



Ecole Nationale
Supérieure
de l'Electronique
et de ses Applications

Evaluation of the internship for a student of E.N.S.E.A.

First name : COMTE-GAZ
Name Quentin
Year : 2012

Dates of internship: 25/06/2012 → 25/07/2012
Company : Wuhan University of Technology
Address : 205 Luoshi Road, Wuhan, P.R., China, 430070
Company Supervisor : Yang Jie
Contact Information : 466802324@qq.com

General Impression (check the ones that apply) :

Are you overall satisfied :

- With the behaviour of the intern yes no
- With his/her performance at work yes no
- With his/her general knowledge yes no

Would you be interested in having another intern from ENSEA under similar conditions? yes no

Appreciation of the internship report :

- Clarity of written expression good average weak
- Analytic and conceptual ability good average weak

General remarks about the behaviour of the intern :

- Autonomy good average weak
- Conscientiousness good average weak
- Tenacity at work good average weak
- Ability to work in a team good average weak
- Scientific and technical curiosity good average weak
- Creativity good average weak
- Clarity of oral expression good average weak
- Expertise with the tools used good average weak
- Work efficiency good average weak
- Interpersonal skills good average weak

Date and Signature of the Company Supervisor :

2012. 7. 25

Yang Jie



Cognitive radio

Project made by
Comte-Gaz Quentin

Internship in Wuhan University of Technology (China)
Years 2011 - 2012

ACKNOWLEDGMENT

First I want to thank my supervisor Mrs. Yang and my tutors Mr. Zhang Liangjun and Mrs. Wen Hanqing for allowing me to make this internship both interesting and rewarding within Wuhan University of Technology.

I also like to thank Mr. Tang and Mr. David (teachers at the ENSEA) and Mr. Cao Jiangshu without whom this internship would not have been possible.

To conclude, I would like to thank the many Chinese students (especially Ling and Meng) for allowing me to feel at ease and making me discover China.

TABLE OF CONTENTS

INTRODUCTION	4
PART 1 : DEFINITION AND PRINCIPLES	5
I) My project and how I managed my time	5
II) How cognitive radio works	5
1. Definition of Cognitive Radio.....	5
2. Interaction with its surroundings	6
PART 2 : ARCHITECTURE.....	9
I) Functional components of Cognitive Radio architecture	9
II) Classification of cognitive frameworks and implementations.....	9
III) The different kind of Unified Theory of Cognition frameworks.....	10
1. Simple.....	10
i. OODA.....	11
ii. CECA.....	12
2. Higher Complexity	12
i. SOAR.....	12
ii. STORM.....	13
iii. ACT-R.....	13
IV) The different kind of implementations.....	14
1. Mitola architecture (OODA)	14
2. Virginia Tech (OODA).....	14
3. OSCR (SOAR).....	15
4. xG Radio	15
PART 3 : HOW DO WE USE SOAR ARCHITECTURE.....	16
I) Description of Soar architecture and Soar agent.....	16
II) How to implement Soar agent	16
1. What kind of software did I use ?.....	16
2. Example of basic implementation	17
3. Example of complex implementation.....	18
i. Basic operations.....	18
ii. Never stay stuck, save valuable information and learn of your mistakes.....	19
GLOSSARY	21
REFERENCES	22

INTRODUCTION

Wireless technologies have multiplied during the last decade so there is a raising demand of electromagnetic spectrum. However, due to nowadays spectrum management, the offers did not meet the demands so that the electromagnetic spectrum became scarce.

In order to solve this problem, the spectrum needs to be well managed, therefore it needs to be used efficiently. Studies in cognitive radio area are led in that context. Cognitive radio is a system which allows a terminal to interact with its surroundings. In other words, cognitive radio can collect information from its surroundings, model them and adapt its behaviour according to them. Consequently, cognitive radio can detect free frequencies and use them, that's way it helps to improve the management of spectrums.

In this project, we will study different aspects of cognitive radio : Principles, architecture, specific implementations and design Soar agents.

DEFINITION AND PRINCIPLES

I. My project and how I managed our time

The working group is composed of one doctoral student named Zhang Liangjun, one Chinese student named Wen Hanqing, two french students (Kanto Andiambelo and myself) and chinese and french students in other laboratories (Chen Ling, Meng, Steven, Quentin, Mahéva ...).

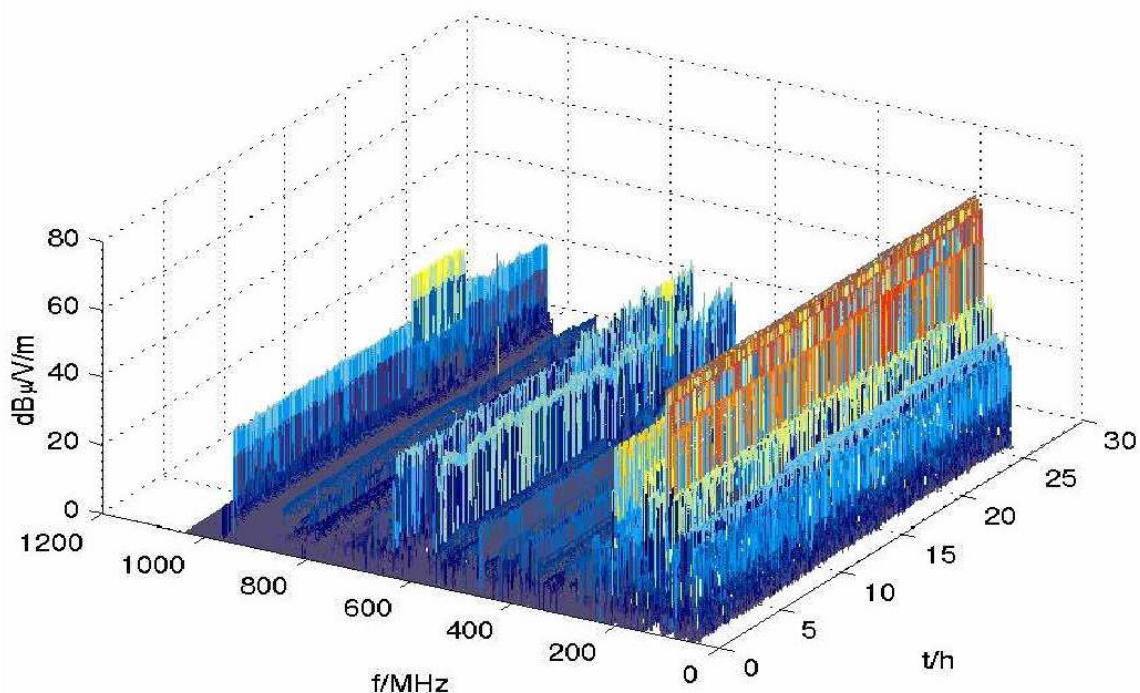
Cognitive radio is a complex area of study, the role of Kanto and I was largely to educate ourselves on this area. During the first week, I studied a lot of theoretical literature on the subject. During the second week, I made a record of my knowledge and I started to get into a technical description of Cognitive Radio and to study some architectures and implementations. The remaining time was devoted to design Soar agents with the help of some tutorials.

Knowledge of Mr. Liang has been very valuable for this internship.

II. How cognitive radio works

1) Definition of Cognitive Radio

As shown in Figure 1, there is substantial variation in underutilized spectrum in time and frequency, and although, even if it is not shown, there is also wide variation in term of space.



[Figure 1 : Measurement made in Germany [Jondral_04]]

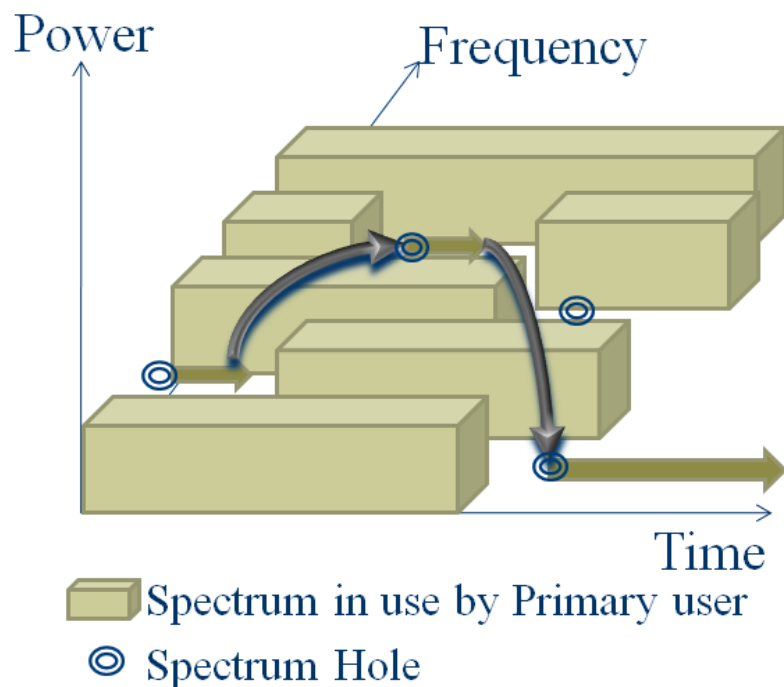
Cognitive radio (CR) is an emerging technology recently proposed to implement some form of intelligence that allows a terminal to have the capacity of learning: "feel" the radio environment and adapt it to the terminal. This provides to users increased throughput and overall increased comfort in their communications.

2) Interaction with its surroundings

To be intelligent, cognitive radio uses:

- **A spectrum sensing:** To find the frequencies used at one time, at a given location (the study must be made in all directions). Spectrum holes can then be detected.
- **A spectrum manager:** process of regulating the use of radio frequencies to promote efficient use and gain a social benefit.
- **A spectrum sharing:** sharing of spectrum between users according to their use (frequency, location, time) and spectrum occupancy.

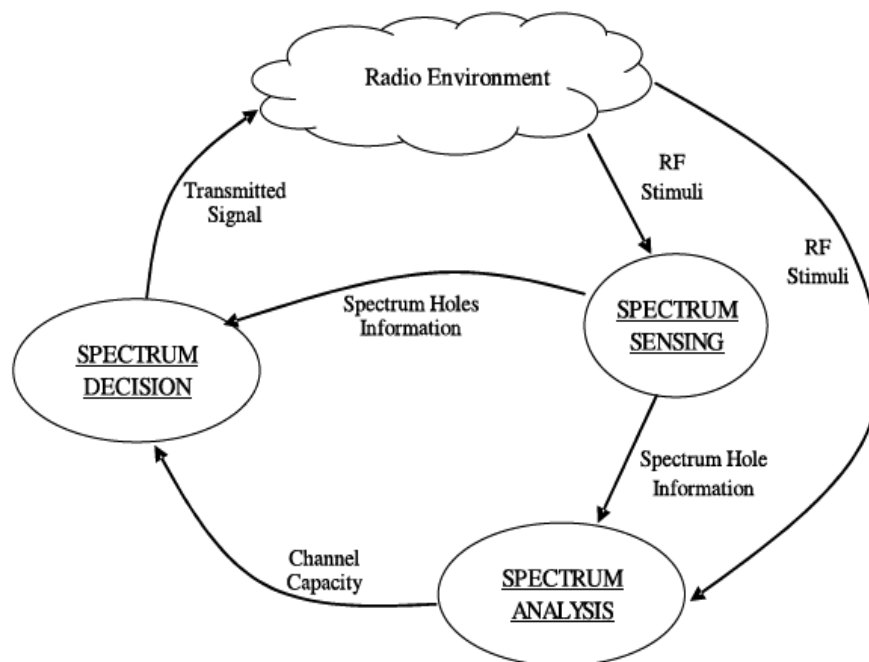
To solve the problem of **spectrum holes**, we can exploit locally unused spectrum to provide new paths to spectrum access (as shown in figure 2).



[Figure 2 : The main purpose of Cognitive Radio : exploit unused spectrum holes]
[The primary user is the first user using the frequency]

→ Finally, the spectrum is detected and we mark the holes (spectrum sensing). Then, we analyze : sending data to spectrum sensing spectrum analysis. We deduce which users can be assigned to which holes. Finally, decisions are based on the results of the analysis and the number of holes available. And so on.

Illustration of this cycle :



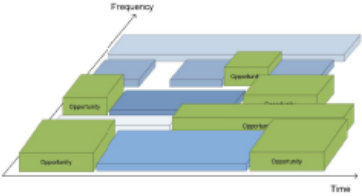
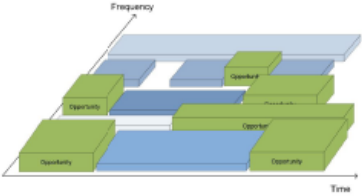
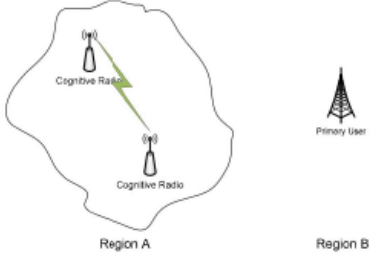
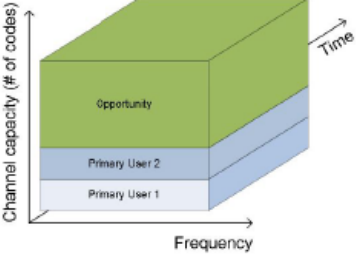
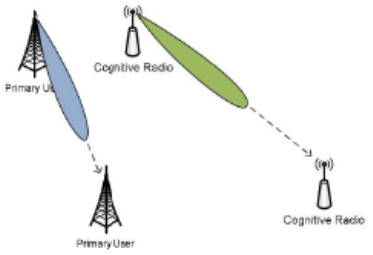
[Figure 3 : Basic cycle of Cognitive Radio]

To interact with its surroundings, a system needs to learn things about it. Without this we can't even hope to have an **intelligent** cognitive radio.

This is why **our system needs to capture and measure different sizes.**

Here are the main sizes we need to measure :

MULTI-DIMENSIONAL RADIO SPECTRUM SPACE AND TRANSMISSION OPPORTUNITIES

Dimension	What needs to be sensed?	Comments	Illustrations
Frequency	Opportunity in the frequency domain.	Availability in part of the frequency spectrum. The available spectrum is divided into narrower chunks of bands. Spectrum opportunity in this dimension means that all the bands are not used simultaneously at the same time, <i>i.e.</i> some bands might be available for opportunistic usage.	
Time	Opportunity of a specific band in time.	This involves the availability of a specific part of the spectrum in time. In other words, the band is not continuously used. There will be times where it will be available for opportunistic usage.	
Geographical space	Location (latitude, longitude, and elevation) and distance of primary users.	The spectrum can be available in some parts of the geographical area while it is occupied in some other parts at a given time. This takes advantage of the propagation loss (path loss) in space. These measurements can be avoided by simply looking at the interference level. No interference means no primary user transmission in a local area. However, one needs to be careful because of hidden terminal problem.	
Code	The spreading code, time hopping (TH), or frequency hopping (FH) sequences used by the primary users. Also, the timing information is needed so that secondary users can synchronize their transmissions w.r.t. primary users. The synchronization estimation can be avoided with long and random code usage. However, partial interference in this case is unavoidable.	The spectrum over a wideband might be used at a given time through spread spectrum or frequency hopping. This does not mean that there is no availability over this band. Simultaneous transmission without interfering with primary users would be possible in code domain with an orthogonal code with respect to codes that primary users are using. This requires the opportunity in code domain, <i>i.e.</i> not only detecting the usage of the spectrum, but also determining the used codes, and possibly multipath parameters as well.	
Angle	Directions of primary users' beam (azimuth and elevation angle) and locations of primary users.	Along with the knowledge of the location/position or direction of primary users, spectrum opportunities in angle dimension can be created. For example, if a primary user is transmitting in a specific direction, the secondary user can transmit in other directions without creating interference on the primary user.	

ARCHITECTURE

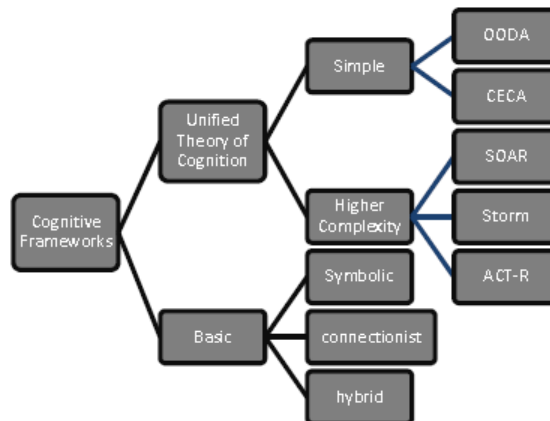
I. Functional components of cognitive radio architecture

The six functional components of cognitive radio architecture are:

- The sensory (Sensory Perception: SP) of the user includes haptic interface (touch), audio, video and detection and perception functions.
- The sensors of the local environment (location, temperature, accelerometer, etc..).
- The system applications (media services as an independent network game).
- The SDR₍₁₎ functions which include the detection and RF₍₂₎ radio applications of SDR.
- The functions of cognition (for control, planning and learning systems).
- The local effector functions (speech synthesis, text, graphics and multimedia posters).

II. Classification of cognitive frameworks and implementations

Cognitive radio is a concept. Thus, there are several ways to realize it. Here are the most common architectures :



[Figure 4 : Classification of Cognitive Frameworks]

	Name	Type	Description
Cognitive Architecture	OODA	Simple	Feedback loop developed for modeling situations requiring adaptation to changing conditions
	CECA	Simple	Expansion on OODA Loop to model situation in broader context
	SOAR	Complex	General cognitive architecture for modeling systems of complex behavior
	Storm	Complex	Extension on SOAR
	ACT-R	Complex	Similar to SOAR a programming language enables representation of tasks
Implementation	Mitola Architecture	OODA based	First cognitive architecture developed based on OODA feedback loop
	Virginia Tech 802.22	OODA using Case Based Reasoning (CBR)	Incorporates long term learning through the use of historical database
	Virginia Tech Public Safety Radio	OODA with CBR and Genetic Algorithms	Two loop cycle that first makes a decision on radio action then optimizes actions using genetic algorithms
	OSCR	SOAR based	Utilizes the SOAR Cognitive software to model and operate software defined radios
	xG Radio	Ontology based	Utilizes policy rules to govern radio behavior driven by spectrum observations

[Figure 5 : Summary of cognitive architectures and implementations]

III. The different kind of unified Theory of Cognition frameworks

As we said in the previous paragraph, there are many possible cognitive frameworks :

A concept known as the Unified Theory of Cognition (**UTC**) makes the case for a general set of assumptions that account for all cognition that remain constant across varied domains and knowledge spaces.

Many of the complex and basic cognitive architectures are based on this concept of UTC where **the architecture shows how intelligent entities react to inputs from the environment with goal driven behavior.**

1) Simple

There are two simple models : OODA and CECA.

i. OODA :

The OODA loop was originally developed to describe the methodology which fighter pilots utilize in a constant cycle during fight.

Cognitive radio architectures with OODA loop are **primarily designed to wait until an event happens before making a change to their configuration.**

The CR **must integrate the environmental observations with radio policies, user goals, limitations on configurations and past experiences in order to synthesize a complete image of the situation.**

This model must identify the configuration changes that are available and identify the best option to meet the new situation.

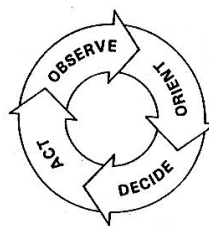
This model is composed of 4 parts :

- **Observe** : This part observes many outside information, the influence of the previous action of the loop, the influence of the field on the action.

- **Orient** : It is here that all previous information are, it's the "inheritance part". It is the most important part of the loop because it shapes the way we observe, the way we decide and the way we act. There is also a filtering of information that this bloc will send to the next.

- **Decide** : It is here that we decide. The decision is considered as a hypothesis. This part will find the best action in all possible one.

- **Act** : The hypothesis of the Decision part will be tested and will improve the Orientation and Decision part in future cycles.



This model is interesting because as long as a hypothesis of action is not made, immobility is guaranteed. It is **a way to lose**. This approach may encourage an opponent to lose if he does not understand the actions of the one in front of him.

Feint and tricks can also destroy the picture the enemy thinks about the situation.

Another form of encouragement to the defeat **is to have yourself a correct picture of the behavior and actions of the enemy**. This picture shows us where, how and when we can hit the enemy.

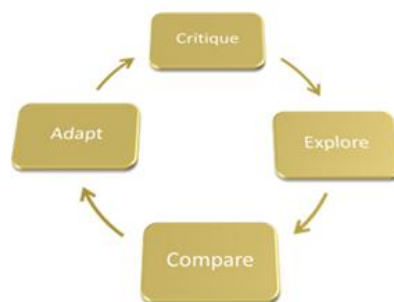
Another advantage of the OODA cycle is **that it accepts the chaos of the action**.

ii. CECA :

The OODA loop requires individualization when applied to specific **reactive situations**. The Critique-Explore-Compare-Adapt (CECA) loop expands on the OODA loop in order to study the broader context of **Command and Control**. Indeed, the CECA model has been created because **the OODA loop does not adequately describe the goal-oriented command decision making process**.

In contrast to OODA, CECA intends to **apply social cognition that incorporates multiple users working on complex problems**. This is a really powerful way to see things.

The four phases of the CECA Loop broadly correspond to the identification of information needs (**Critique**), active and passive data collection and situation updating (**Explore**), comparison of the current situation to the conceptual model (**Compare**), and adaptation to aspects of the battlespace that invalidate the conceptual model or block the path to goal completion (**Adapt**).



Among the advantages of the CECA Loop over the OODA Loop are **greater insight into the nature of perception and understanding, introduction of critical thinking elements, and exposition of the central role of planning and the mental representation of operational concepts**.

2) Higher Complexity

Higher complexity architectures are based on the Unified Theory of Cognition (UTC) philosophy.

I'll explain you some of the most prominent cognitive architectures in development :

i. SOAR Cognitive Engine :

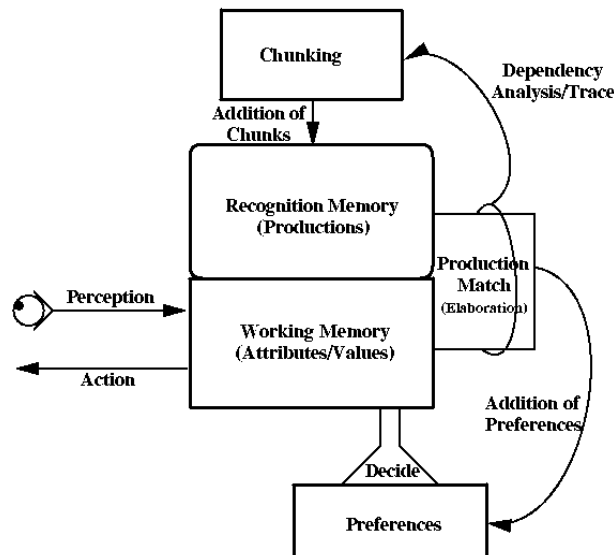
Developed in 1983, this popular general cognitive architecture is used to develop systems that exhibit **intelligent behaviour**.

This project **is made to work on the full range of tasks expected of an intelligent agent, from highly routine to extremely difficult, open-ended problems**.

This architecture must use and create an appropriate representation form of knowledge, such as procedural, declarative, episodic, and possibly iconic.

The goal of Soar project is to reach a general intelligence (which has not yet been reached) which employ the full range of problem solving methods; interact with the outside world, and; learn about all aspects of the tasks and its performance on them.

This architecture uses **Short-term (working) and Long-term (production) memory** to be **more efficient**.

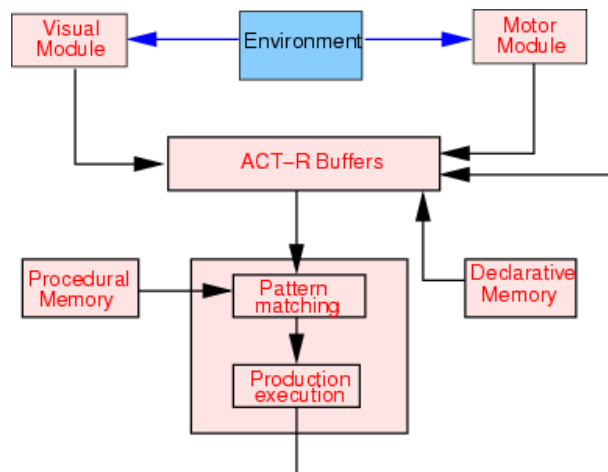


ii. STORM :

The Storm Cognitive framework was developed as an extension to the Soar Cognitive Architecture. Using knowledge gained from the fields of psychology and brain based science, the researchers at Michigan University adapted memory, learning and emotion to the original Soar architecture. Storm is an overarching framework designed to develop biologically inspired cognitive architectures (BICA).

iii. ACT-R :

ACT-R, developed at Carnegie Mellon University, is a cognitive architecture theorizing how human cognition works. ACT-R looks like a programming language; however, its constructs reflect assumptions about human cognition. These assumptions are based on numerous facts derived from psychology experiments. This architecture enables representation of tasks. This model allows a writer to add in their own assumptions of a specific task.



Use other algorithms :

Incorporating novel algorithms into architectures :

Another possibility is to use bio-inspired algorithms such as Particle Swarm Optimization (PSO). Indeed, the behavior of birds that travel in groups (PSO algorithms) is the same as cognitive radio : each individual only has knowledge of their nearest neighbor.

We can also use Grey Modeling and Grey Relational Analysis (GRA) which identify similarity between observed data and a reference dataset set of cognitive radio.

IV. The different kind of implementations ⁽³⁾

The cognitive architecture in use today are all built upon a basic premise of using observations of the environment as a catalyst for reconfiguring system parameters to achieve more optimal performance. Long term learning is also incorporated as part of initial decision making. Their primary differences lie in the implementations of this general goal. I will present you some of those numerous implementations :

1) Mitola architecture (OODA)

Joe Mitola proposed that the **reconfigurability enabled by software defined radio created an incredible potential for improving radio operation**. He went on to introduce a simplified cognition architecture that is founded upon the concepts of the OODA loop introduced earlier.

Mitola's adaptation of the basic OODA loop within a radio framework provided the seed for a new evolution in wireless communications. The observation step incorporates spectrum and network sensing while the orientation step prioritizes the incoming observations. Much of the ensuing work within cognitive radio has centered on the decision making and learning stages within this cognitive architecture as discussed in the next sections.

2) Virginia Tech (OODA)

Virginia Tech is developing an open source Cognitive Radio architecture named CROSS. The process includes a methodology for representing the situation as well as the procedures for making a decision and learning from a decision. The objective of the design is to develop a **distributed & modular system** that provides portability and interoperability between components developed in different programming languages, across different SDR and hardware platforms. This will enable **more flexible and streamlined development** of cognitive radio systems. **Users of CROSS can focus entirely on one aspect of the cognitive radio without developing or modifying components that have no direct relevance to their specific focus of research.**

3) OSCR (SOAR)

The Open Source Cognitive Radio (OSCR) project designed a framework to enable integration of cognitive engines with multiple Software Communications Architecture (SCA). The OSCR links multiple radios' application programming interface (API) with a single cognitive engine. This allows **one cognitive engine to operate multiple radios which may have differing capabilities**. The Soar cognitive engine was integrated with an SCA compliant SDR. The CE was programmed with a goal of maximizing the capacity of a noisy channel within a fluctuating noisy environment.

Here are some interesting things OSCR can do :

- **An operator is proposed for all possible combinations of settings.**
- Calculations are elaborated as **short term memory** elements.
- Operators are proposed and **ranked by capacity**.
- Given the **unimodal nature of the solution space for a given noise environment**, changes in the direction of decreasing capacity will never be applied again.
- When a new noise environment is detected, certain changes will be once again allowed.

4) xG Radio

The xG Radio architecture utilizes a **Spectrum Policy analyst** to create and modify policies using the Web Ontology Language (OWL). OWL is a web based family of languages that creates a **formal knowledge representation**. These policies are based on prior coordination and goals of the force commanders. A spectrum manager downloads policies to all xG enabled radios. The DSA capability of the radios provides a measure of spectrum occupancy that is updated to the fusion nodes that provide spectrum white space probability tables to all xG nodes.

HOW DO WE USE SOAR ARCHITECTURE ?

I) Description of Soar architecture and Soar agent⁽⁴⁾

You can review the description of the SOAR architecture description on page 12 before beginning this “technical description” of SOAR.

Soar is a unified architecture for developing intelligent systems. It provides the fixed computational structures in which knowledge can be encoded and used to produce action in pursuit of goals. In many ways, it is like a programming language, albeit a specialized one. It differs from other programming languages in that it has embedded in it a specific theory of the appropriate primitives underlying reasoning, learning, planning, and other capabilities that we hypothesize are necessary for intelligent behavior. Soar is not an attempt to create a general purpose programming language. Some computations are difficult or awkward to do in Soar (such as complex math) and they are more appropriately encoded in a programming language such as **C, C++, or Java**. **Soar is appropriate for building autonomous agents that use large bodies of knowledge to generate action in pursuit of goals.**

All of the knowledge in a Soar agent is represented as **if-then rules**. In Soar, rules are called productions, and we will use the terms interchangeably. Rules are used to select and apply things called operators.

So, **Soar works by using conditions (like “if-then” or “while” conditions).**

To determine if the conditions are true, Soar compares them to data structures in working memory. **Working memory defines the current situation**, which for an agent consists of its perception of its world, results of intermediate calculations, active goals, and operators.

When rules fire, they can make changes to working memory, as well as performing simple actions such as printing messages in the interaction window.

II) How to implement Soar agent

1) What kind of software did I use ?

Soar has its own editor, called VisualSoar, which is highly recommended if we want to implement Soar programs.

There is also a debugger named “SoarDebugger”.

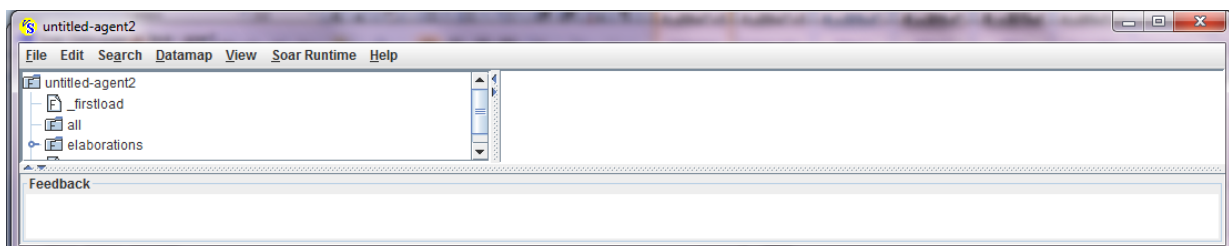


Figure 6 : Visual Soar (Soar editor)

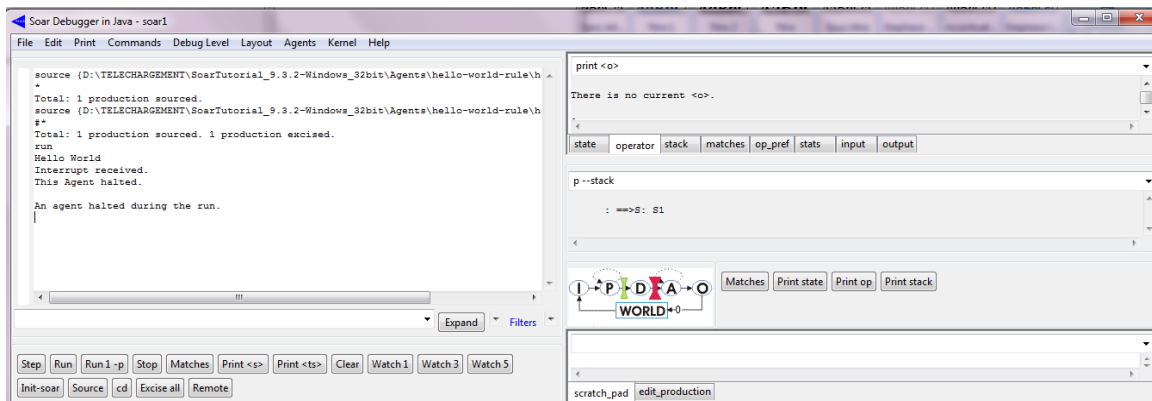


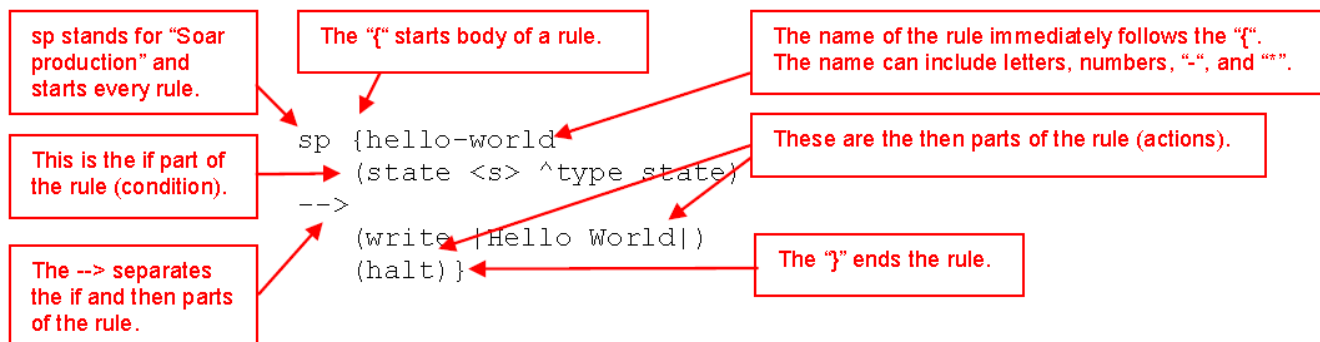
Figure 7 : Soar Debugger

Here are all software and tutorials I used : <http://sitemaker.umich.edu/soar/home>

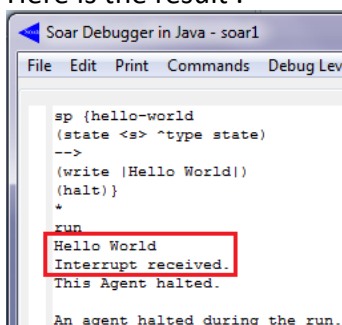
2) Example of basic implementation

The easiest program we can implement is a “Hello world” (the goal is to execute this : “If I exist, then write ‘Hello World’ and halt”).

Here is the source code :



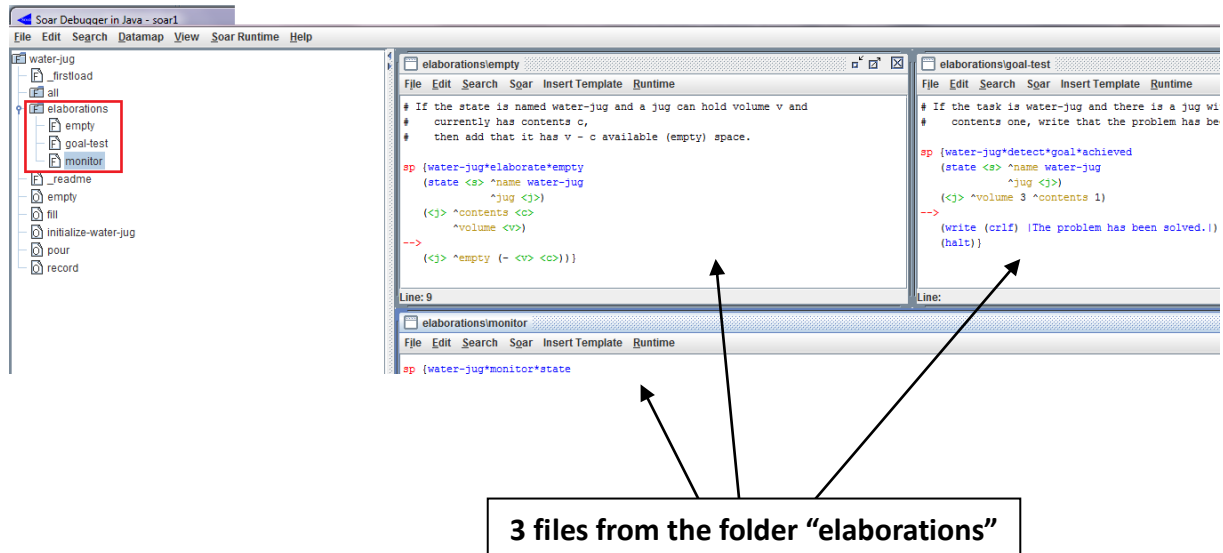
Here is the result :



We can note that the program is working well.

3) Example of complex implementation

With Visual Soar, it is possible to design a complex source code with many files.



Now I will show you a game named “Pacman” : The goal is to eat as much food as possible before getting eaten by the A.I.(5).

If you want to try this game, go to this address : <http://gamezone.2001jeux.com/play-00562-b-pacman.html>

We will design some A.I. that can play for us.

i. Basic operations

The basic way is the move-to-food operation in any direction : **If there is no food nearby, no instances of the operator will be proposed and the halt operator will be proposed.**

If there is food in an adjacent cell, this program will propose to move-to-food in the direction of that cell and indicate that this operator can be selected randomly.

If the move-to-food operator for a direction is selected, it will generate an output command to move in that direction.

If the move-to-food operator is selected and if there is a completed move command on the output link, then it will remove that command.

Here is the source code :

```
sp {propose*move-to-food
  (state <s> ^io.input-link.my-location.<dir>.content
    << normalfood bonusfood >>)
-->
  (<s> ^operator <o> + =)
  (<o> ^name move-to-food
    ^direction <dir>)}

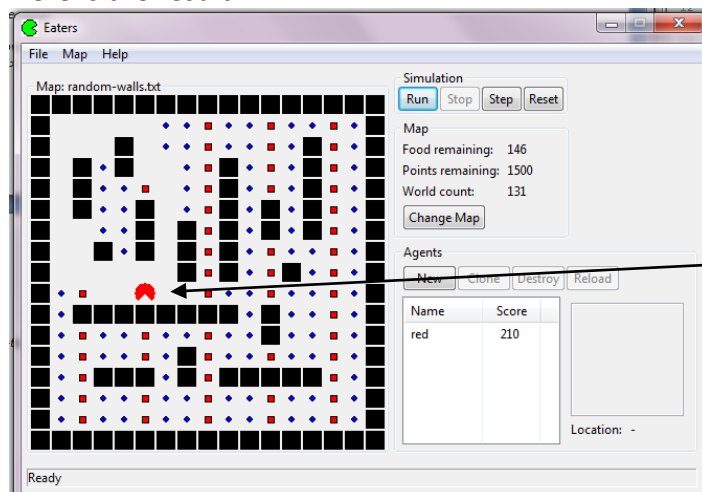
# Apply*move-to-food

sp {apply*move-to-food
  (state <s> ^io.output-link <o>
    ^operator <o>)
  (<o> ^name move-to-food
    ^direction <dir>)
-->
  (<o> ^move.direction <dir>)}

# Apply*move-to-food*remove-move:

sp {apply*move-to-food*remove-move
  (state <s> ^io.output-link <o>
    ^operator.name move-to-food)
  (<o> ^move <move>)
  (<move> ^status complete)
-->
  (<o> ^move <move> -)}
```

Here is the result :



The A.I. is blocked because there is no food in its neighborhood.

We can see that this program is **not really effective**.

Indeed, **it just goes on a random food and it is stuck when there is no food** (it will not try to go around).

ii. Never stay stuck, save valuable information and learn of your mistakes

Let's modify the previous program to make it more effective.

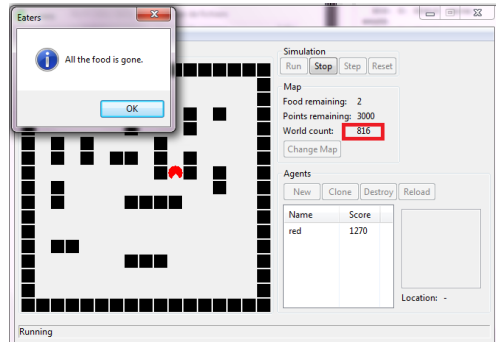
Assuming we change the map each time we run the game, key points we need to improve are :

- **Save routes**
- **Do not walk randomly while we have information at our disposal**
- **If the game drags on, try some boxes randomly**
- **Save the actions and incorporate them into the system if they were useful.**

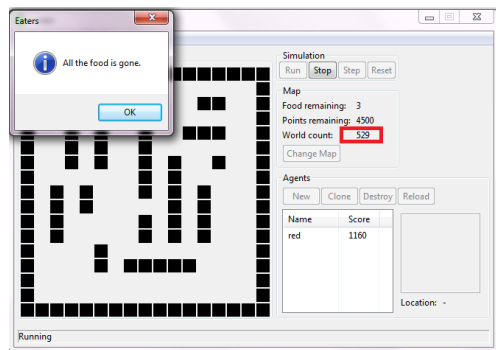
As in the basic program, we must use only “if-then” conditions.

Here is the result :

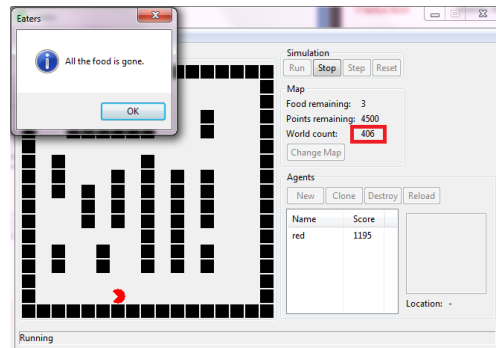
-First run :



-Second run :



- Third run :



We can see that the more you run the program, the more efficient it is (the first time we launch the program, we needed 816 actions, the second time, 526, the third time, only 406 ...).

This kind of program is much more interesting than the previous one.

It is however possible to improve this program.

For example, we could assume that there are 2 players on the map.

Then, the goal would be to take the food before the opposing team.

Unfortunately, this kind of algorithm quickly becomes complicated.

It was therefore impossible to implement such a program in the allotted time.

In conclusion, it is possible to design a software learning of its previous actions.

Those kind of software are not perfect but since some years, they are becoming more and more efficient.

GLOSSARY

- (1) SDR** : A software-defined radio system, or SDR, is a radio communication system where components that have been typically implemented in hardware are instead implemented by means of software on a personal computer or embedded computing devices.
- (2) RF** : Radio frequency (RF) is a rate of oscillation in the range of about 3 kHz to 300 GHz, which corresponds to the frequency of radio waves, and the alternating currents which carry radio signals.
- (3) Implementation** : In computer science, an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through programming and deployment. Many implementations may exist for a given specification or standard.
- (4) Agent** : In computing, an agent is the equivalent of a software robot. This is a program that performs tasks in the manner of an automaton and depending on what its author asked. It is an autonomous entity capable of communicating with a partial knowledge of his surroundings and private behavior, as well as the ability to execute its own.
- (5) A.I.** : Artificial intelligence (AI) is the intelligence of machines and the branch of computer science that aims to create it.

REFERENCES

- [1] **"Survey of Cognitive Radio Architectures"** by Ashwin Amanna, Jeffrey H. Reed – VirginiaTech
- [2] IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 23, NO. 2, FEBRUARY 2005 Page 201 to 220 **"Cognitive Radio: Brain-Empowered Wireless Communications"**
- [3] **"A Survey on Spectrum Management in Cognitive Radio Networks"** by *Ian F. Akyildiz, Won-Yeol Lee, Mehmet C. Vuran, and Shantidev Mohanty, Georgia Institute of Technology*
- [4] ScienceDirect, Ad Hoc Networks 9 (2011) Page 228 to 248 **"Routing in cognitive radio networks: Challenges and solutions"**
- [5] ScienceDirect, Ad Hoc Networks 7 (2009) Page 1315 to 1329 **"A survey on MAC protocols for cognitive radio networks"**
- [6] EURASIP Journal on Advances in Signal Processing Volume 2010, Article ID 381465, 15 pages **"A Review on SpectrumSensing for Cognitive Radio: Challenges and Solutions"**
- [7] ScienceDirect, Physical Communication 4 (2011) Page 40 to 62 **"Cooperative spectrum sensing in cognitive radio networks: A survey"**
- [8] **"Joint Routing and Spectrum Selection for Multihop - Cognitive Radio Networks"** by Hicham Khalife, Satyajeet Ahuja, Naceur Malouch and Marwan Krunz.
- [9] Wireless Pers Commun (2011) **"A Distributed Message-passing Approach for Clustering Cognitive Radio Networks"** by Kareem E. Baddour, Oktay Üreten and Tricia J. Willink
- [10] ScienceDirect, Computer Communications 32 (2009) 1983–1997 **"Search: A routing protocol for mobile cognitive radio ad-hoc networks"**
- [11] ScienceDirect, Ad Hoc Networks 9 (2011) 228–248 **"Routing in cognitive radio networks: Challenges and solutions"**
- [12] IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 29, NO. 4, APRIL 2011 p794 to 804 **"CRP: A Routing Protocol for Cognitive Radio Ad Hoc Networks"**
- [13] ScienceDirect, Performance Evaluation 68 (2011) 859–875 **"End-to-end protocols for Cognitive Radio Ad Hoc Networks: An evaluation study"**
- [14] **"Non-Cooperative Spectrum Access in Cognitive Radio Networks: a Game Theoretical Model"** by Jocelyne Elias, Fabio Martignon, Antonio Capone and Eitan Altman
- [15] IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, VOL. 10, NO. 9, SEPTEMBER 2011, Page 2940 to 2950 **"Joint Routing and Spectrum Allocation for Multi-Hop Cognitive Radio Networks with Route Robustness Consideration"**
- [16] IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 10, NO. 7, JULY 2011, Page 954 to 967 **"Maximizing Capacity in Multihop Cognitive Radio Networks under the SINR Model"**
- [17] IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 29, NO. 4, APRIL 2011, Page 784 to 794 **"Multicast Communications in Multi-Hop Cognitive Radio Networks"**
- [18] IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 26, NO. 1, JANUARY 2008, Page 146 to 156 **"Spectrum Sharing for Multi-Hop Networking with Cognitive Radios"**