

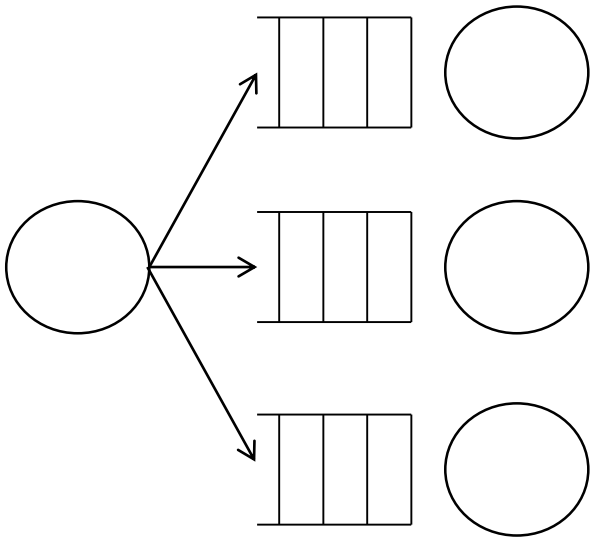
We can estimate this mean value by means of a **Monte-Carlo simulation**: we draw a random n points in the range [a, b] and calculate the value of f in each of this point and then calculate their empirical mean.

This method works in any dimension. The convergence speed is the same, whatever the number of dimensions, namely $\frac{1}{\sqrt{n}}$.

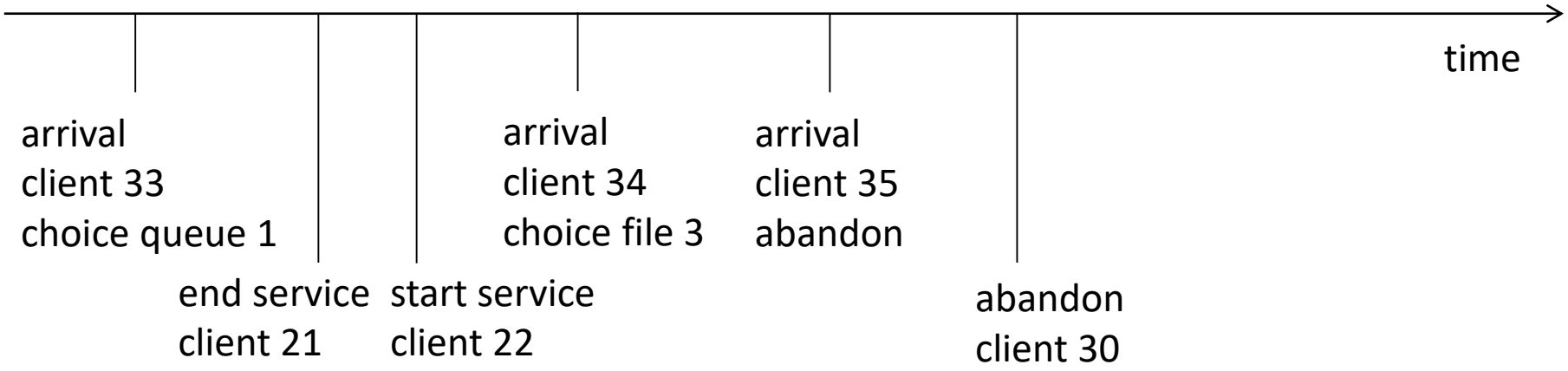
This method has been applied for the first time during the Manhattan project. It is notably used for the assessment of option prices in finance (« call » and « put »).

CentraleSupélec

# Queues



| Event | Delay & choice |
|---|---|
| Arrival of a clients | exponential |
| Choice of a queue or abandon | immediate & probability |
| Start client service | immediate |
| End client service | normal truncated |
| Abandon of a client | exponential |
| … | … |



CentraleSupélec

# LECTURE 8. PART 3.
# STOCHASTIC SIMULATION: TECHNICAL ISSUES

# Key Issues for Monte-Carlo Simulation

It is thus quite simple to implement. However, you must take care at:

- Selecting the right **probability distributions** and the right **performance indicators**;
- Performing **sufficiently many runs** (a **sufficiently large sample size**) regarding the probability of the **observed phenomenon**;
- Selecting a good **pseudo-random number generator**;
- To verify that the **model** meets the **reality.**

# Estimators

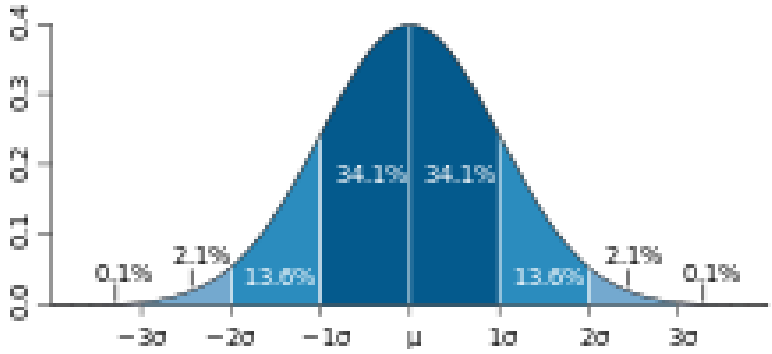
Three **estimators** are in general calculated for each indicator: its mean, its standard-deviation and a confidence range (in general 95% confidence range).

- The **mean**: $\mu_X = \frac{\sum_n x_i}{n}$

- The **standard deviation** is a measure of dispersion of data (around the mean).

  Formal definition: $\sigma_X = \sqrt{E[X^2] - E[X]^2}$

  Usual calculation method: Welford algorithm.




- The p% **confidence range** is the interval centered on the mean such that the actual value of the indicator has a probability p(/100) to be in the interval. For p=95, its definition is has follows:

$$I_{95\%} = \left]\mu_X - 1.96\frac{\sigma_X}{\sqrt{n}}, \mu_X + 1.96\frac{\sigma_X}{\sqrt{n}}\right[$$

Tables give the values of the parameter (here 1.96) for different values of p.

CentraleSupélec

# Quantiles

The standard-deviation and the confidence range(s) are of interest only if the indicator is distributed according to a normal law. In practice, this assumption is seldom verified. To get more insights about indicators, we can estimate their **distributions** and their **quantiles**.

The principle is to sort the observed values and to organize them into **bins** of same size. For each bin, it is then possible to calculate a minimum, a mean and a maximum value. The maximum value is the **quantile**. The value M such that 50% of observed values are smaller than M and 50% bigger is called the **median**.

| | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|
| minimum | 2 | 2 | 4 | 10 | 32 | 75 | 127 | 191 | 278 | 388 |
| moyenne | 2.00 | 2.89 | 6.71 | 19.34 | 52.01 | 99.53 | 155.09 | 230.57 | 327.85 | 525.90 |
| maximum | 2 | 4 | 10 | 32 | 74 | 127 | 191 | 278 | 386 | 880 |


(*) In 2011, the mean of monthly salaries in France was 2172€, while the median was 1712€

CentraleSupélec

# Size of Samples

It is well know that is not necessary to consider too large samples (number of runs). This is justified by the definition of confidence ranges:

$$I_\alpha = \left]\mu_X - t_\alpha \frac{\sigma_X}{\sqrt{n}}, \mu_X + t_\alpha \frac{\sigma_X}{\sqrt{n}}\right[$$

This definition says that to reduce the uncertainty by a factor k, one has to increase the sample size by a factor $k^2$. However,

1) This applies only in case where the indicator is distributed according to a normal law;
2) This applies only if the sample size is big enough.

In practice, if we want to observer a phenomenon that has 1 chance out of 1 million to occur, we need to draw a least 100 millions runs to get some valuable information.

This high number of runs is a limiting factor of the Monte-Carlo simulation.

Here is a general rule of experimental sciences: to observe a phenomenon, we need to have a a priori idea of its **nature** and its **amplitude**.

CentraleSupélec

# Random number generators

Stochastic simulation relies on the generation of sequences of (pseudo-)random numbers. From a mathematical standpoint, a random sequence is an infinite sequence of symbols of an alphabet (typically {0, 1}) that has no structure, regularity or identifiable prediction rule. It is actually quite difficult to give a formal definition of random sequences and several alternative definitions have been proposed.

In practice, the problem is to find **pseudo-random number generator** that "mimics" as much as possible the "real" randomness. There exists two main categories of generators:

- Generators relying on physical devices. They have the double drawback of being hardly testable and non reproducible.
- Algorithmic generators are used most of the time, possibly by initializing them with physical devices (like the computer clock).

# Mersenne-Twister Generator

The **Mersenne-Twister** generator is known for its quality.

- It has a $2^{19937}$-1 period;
- It is uniformly distributed even in a large number of dimensions (623 for 32 number bits) ;
- It is faster than most of the “good” generators;
- It passes the most advanced statistical tests on each bit.

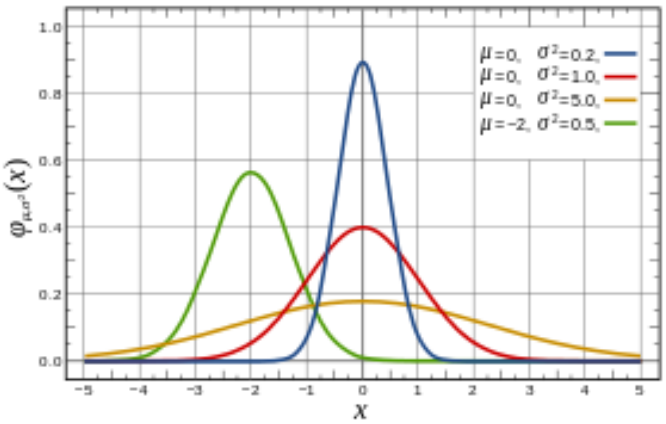
It is however too complex to be described here (those who are interested can have a look to: http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html.

That’s the generator used in Python, Matlab, Ruby, R, gnu...
It is used for many application, including cryptographic applications.

# Some Classical Distributions (1)

Pseudo-random number generators give sequences of integers uniformly distributed in the range [0, $2^{31}$-1] (on 32 bits machines) and [0, $2^{63}$-1] (on 64 bits machines). To obtain a floating point number uniformly distributed between 0.0 and 1.0, it suffices to divide the generated number by $2^{31}$-1 (or $2^{63}$-1).
Beyond the uniform distribution, several other distribution are often used. Two are especially important: the normal distribution and the exponential distribution.

The **normal distribution** is one of the most convenient to represent phenomena issued from several random sources. It is defined by means of two parameters: its means and its standard-deviation (or equivalently its variance).
There exist relatively simple methods to generate a normal distribution from an uniform distribution (e.g. Box-Muller method, Marsaglia and Bray method).

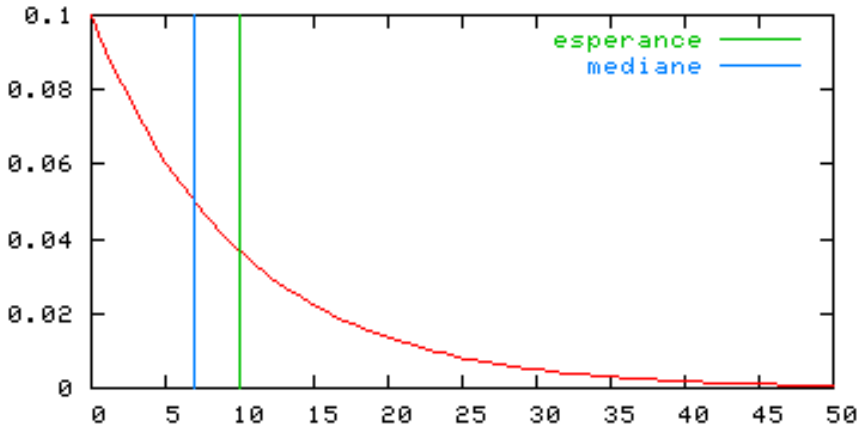


# Some Classical Distributions (2)

The **exponential distribution** represent typically the life-span of a component without memory, aging nor wearing (**Markovian hypothesis**). The probability that the component is working at least t + d hours knowing that it worked already t hours is the same as the probability that it works d hours after its entry into service. In other words, the fact that the component worked correctly for t hours does not change its expected life duration after this delay.

The exponential distribution is defined by means of a single parameter, the **transition rate**, usually denoted by $\lambda$. This transition rate is the inverse of the mean life expectation. The probability density of this distribution is:

$$F(t) = 1 - e^{-\lambda t}$$




To draw a delay d according to an exponential distribution of parameter $\lambda$, it suffices to draw a number z uniformly in the range [0, 1[ and to calculate:

$$F^{-1}(z) = -\frac{1}{\lambda} \times \log(1 - z)$$

CentraleSupélec

# LECTURE 8. PART 5.
# WRAP-UP & ASSIGNMENT

CentraleSupélec

# Wrap-Up

- **Monte-Carlo simulation** is one of the fundamental tools of complex systems engineering. It makes it possible to study problems for which **no analytical solution** exists or analytical solutions are too difficult to find.
- Monte-Carlo simulation is easy to implement. It is a **anytime algorithm**: the more runs are simulated the better the result, but the algorithm can be stopped any time.
- However there are two limits to the application of Monte-Carlo simulation:
  - First, the **size of the sample** (number of runs) necessary to get accurate results. To measure a phenomenon, you have to have an idea, even imprecise of its nature and its amplitude;
  - Second and more importantly, stochastic models are **hard to verify**.
- Monte-Carlo simulation implements a randomness which is controlled in two ways:
  - First, it relies on **statistical data** that must be available a priori.
  - Second, runs a perfectly **reproducible**, thanks to the use of **algorithmic random number generators**.

CentraleSupélec

# Assignment

The “Premier League”, the English football championship is a huge business. Hundreds of millions of pounds are invested every year to get the build up the best teams by hiring the best players, the best trainers... Does investing huge amounts of money warranty a success? More exactly, what does result from the talent of teams and what is aleatory? A football game is by no means deterministic. Even the best player in the world cannot control precisely the trajectory of a ball.

Let’s study that!

The idea is to do a post-mortem analysis a past season. We shall introduce some small random variation in the results of the game (one additional goal scored here, one less goal scored there) and see whether the top 5 is robust to these small variations.

To do so, you will have to adjust the program “myleague.py”

# Recommend Readings

**Recommended books on discrete event systems:**

Christos G. Cassandras and Stéphane Lafortune. *Introduction to Discrete Event Systems*. Springer US, 2008. 978-0-387-33332-8

**Recommended books on cellular automata:**

Stephen Wolfram. *A New Kind of Science.* Wolfram Media, 2002, ISBN 1-57955-008-8

**Recommended books on randomness:**

Gregory Chaitin. *Exploring Randomness (Discrete Mathematics and Theoretical Computer Science).* Springer London, 2001, ISBN 978-1-85233-417-8

CentraleSupélec

**Louis Charles Joseph Blériot** (1872 -1936) is an airplane designer and one of the pioneer pilot of French aviation. He has been the first to cross the channel on July the 25th onboard of the Blériot XI.
He graduated from Ecole Centrale de Paris

**Henri Marie Léonce Fabre** (1882 -1984) is a French engineer and pilot. He invented the seaplane in 1910.
He graduated from Supélec.