Tecnológico de Monterrey

TC2006. Programming languages.

Project staffing with logic programming.

Jose Eduardo Cadena Bernal | A01704641

Benjamín Valdés Aguirre PhD

May 26th 2021

Contents

Abstract.

The present project makes usage of the logic programming paradigm to be able to simulate a staff assignment program based on project requirements and staff availability, role, and skills. To make the solution more user friendly I decided to connect the logic implemented in prolog with Java for the graphical user interface, using the JPL library.

Context of the problem.

There are a lot of factors why projects end up being unsuccessful:

- Unclear objectives.
- Wrong selection of projects.
- Wrong staff assignment.
- Non-detailed work plans.
- Unrealistic commitments.
- Non-existent risk management.
- Constant change of scope.
- Inconsistent processes.
- Poor communication.
- Lack of clear and timely indicators.

According to the Standish group in North America only the 30% of projects are successful.

CHAOS RESOLUTION BY AREA OF THE WORLD

	SUCCESSFUL	CHALLENGED	FAILED
North America	30%	53%	17%
Europe	29%	54%	17%
Asia	23%	57%	20%
Rest of World	26%	51%	23%

The resolution of all software projects from FY2012–2016 by the four major areas of the world.

What does that mean? A successful project is one that ends on time, cost and scope.

Time and Cost over run

	1998	2000	2002	2004	2006
◆ Cost	69%	45%	43%	56%	47%
■ Time	79%	63%	82%	84%	72%
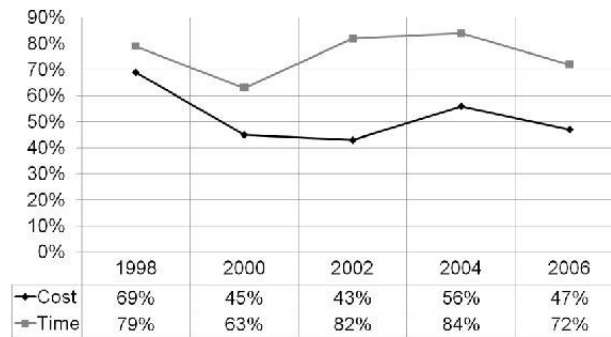
Source Standish Group 2007 Chaos study

25

When we are talking about project management, it is fundamental to choose the right members according to the tasks that they will need to perform. A big part of an efficient administration is based on be sure that the right members are performing the right task and participating in the corresponding project based on their skills.

According to the site CNN expansion, companies make the mistake to assign people to projects based on their availability, luck, or favoritism, by staffing projects that way there are a lot of probabilities that the project is not going to be successful. That's the worst way to assign people to a project. Not all of us have the same training, capacities, aptitudes, or skills. Assigning human resources that fit the needs of the project ensures a very high probability of success.

Not having a good selection and recruitment method can trigger a series of errors that lead to loss of various resources.

What consequences can a bad decision have when it comes to recruiting?

- Loss of productivity during the integration process. This loss is normal in new hires, but if the learning curve is too slow because integration is being

problematic, the profitability of the operation can be seriously damaged and cause additional stress on all those involved.

- Administrative costs involved in the selection and hiring. In addition to those derived from an eventual dismissal.

- Decrease in the confidence of the rest of the actors that make up the company's ecosystem. And trust in a company affects both employees, customers, suppliers and partners. The discredit is directly linked to the decline in profits.

- Increased turnover. If the error is not corrected in time, it is not refined to adjust and improve the process of selecting new recruits, this instability in the staff may be too recurrent.

- Bad environment. The inadequate integration of a person in a logical time to the rhythms of the company, to its culture, will inevitably cause a bad work environment.
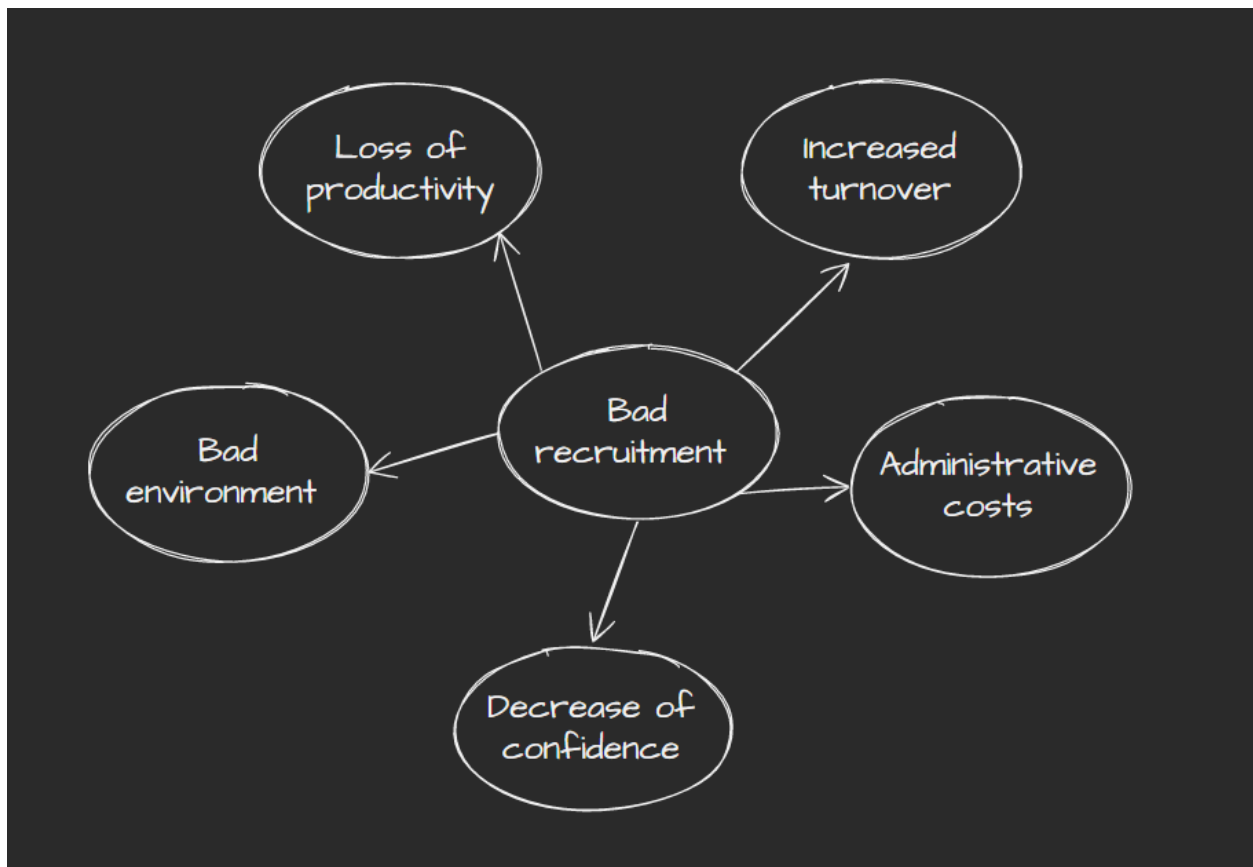
Solution.

For this project I was inspired by my past experiences, I have been in projects where members don't fit project requirements which in consequence reduces productivity, increases stress levels, and delays project deliverables.

The idea of this program is to give all the possible combinations of members according to project requirements, business rules and staff skills, availability, and roles within the organization.

Implementation.

The project is programmed in Prolog, this language is a declarative logic programming language.

The execution of prolog progam is equivalent to searching possibilities and determining the objects which satisfy the given rules.There can be several answers which are true in the given circumstances.The program does not terminate as soon as the first found answer, it keeps until the entire tree of possibilities has been checked .

Logic programming is a [programming paradigm that is based on logic. This means that the language has sentences that follow logic, the express facts and rules.Computation using logic programming is done by making logical inferences based on all the available data. In order for computer programs to make use of logic, there must be a base of existing logic , called predicates. Predicates are used to build atomic formulas or atoms which state true facts. Predicates and atoms are used to create formulas and perform queries.

Logic programming can help trough:

- Natural language processing: allows for better interactions between humans and computers
- Database management: can be used for creating , maintaining and querying of noSQL database
- Predictive analysis: with a lot of data, the language can search for inconsistencies or areas of differentation in order to make predictions.
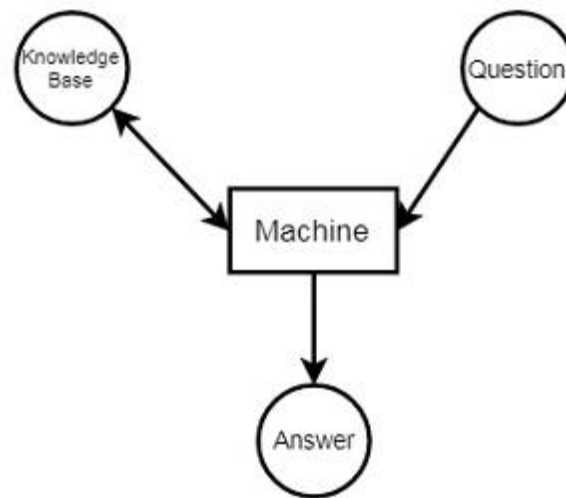
Logical Programming

Diagram

I decided to make a diagram to visually demonstrate my data model:
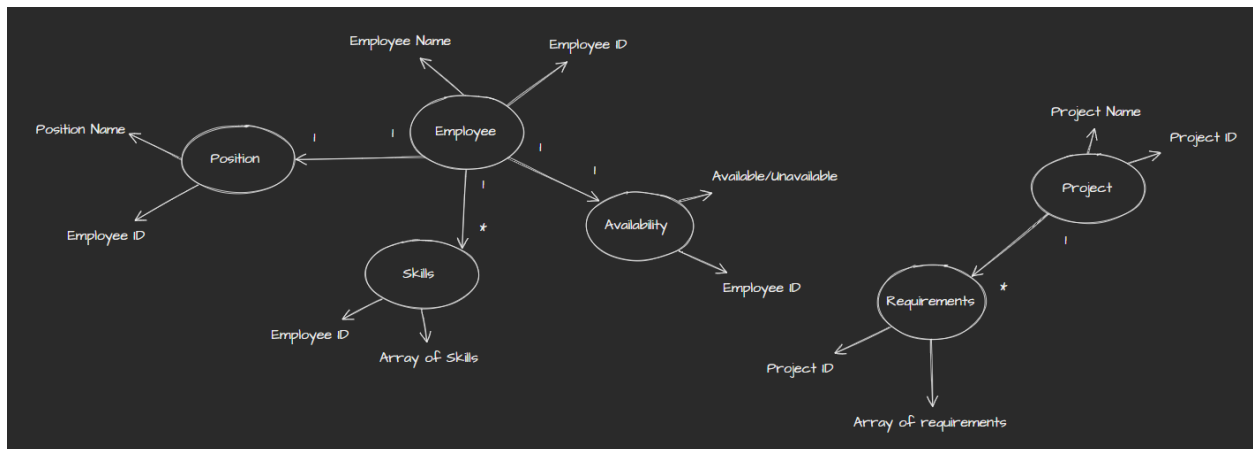
- ➢ Position, Project and Employee have name and ID as attributes.
- ➢ Skills and Requirements are represented as an array.
- ➢ Position has a 1 to 1 relationship with Employee, having as foreign key the ID of the employee.
- ➢ Availability has a 1 to 1 relationship with Employee, having as foreign key the ID of the employee.
- ➢ Skills has a 1 to many relationship with Employee, having as foreign key the ID of the employee.
- ➢ Requirements has a 1 to many relationship with Project, having as foreign key the ID of the project.

Business rules

To assign employees to a project the program follows these rules:

- The total number of members for each project is 3.
- At least one member must be Solution Architect.
- At least one member must be Developer Senior.
- At least one member must be Developer Junior.
- The Solution Architect must satisfy 50% of the requirements of the project.
- The Developer Senior must satisfy 33% of the requirements of the project.

- The Developer Junior must satisfy 10% of the requirements of the project.

- All members must be available.

- If no one in any position satisfy with the percentage of the requirements, the program returns all the employees in this position who are available.

Establish facts and relations.

We can define the base cases and establish facts such as:

Defining employees:

employee('Angeles Anaya',1).

Stablishing employee position:

employee_position(1,solutionArchitect).

Defining employee skills:

employee_skill(1,[azure,cloud,java,cpp,excel,python,prolog,db,c]).

Defining the availability:

employee_available(1,unavailable).

Defining the project:

project_id(overtimetool,00099).

Defining project requirements:

project_requirements(00099,[cloud,azure,excel,python]).

Unification.

Unification is very useful in this paradigm, by replacing certain sub-expression variables with other expressions, unification tries to identify two symbolic expressions.

```
check_requirements(Proyect,[H|TA],[H|R],P):-
    employee(_Nombre,H),
    employee_available(H,Available),
    available(Available),
    project_id(Project,IdProject),
    employee_skill(H,Skills),
    project_requirements(IdProject,Requirements),
    intersect(Skills,Requirements,ListIntersected).
    list_length(ListIntersected,Size),
    list_length(Requirements,SizeRequirements),
    fits_criteria(SizeRequirements,Size,P),
```

In this call we unify the variable Available as shown in the image below:

Call: *employee(_30680, 1)*
Exit: *employee('Angeles Anaya', 1)*
Call: *employee_available(1, _30552)*
Exit: *employee_available(1, unavailable)*

Backtracking.

This searches the truth value of different predicates by checking whether they are correct or not. In other words the program will go back to the previous goal, and it will try to find another way to satisfy the goal.
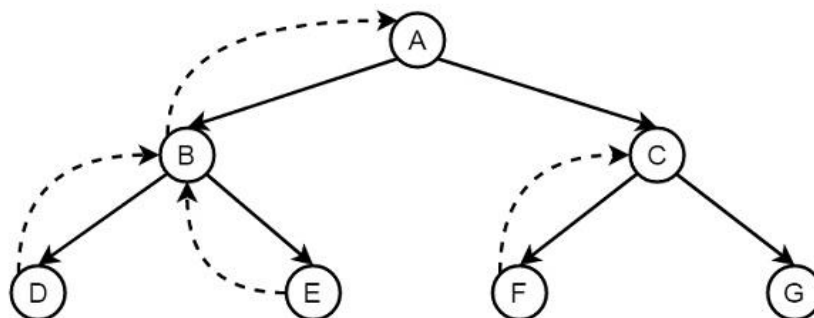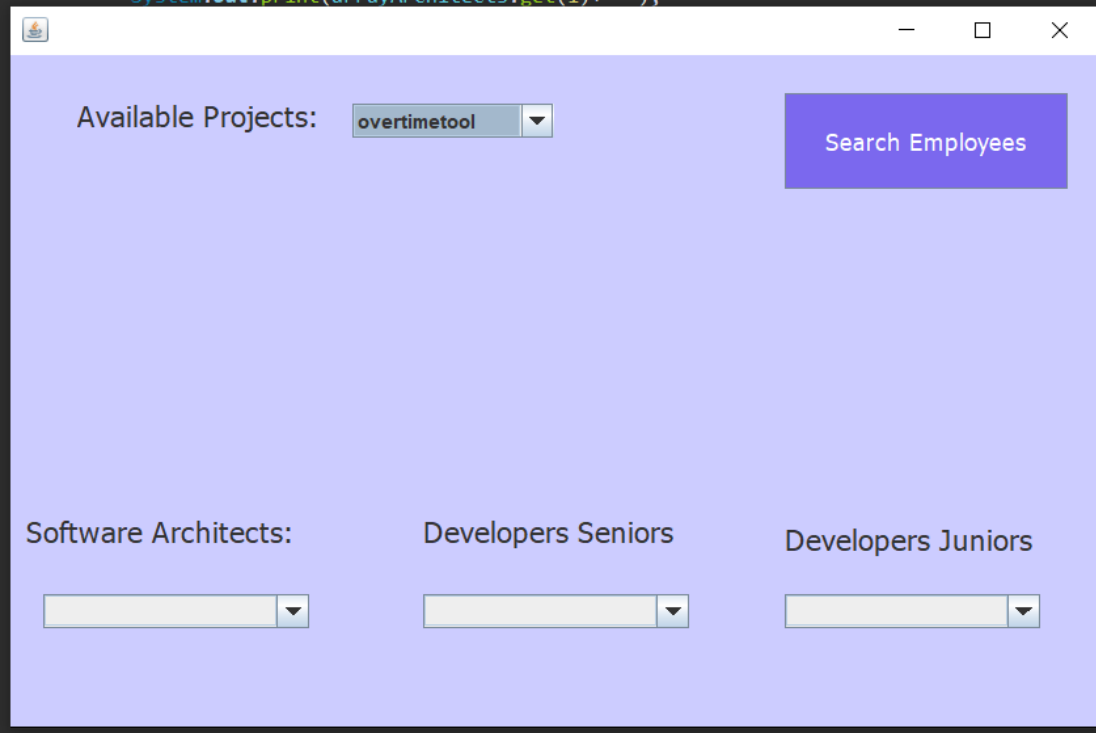

Recursion.

Recursion in any type or programming language are functions that call itself until the goal has been successful.

In prolog recursion appears when a predicate contains a goal that refers to itself.

```prolog
%function to check the available employees
%receives a list of id and return the available ids of the employees
get_available([],[]).
get_available([H|TA],[H|R]):-
    employee_available(H,Available),
    available(Available),
    get_available(TA,R).
get_available([_H|TA],NE):-
    get_available(TA,NE).
```

Results.

When you open the user interface you will find a dropdown which shows you the different projects, also you have 3 empty dropdowns, one for the Software Architects, other for Developers Seniors and another for the Developers Juniors.

You select the Project for which you want to assign employees and click in search for options.

Immediately the program will send the query to the prolog program, and it will start to implement the different rules to search the best members for the team.

The query looks like this:

```
create_Team_Options(datawarehouse,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)
```

The first parameter you send is the name of the project, and the next ones are three different variables that will contain the list of architects, senior and junior developers that fit project criteria.

The list of Flags returns either a 1 or a 0 if there was at least one employee who fits in that project.

Example, there is at least one employee for 1 position that fits the criteria.

Flags = [0, 0, 0],
Nameavailableseniors = ['Renata Garcia', 'Sofia Rodriguez', 'Mateo da Silva', 'Leonardo Di Caprio', 'Emiliano Persaud', 'Miguel Smith', 'Alexander Smith'],
Namesavailablearquitects = ['Jonathan Cruz'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Antonio Rusu', 'Daniel Rossi', 'Miguel Tamm', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']

Example, there is nobody that fits the criteria.

Flags = [1, 1, 1],
Nameavailableseniors = ['Renata Garcia', 'Sofia Rodriguez', 'Mateo da Silva', 'Santiago Gonzalez', 'Daniel de la Hoya', 'Regina Quisque', 'Leonardo Di Caprio', 'Emiliano Persaud', 'Miguel Smith', 'Alexander Smith', 'Ali Mohammed'],
Namesavailablearquitects = ['Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose', 'Gilberto Ortega'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Antonio Rusu', 'Daniel Rossi', 'Miguel Tamm', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']

Example if some of the positions does not fit to the criteria, in this case the architects does not fit.

create_Team_Options(newDB,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [1, 0, 0],
Nameavailableseniors = ['Renata Garcia', 'Sofia Rodriguez', 'Mateo da Silva', 'Leonardo Di Caprio', 'Emiliano Persaud', 'Miguel Smith'],
Namesavailablearquitects = ['Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose', 'Gilberto Ortega'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Antonio Rusu', 'Daniel Rossi', 'Miguel Tamm', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']

Next   10   100   1,000   Stop

Example of the main view.

Example of the view if someone of the positions does not fit with the project criteria.
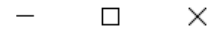
Available Projects: newDB ▼

Search Employees

There are no architectures that fits your project needs,there is a list of all the Architectures.

Developers Seniors that fits your project needs

Developers Juniors that fits your project needs

Software Architects:

Developers Seniors

Developers Juniors

Jonathan Cruz ▼

Renata Garcia ▼

John Yakimeshi ▼

Example of the view if no positions have employees that fits with the project criteria.

Tests.

The program was tested with different real scenarios and 33 people. The results are showed above.
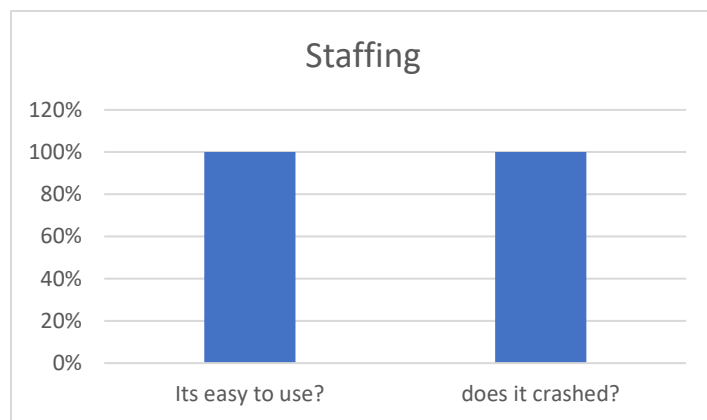
	Its easy to use?	does it crashed?
Person 1	yes	no
Person 2	yes	no
Person 3	yes	no
Person 4	yes	no
Person 5	yes	no
Person 6	yes	no
Person 7	yes	no
Person 8	yes	no
Person 9	yes	no
Person 10	yes	no
Person 11	yes	no
Person 12	yes	no
Person 13	yes	no
Person 14	yes	no
Person 15	yes	no
Person 16	yes	no
Person 17	yes	no
Person 18	yes	no
Person 19	yes	no
Person 20	yes	no
Person 21	yes	no
Person 22	yes	no
Person 23	yes	no
Person 24	yes	no
Person 25	yes	no
Person 26	yes	no
Person 27	yes	no
Person 28	yes	no
Person 29	yes	no
Person 30	yes	no
Person 31	yes	no
Person 32	yes	no
Person 33	yes	no
Total	100%	100%

Staffing

Scenarios tested:

There are only 1 position available.

create_Team_Options(overtimetool,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [0, 1, 1],
Nameavailableseniors = Namesavailablejuniors, **Namesavailablejuniors** = [],
Namesavailablearquitects = ['Jonathan Cruz', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose', 'Renata Garcia', 'Sofia Rodriguez', 'Mateo da Silva', 'Regina Quisque', 'Leonardo Di Caprio', 'Emiliano Persaud', 'Miguel Smith', 'Alexander Smith', 'Ali Mohammed', 'John Yakimeshi', 'Antonio Rusu', 'Daniel Rossi', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas']

Next | 10 | 100 | 1,000 | Stop

create_Team_Options(overtimetool,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [1, 0, 1],
Nameavailableseniors = ['Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose', 'Gilberto Ortega', 'Renata Garcia', 'Sofia Rodriguez', 'Mateo da Silva', 'Daniel de la Hoya', 'Regina Quisque', 'Leonardo Di Caprio', 'Emiliano Persaud', 'Miguel Smith', 'Alexander Smith', 'Ali Mohammed', 'John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Antonio Rusu', 'Daniel Rossi', 'Miguel Tamm', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo'],
Namesavailablearquitects = Namesavailablejuniors, **Namesavailablejuniors** = []

Next | 10 | 100 | 1,000 | Stop

create_Team_Options(overtimetool,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [1, 0, 0],
Nameavailableseniors = ['Daniel Rossi'],
Namesavailablearquitects = [],
Namesavailablejuniors = ['Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose', 'Gilberto Ortega', 'Renata Garcia', 'Sofia Rodriguez', 'Mateo da Silva', 'Santiago Gonzalez', 'Daniel de la Hoya', 'Regina Quisque', 'Leonardo Di Caprio', 'Emiliano Persaud', 'Miguel Smith', 'Alexander Smith', 'Ali Mohammed', 'John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Antonio Rusu', 'Miguel Tamm', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']

Next | 10 | 100 | 1,000 | Stop

All the employees are available:

create_Team_Options(overtimetool,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [0, 0, 0],
Nameavailableseniors = ['Victoria Garcia', 'Renata Garcia', 'Sofia Rodriguez', 'Alexander Quisque', 'Mateo da Silva', 'Daniel de la Hoya', 'Regina Quisque', 'Leonardo Di Caprio', 'Diego Mamani', 'Emiliano Persaud', 'Miguel Smith', 'Alexander Smith', 'Diego Rolle', 'Ali Mohammed'],
Namesavailablearquitects = ['Angeles Anaya', 'Jonathan Cruz', 'Eduardo Cadena', 'Angie Contreras', 'Maria Delgado', 'Jose Jose', 'Rodrigo de la M'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Juana Peeters', 'Antonio Rusu', 'Daniel Rossi', 'Miguel Tamm', 'Fernanda de Jong', 'Petter Ivanov', 'Liam Cuarto', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo', 'Jayden Gomez']

Next | 10 | 100 | 1,000 | Stop

Everybody is unavailable:

create_Team_Options(overtimetool,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [1, 1, 1],
Nameavailableseniors = Namesavailablearquitects, **Namesavailablearquitects** = Namesavailablejuniors, **Namesavailablejuniors** = []

Next | 10 | 100 | 1,000 | Stop

The requirements of the projects do not exist:

create_Team_Options(datawarehouse,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)

Flags = [1, 1, 1],
Nameavailableseniors = ['Victoria Garcia', 'Renata Garcia', 'Sofia Rodriguez', 'Santiago Gonzalez', 'Daniel de la Hoya', 'Regina Quisque', 'Leonardo Di Caprio', 'Diego Mamani', 'Miguel Smith', 'Alexander Smith', 'Diego Rolle'],
Namesavailablearquitects = ['Angeles Anaya', 'Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose', 'Gilberto Ortega'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Daniel Rossi', 'Miguel Tamm', 'Fernanda de Jong', 'Petter Ivanov', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']

Next | 10 | 100 | 1,000 | Stop

The project requirements are very few.

```
create_Team_Options(qms,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)
Flags = [0, 0, 0],
Nameavailableseniors = ['Victoria Garcia', 'Renata Garcia', 'Sofia Rodriguez', 'Santiago Gonzalez', 'Daniel de la Hoya',
'Regina Quisque', 'Leonardo Di Caprio', 'Diego Mamani', 'Miguel Smith', 'Alexander Smith', 'Diego Rolle'],
Namesavailablearquitects = ['Angeles Anaya', 'Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose',
'Gilberto Ortega'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Daniel Rossi', 'Miguel Tamm',
'Fernanda de Jong', 'Petter Ivanov', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']
Next  10  100  1,000  Stop
```

The project requirements are a lot.

```
create_Team_Options(newDB,Namesavailablearquitects,Namesavailablejuniors,Nameavailableseniors,Flags)
Flags = [1, 0, 0],
Nameavailableseniors = ['Victoria Garcia', 'Renata Garcia', 'Sofia Rodriguez', 'Leonardo Di Caprio', 'Miguel Smith'],
Namesavailablearquitects = ['Angeles Anaya', 'Jonathan Cruz', 'Luis Rosado', 'Eduardo Cadena', 'Angie Contreras', 'Jose Jose',
'Gilberto Ortega'],
Namesavailablejuniors = ['John Yakimeshi', 'Cuauhtemoc Segundo', 'Pepe Tercero', 'Guadalupe Martin', 'Daniel Rossi', 'Miguel Tamm',
'Fernanda de Jong', 'Petter Ivanov', 'Ian Ramirez', 'Lucas Muecas', 'Negrito Bimbo']
Next  10  100  1,000  Stop
```
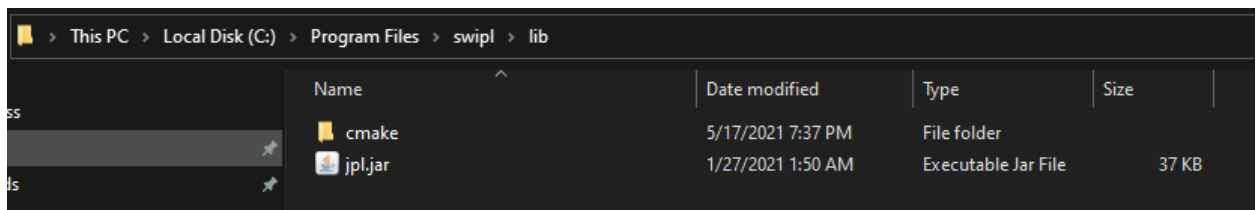
Conclusions.

It is interesting to see how prolog works to model problems that involve objects and relationships between objects, the creation and debugging of such a code is much easier if you compare it with other languages, reading is also made simple because the code is a lot shorter. One of the reasons why I use logic programming is because verifying all these rules is much easier with this paradigm, the unification, backtracking and handling of lists is simple to use, also when adding or modifying rules it is very easy to implement them and make changes to the code.
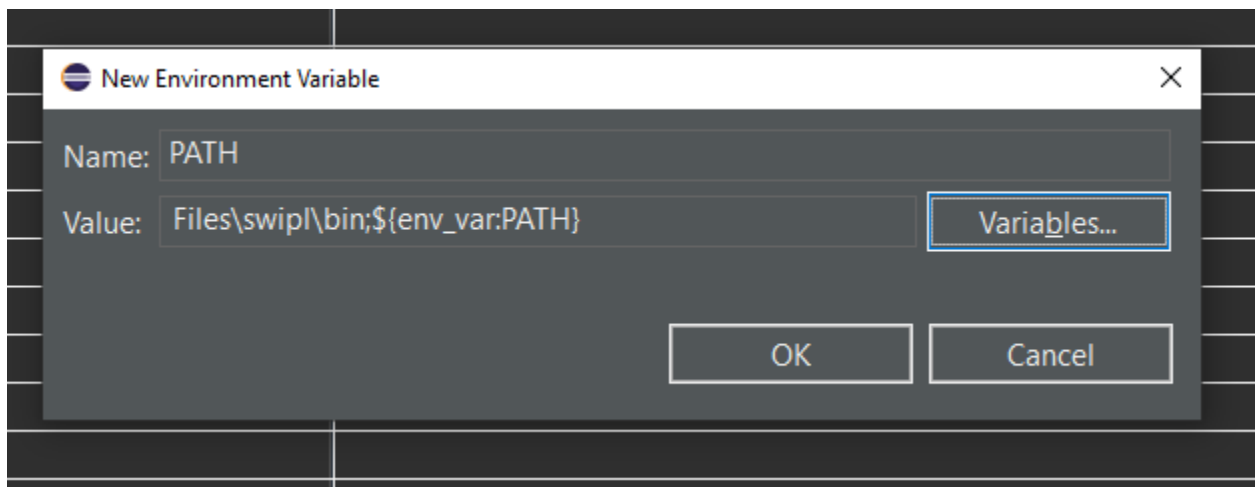
Some improvements for the project are to implement more business rules, this depends on each company in which we want to use it, it can also be to connect it with a database so as not to have to enter this data manually

Setup.

1. Install eclipse and prolog ([SWI-Prolog downloads](#) , [Eclipse Downloads | The Eclipse Foundation](#)).
2. Copy the jpl file located in github repository and paste in the folder lib, the path looks like this:

This PC > Local Disk (C:) > Program Files > swipl > lib				
Name		Date modified	Type	Size
cmake		5/17/2021 7:37 PM	File folder	
jpl.jar		1/27/2021 1:50 AM	Executable Jar File	37 KB

3. The you need to create an environment variable in java .

New Environment Variable   ×

Name: PATH

Value: Files\swipl\bin;${env_var:PATH}   Variables...

OK   Cancel

4. Add the library to build path we just created, you will added in the jpl.jar
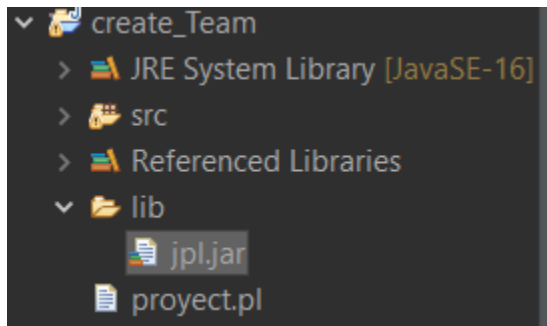
5. You are ready to run the program, just run Main Window.java

References.

Carrasco, L. (2017, February 24). Consecuencias de reclutar de forma... Retrieved
November 22, 2021, from HRTRENDS website:
https://empresas.infoempleo.com/hrtrends/consecuencias-de-reclutar-de-forma-
erronea

(2015). Retrieved November 22, 2021, from Systec.com.mx website:
https://www.systec.com.mx/post/los-factores-que-destruyen-proyectos-4-10-
asignaci%C3%B3n-equivocada-del-personal

Criterios para seleccionar a tu equipo de trabajo en un proyecto. (2015, December 24).
Retrieved November 22, 2021, from OBS Business School website:
https://www.obsbusiness.school/blog/criterios-para-seleccionar-a-tu-equipo-de-
trabajo-en-un-proyecto

PROLOG | computer language | Britannica. (2021). In *Encyclopædia Britannica*. Retrieved from https://www.britannica.com/technology/PROLOG

Learn Prolog Now! (2012). Retrieved November 22, 2021, from Let.rug.nl website: http://www.let.rug.nl/bos/lpn/lpnpage.php?pagetype=html&pageid=lpn-htmlse1