# Software Engineering for DApp Smart Contracts managing workers Contracts


Giorgia Lallai, Andrea Pinna , Michele Marchesi and Roberto Tonelli – DLT 2020 Ancona

# Summary

1. Temporary work contracts
2. The Employment Eco-system
3. System Architecture
4. Implementation
5. Conclusions

# Temporary work contract

A temporary work contract expects the work relationship to have:

- A final term
- A fixed duration.

# Temporary work contract

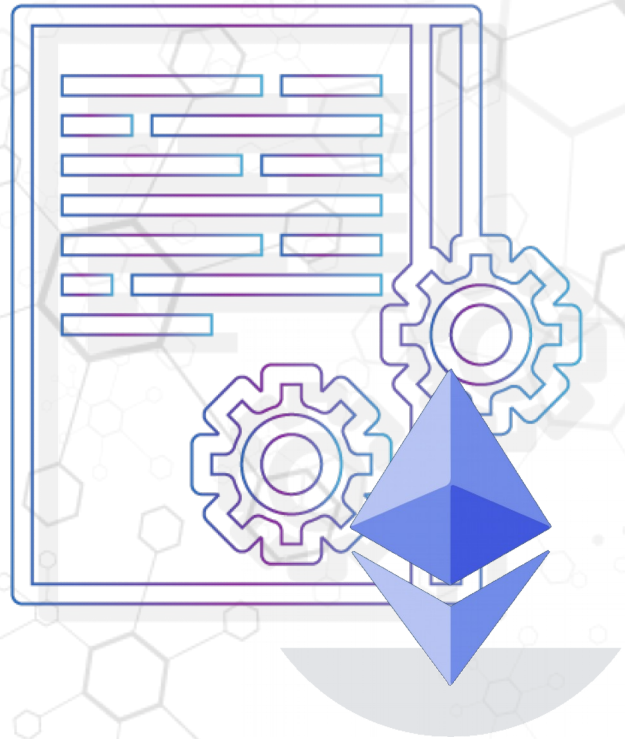
Contracts facilitate access to the world of work but are often characterized by lack of guarantees for workers

# Our Idea: Blockchain e Smart contract for Temporary Employemnt

Blockchain technology allows the rapid registration of employment contracts in accordance with legislation, for full protection of the rights of both the worker and the employer

# Blockchain and Smart contract

Work contracts can be registered in a immutable manner within the blockchain and can be read by the competent authorities in each moment in order to verify the legality



# The employment eco-system

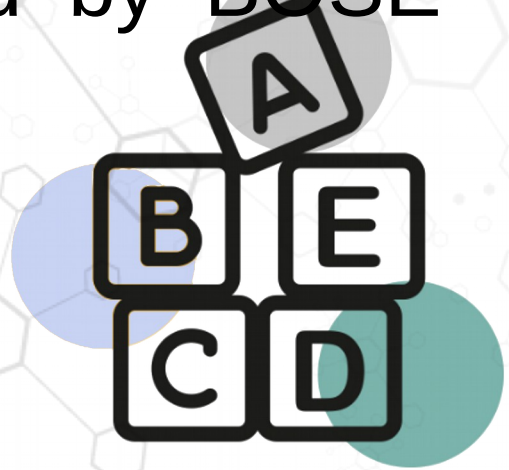
We designed and implemented a decentralized blockchain-based employment system for the management of temporary employment.

We design our system by following the ABCDE Method

# The employment eco-system

We designed our system by following the ABCDE Method.

The ABCDE methodology expects the definition of the system objective, the identifying of the system's actors, and the subdivision of the system in out-of-chain and in-chain components using the diagrams as prescribed by BOSE (Blockchain Oriented Software Engineering).



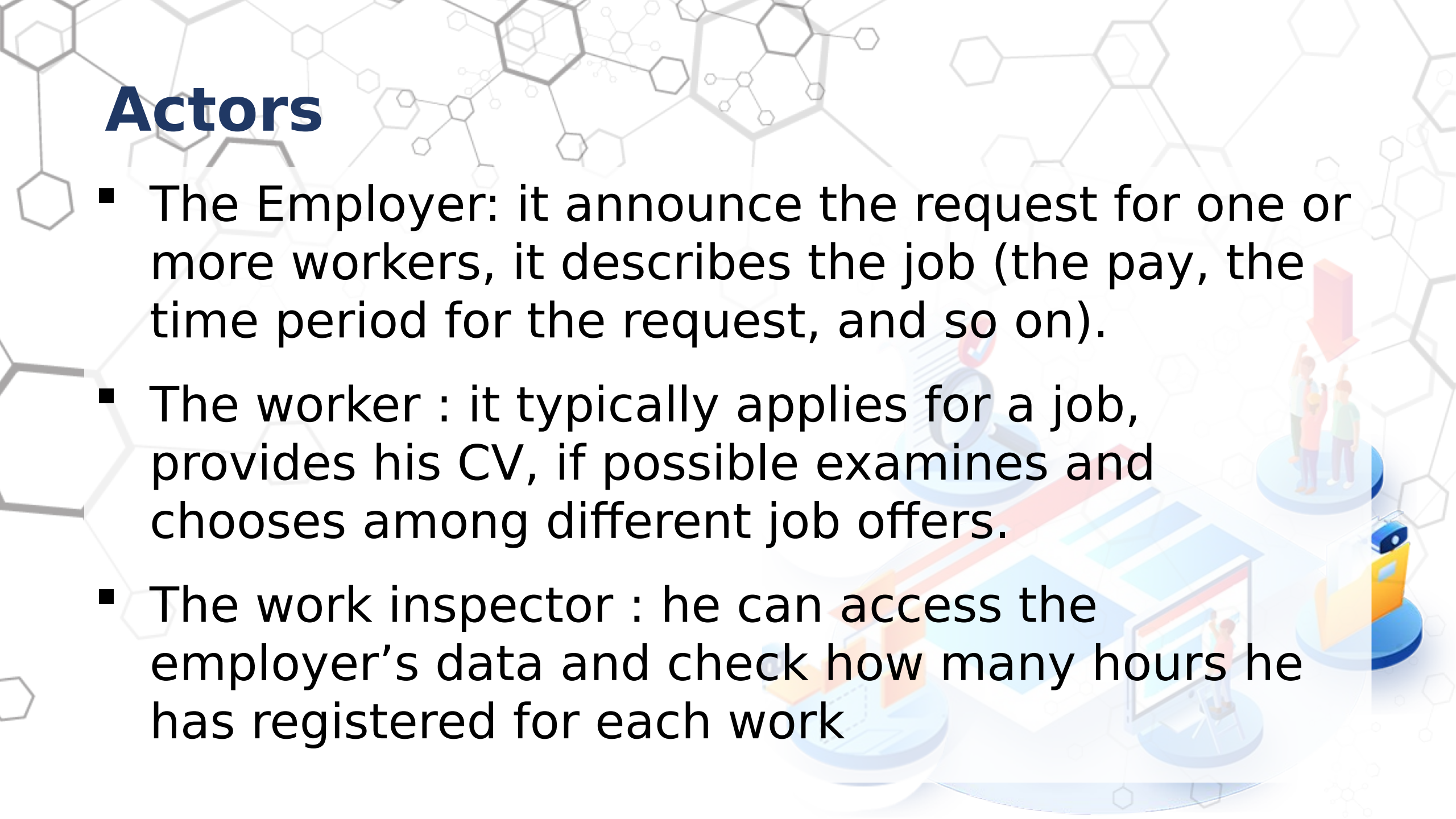
# Objectives

- Make any employment relationship implemented for temporary work transparent and traceable

- Simplify and standardize hiring procedures and prevent undeclared work

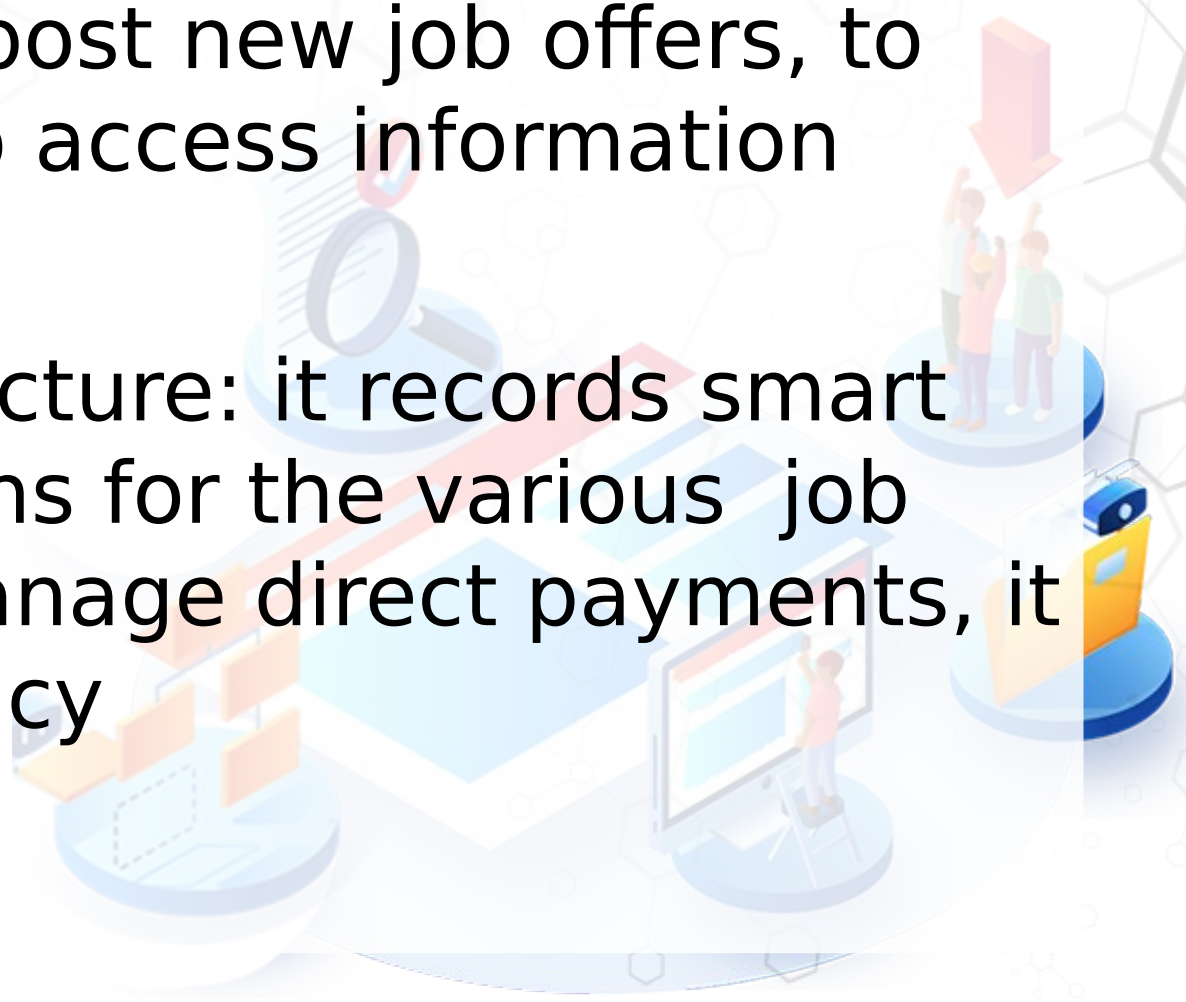
# Actors

- The Employer: it announce the request for one or more workers, it describes the job (the pay, the time period for the request, and so on).
- The worker : it typically applies for a job, provides his CV, if possible examines and chooses among different job offers.
- The work inspector : he can access the employer’s data and check how many hours he has registered for each work

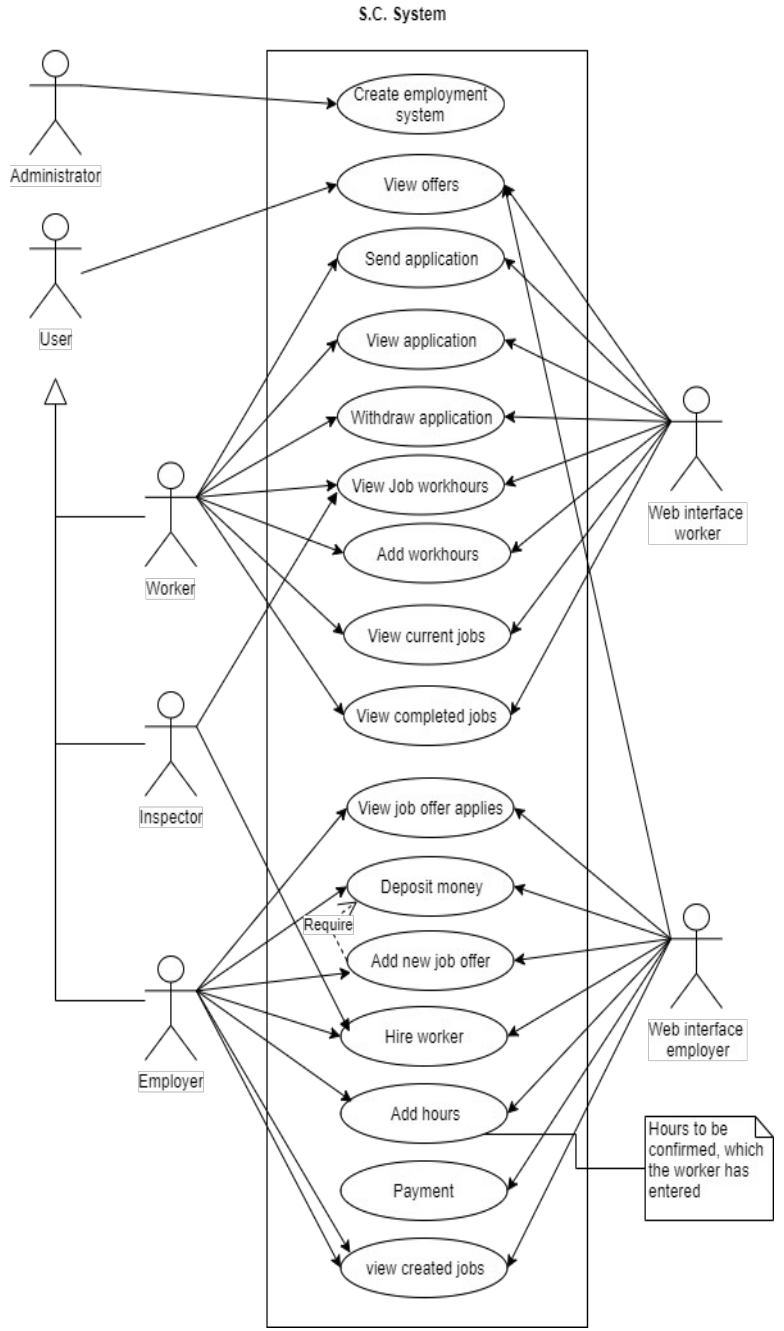


# Additional Actors (system component)

- The web platform: a simplified web platform with an interface allowing to post new job offers, to insert job candidacies, to access information about the posted jobs
- The blockchain infrastructure: it records smart contracts and transactions for the various job contracts, it allows to manage direct payments, it grants security and privacy



# UML Use Case Diagram



# Architecture

The system is composed of the following elements.

On-chain

- Smart contract *JobOfferManager*
- Smart contract *Employment*

*Out-of-chain*

- Web interface

# Smart contract: *JobOfferManager*

Represents the job offer. Implements several functions such as:

- deposit of ETH in the contract
- the creation of a new job offer
- hiring the worker
- the payment after the conclusion of the work.

# Smart contract: *Employment*

Contains all information relating to candidates, employees and the employment relationship.

Allows the worker to:

- apply for different offers,
- withdraw the application,
- request to add work hours for a specific job

# Smart contract: *Employment*

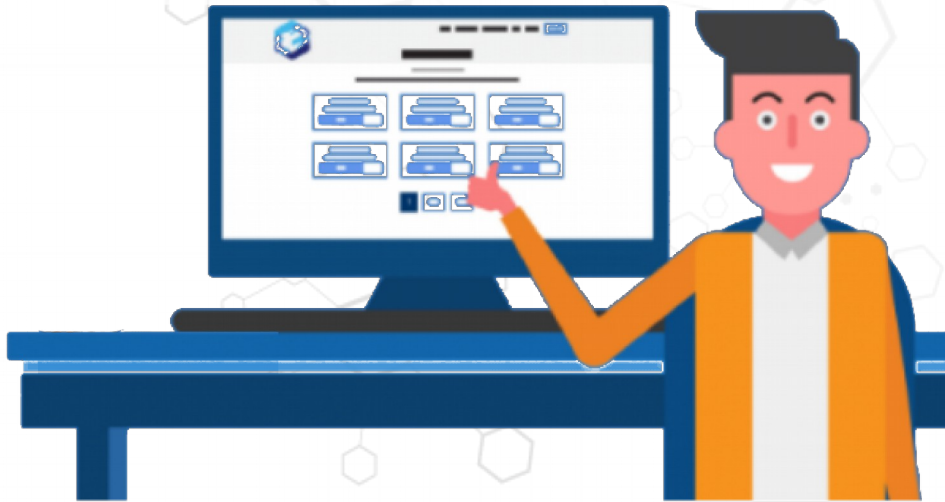
Allows the employer to:

- increase the hours worked by the employee,
- start the procedure for completing the work when the agreed hours have been reached.

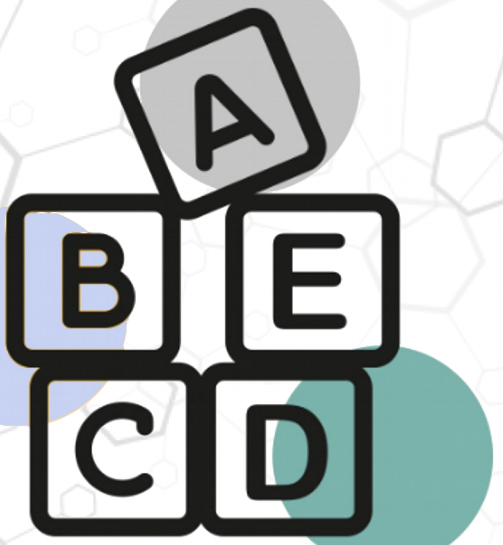
# Web interface

The web based interface:

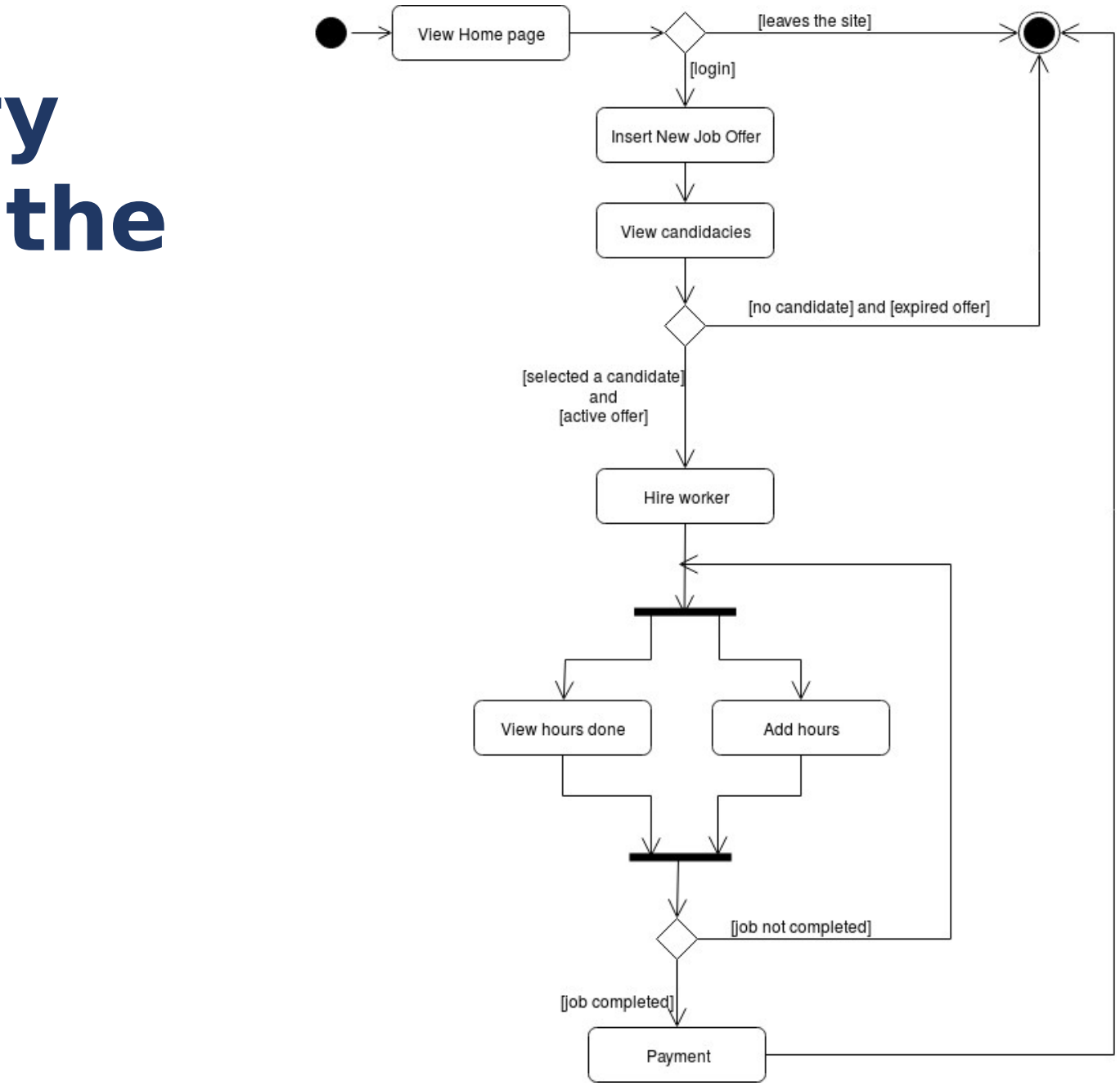
- Allows the initialisation of all the envisaged Smart contracts
- It makes it easy to both create a new job offer, apply and hire.



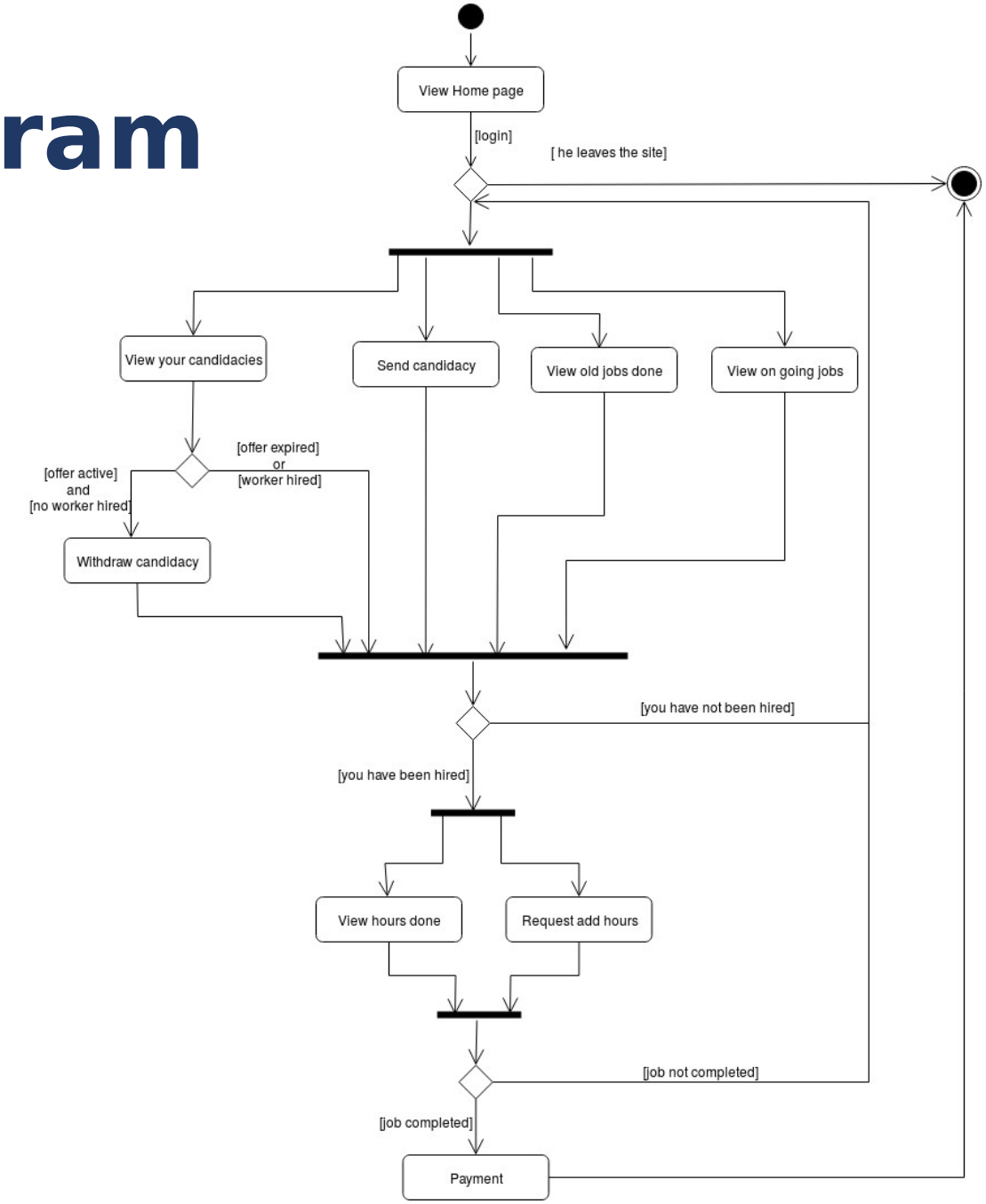
# UML diagrams for system design



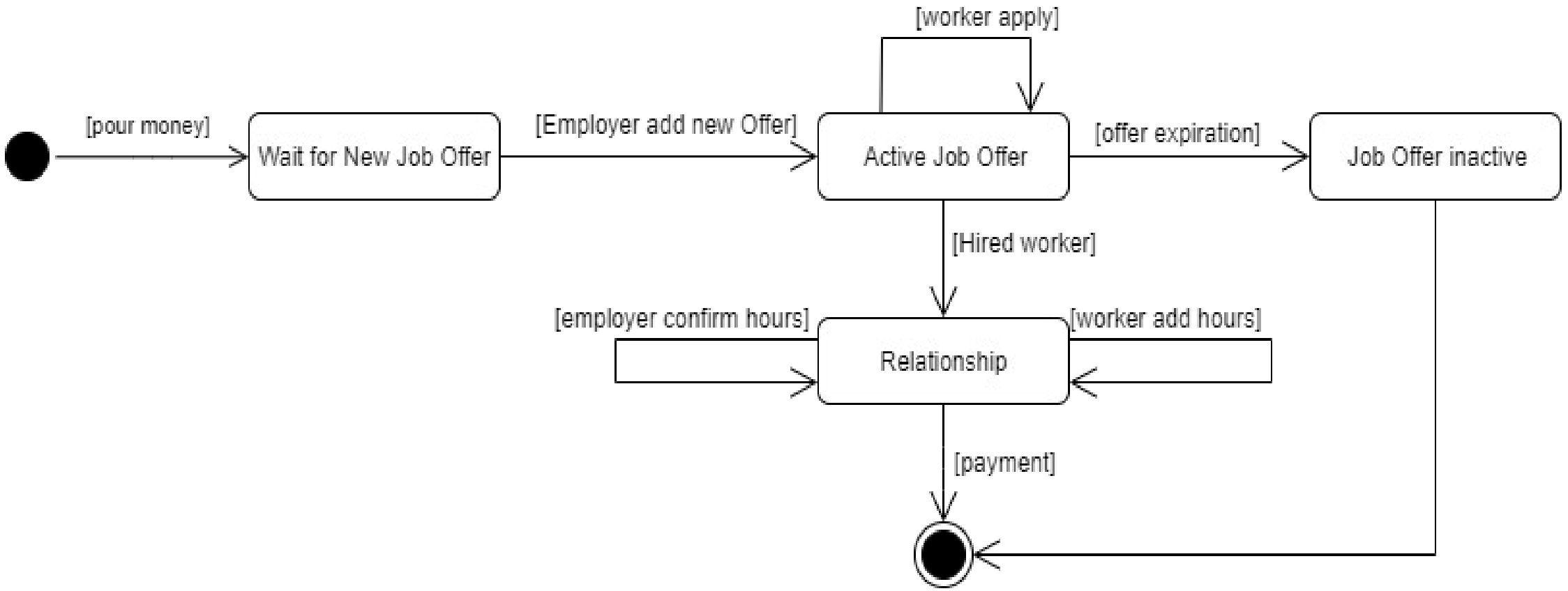
# UML Activity diagram of the Employer



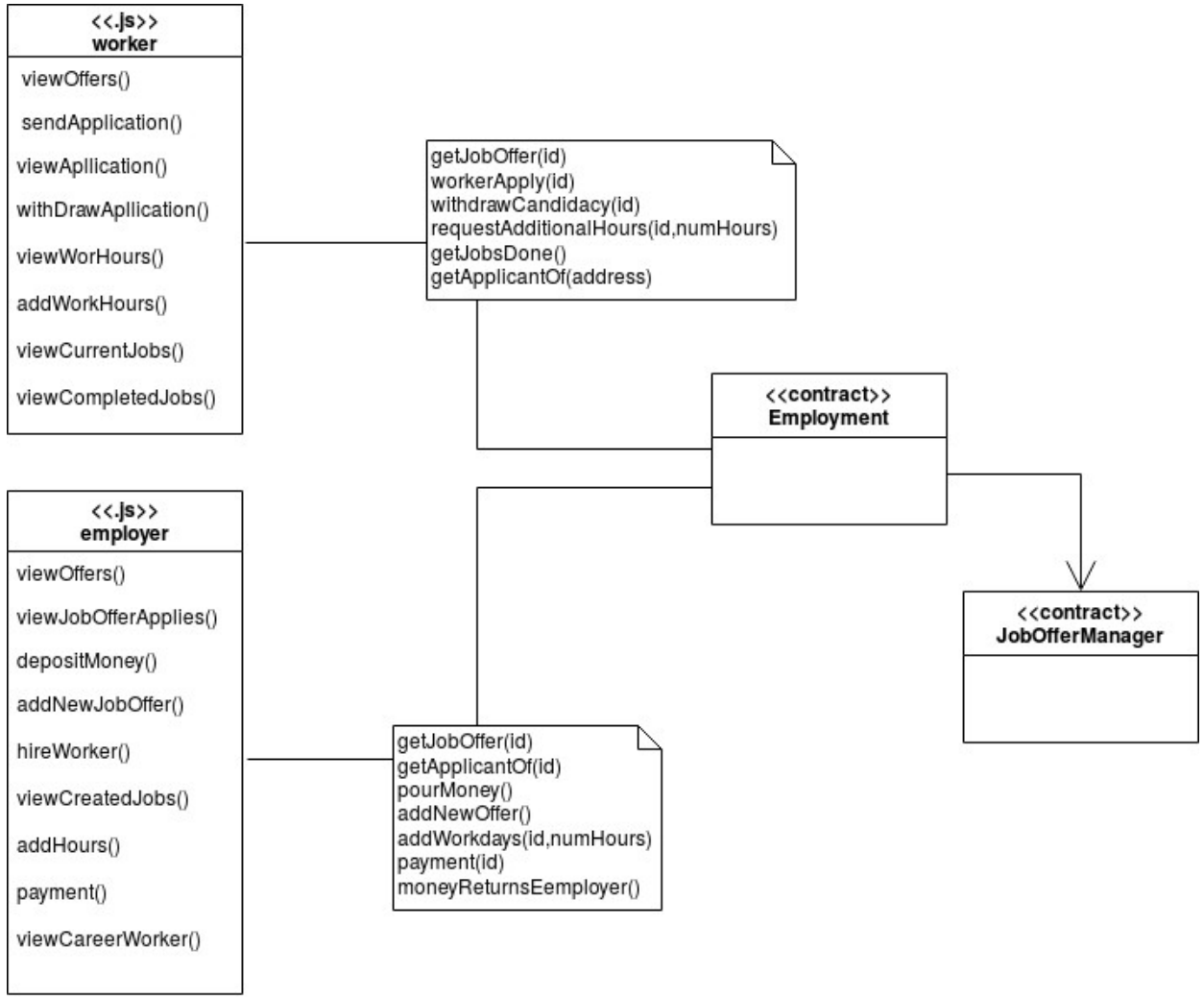
# UML Activity diagram of the worker



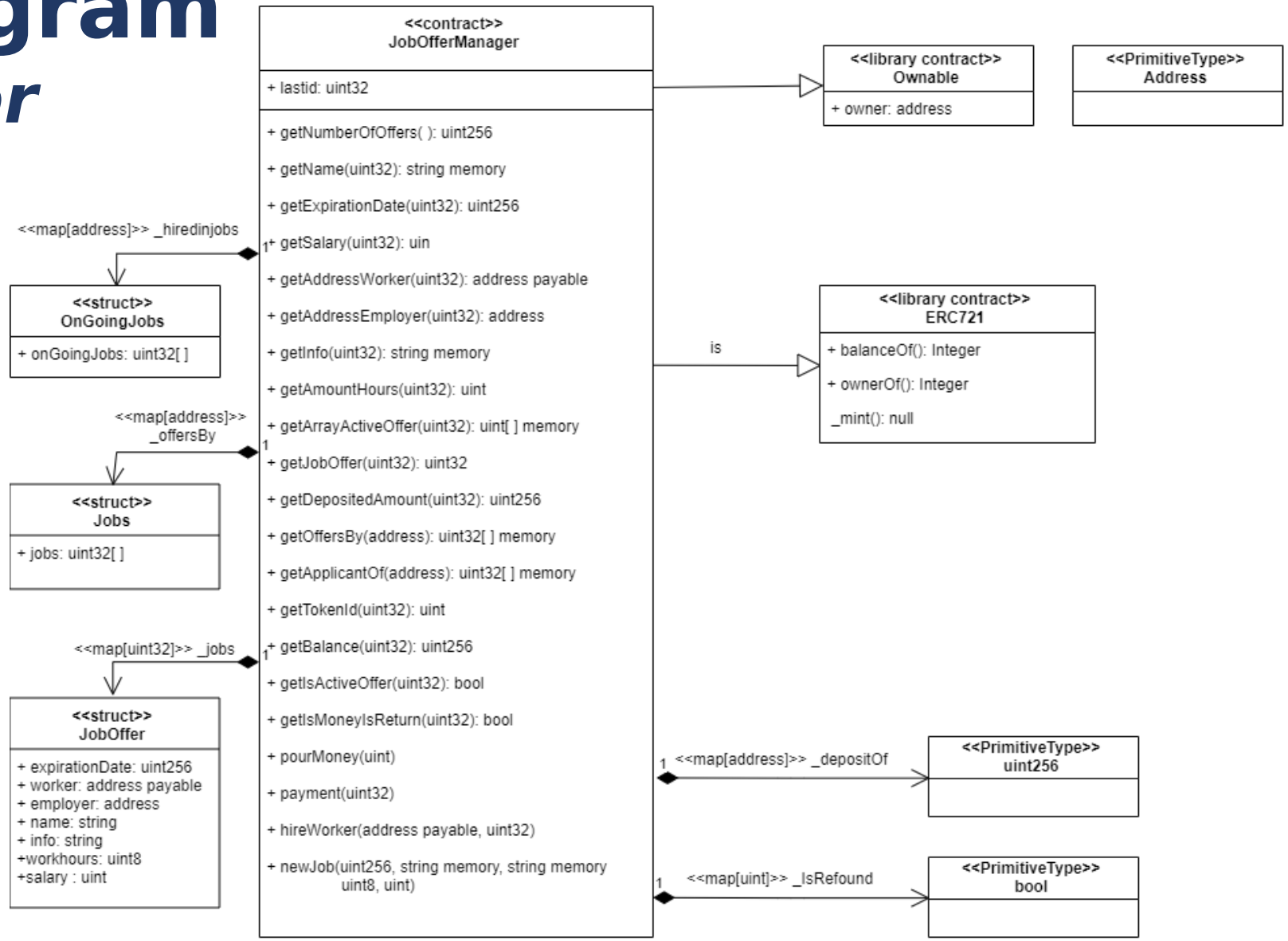
# UML Sate Diagram



# Class diagram

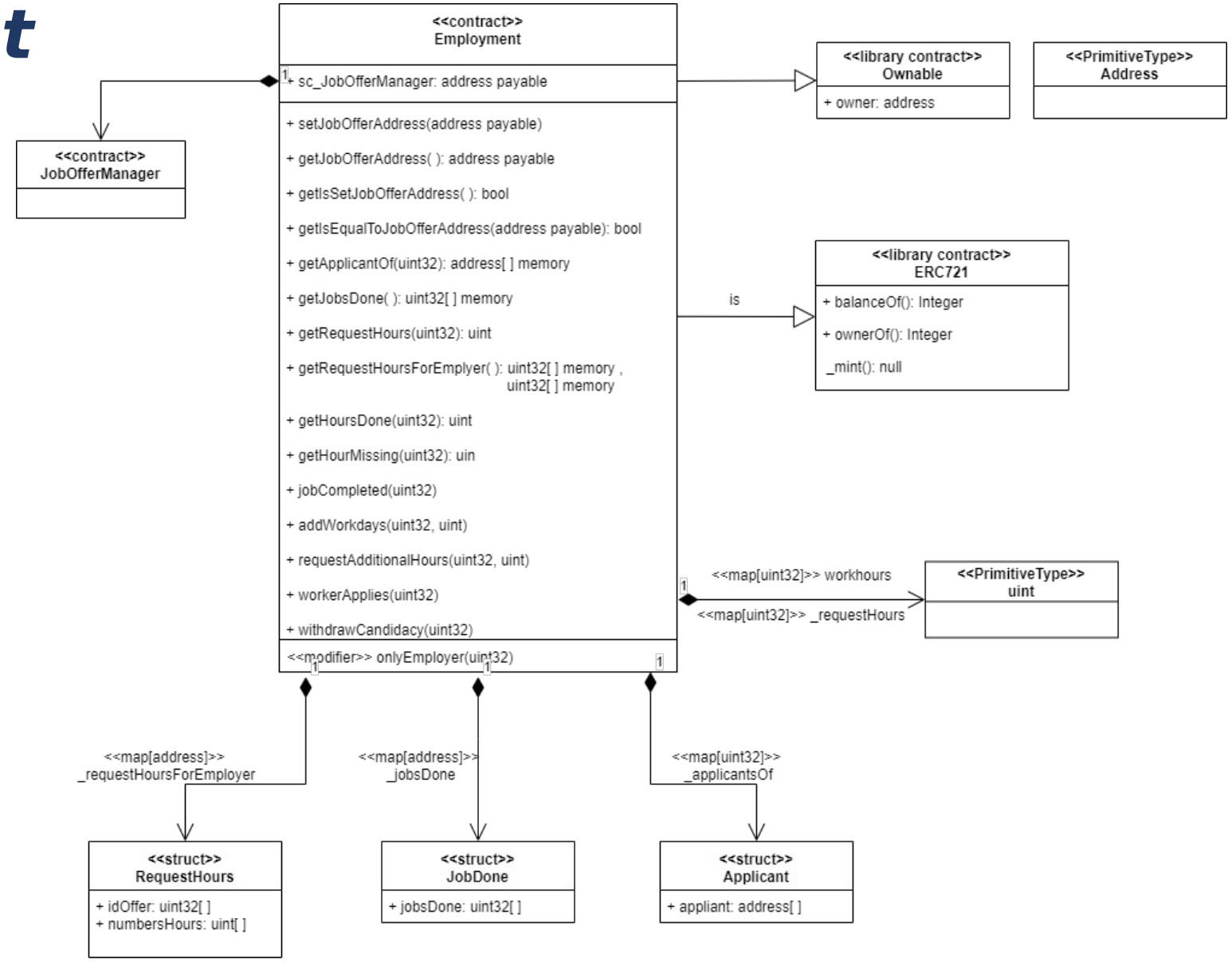


# Contract diagram
## *JobOfferManager*



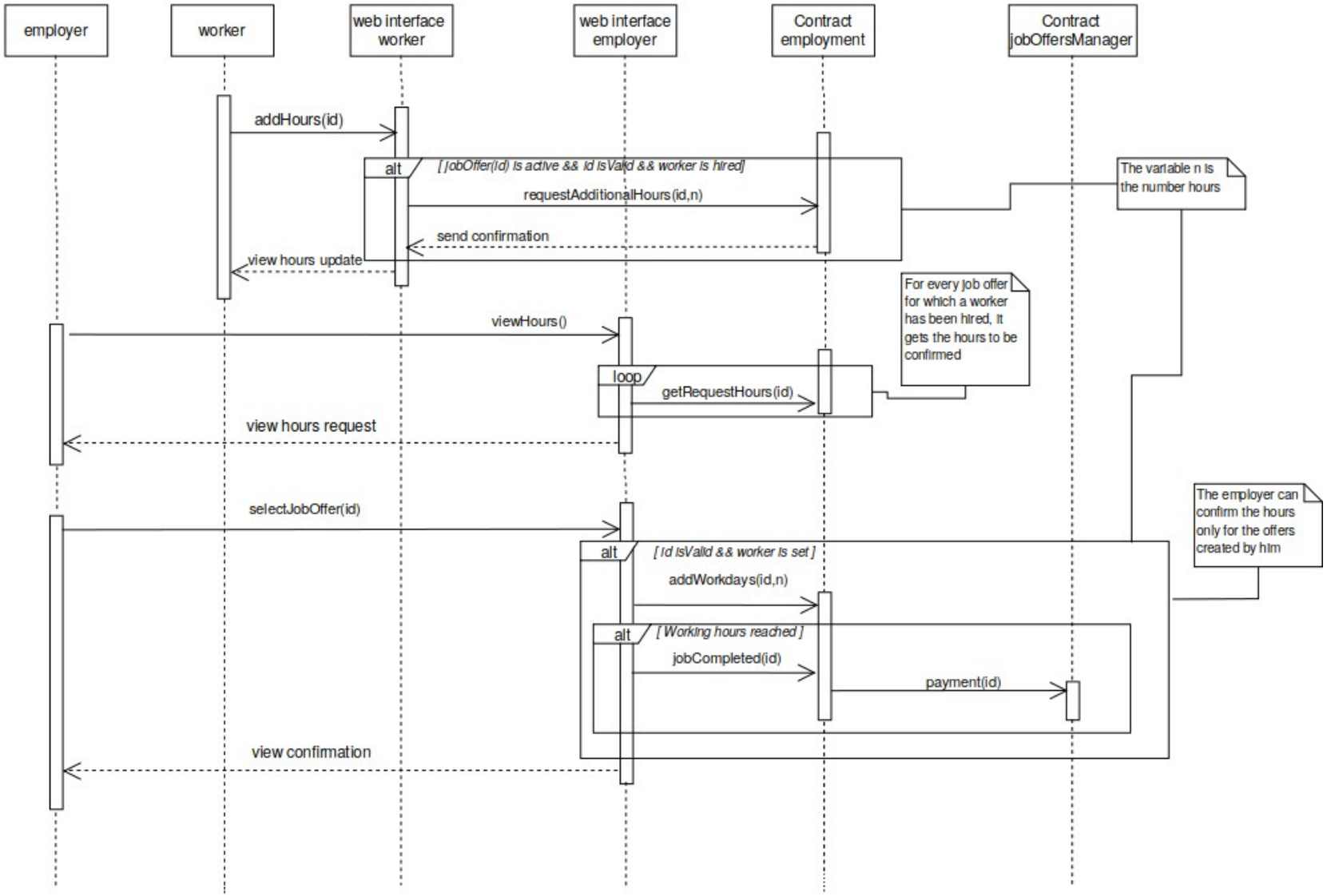
# Contract diagram

## *Employment*



# Sequence diagram

## *job applicati*



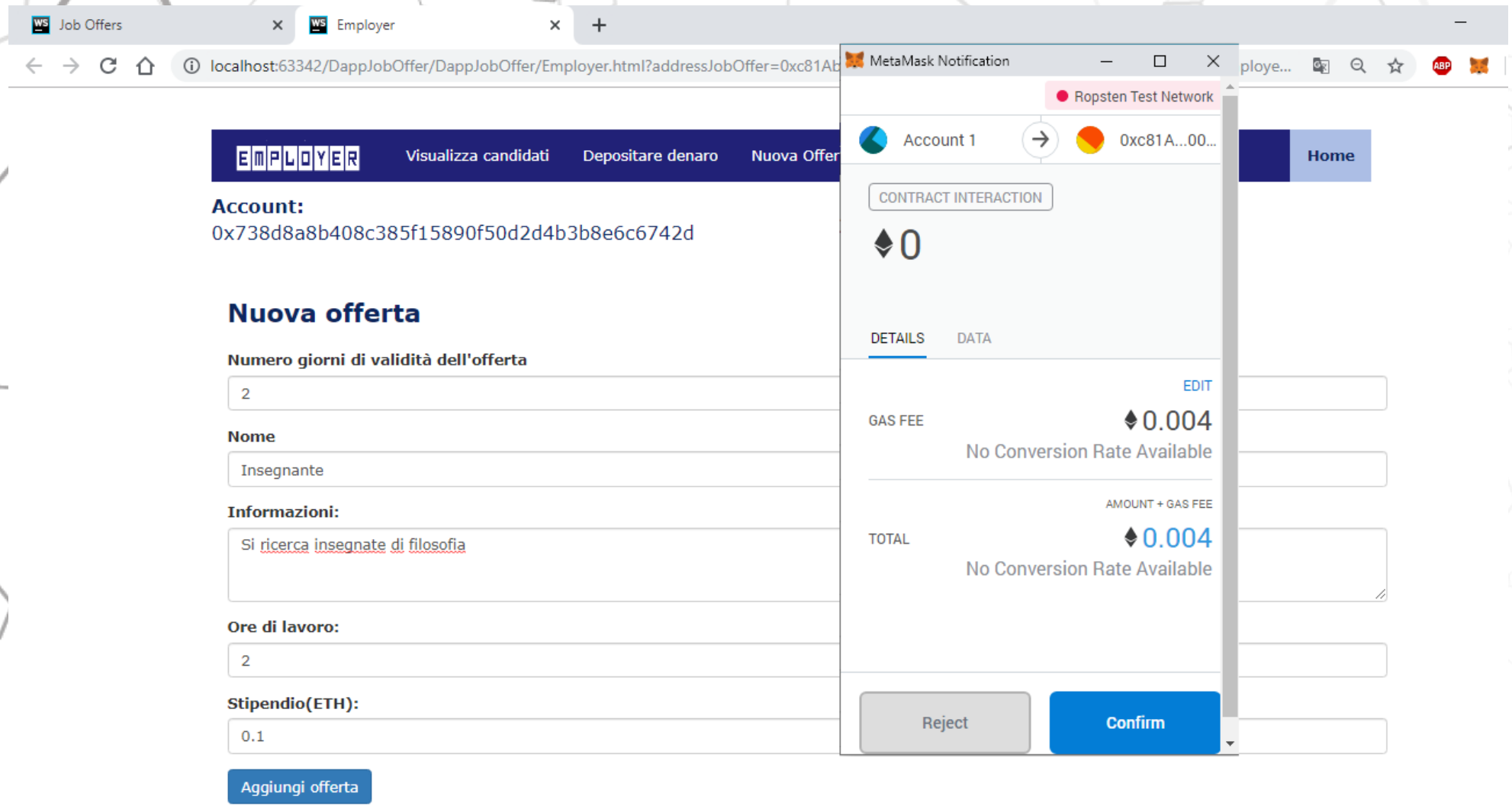
# The prototype

We developed the Solidity code for the Smart Contracts and built the DApp system which provides the users with a user friendly web interface enabling the implementation of all the features described.

# The prototype

The web interface uses “metamask”, a bridge to run Ethereum DApps right in your browser with out running a full Ethereum node, for providing the communication channel between Dapp and blockchain. We deployed the Smart Contracts on the Ropsten test net in order to test our prototype under all working conditions.

# The prototype



# Conclusions

In this case study we applied the BOSE and ABCDE methodology to devise a DApp for managing temporary employments so that by design the employers and the employees are able to easily identify roles, constraints, commitments in the specific domain.

# Conclusions

This approach allows to build a DApp software product in which all the requirements and features are determined and recovered by the diagrams adopted by the methodology so that Smart Contracts variables and ABI are quickly and precisely identified.

# Conclusions

The approach reduce risks of failure since in-chain and out-of-chain components are identified by design since the very beginning, and smart contract structure and interactions are well defined before the software development. We show how the approach successfully guided us to produce a working prototype for managing the case study of the temporary employments.

# Software Engineering for DApp Smart Contracts managing workers Contracts

# Thank you!

Giorgia Lallai, Andrea Pinna , Michele Marchesi and Roberto Tonelli – DLT 2020 Ancona