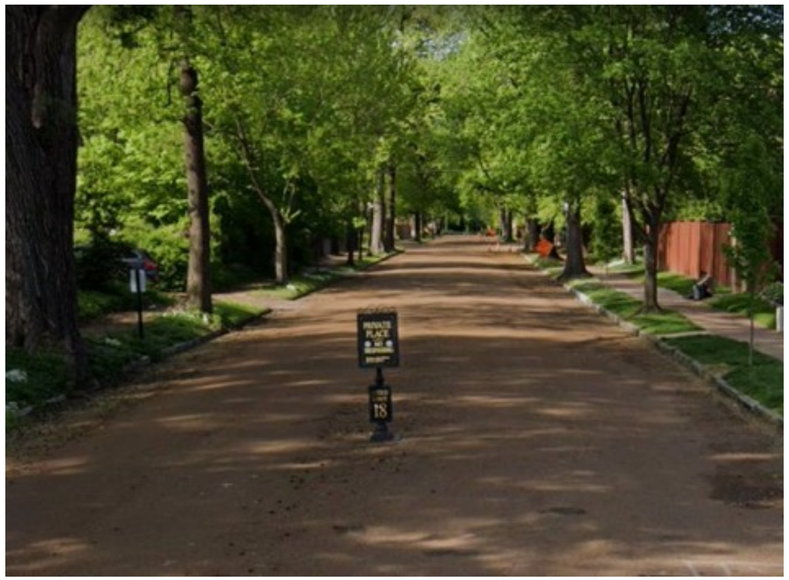
NAMIBIA UNIVERSITY
OF SCIENCE AND TECHNOLOGY



back to agency

when autonomy meets automation

José G. Quenum

inaugural lecture

Nov. 2023

- agent (softbot, knowbot, taskbot, etc.)
  - autonomy
  - heterogeneity
  - perception
  - action
  - proactivity?

- multi-agent systems (MAS)
  - collection of agents
    - share knowledge
    - communicate with each other to solve complex problems beyond the scope of individual agents

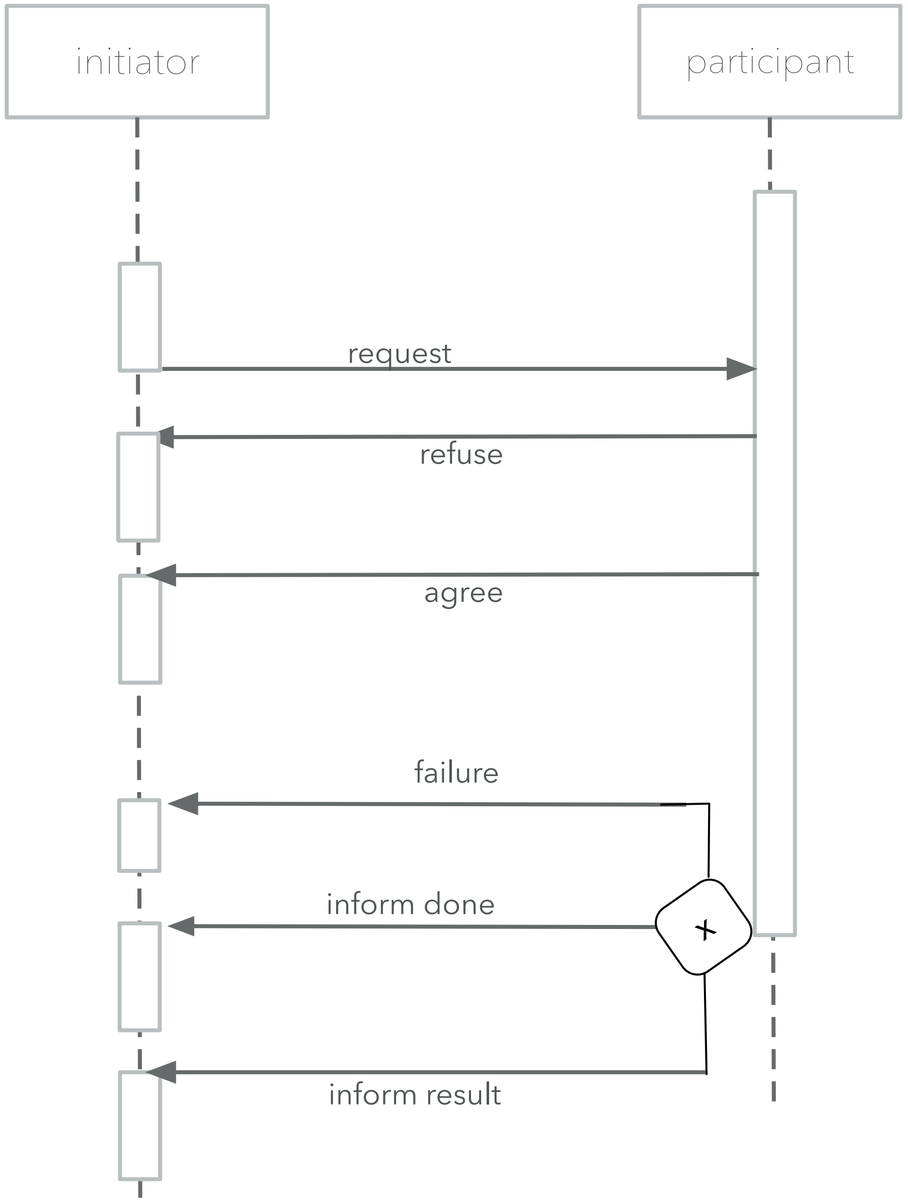

RECYCLING GRADES
LD CLEAR
HD FILM
PET BOTTLES
MIXED PAPER - CARD
WHITE PAPER HI-J
LD COLOUR

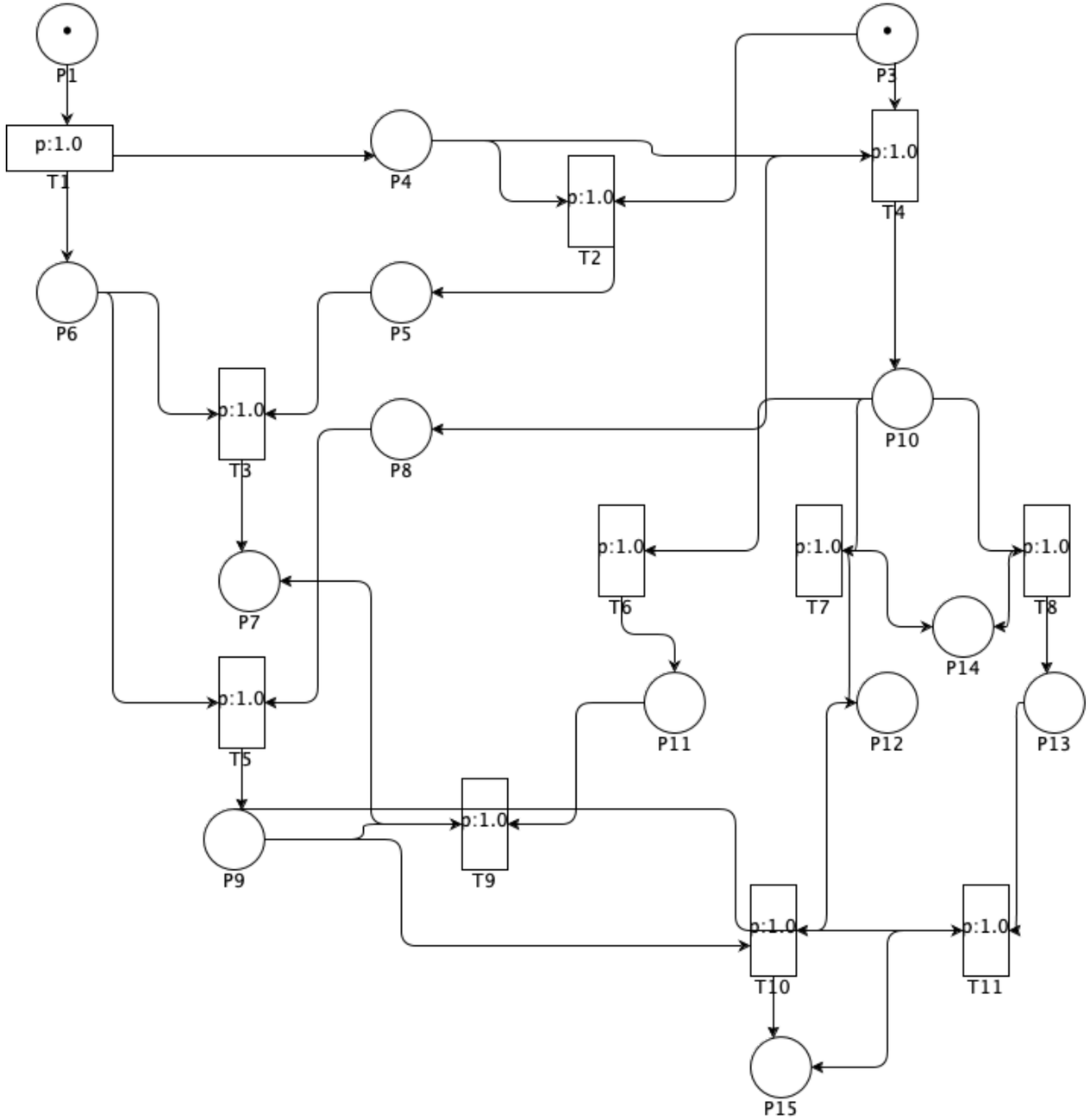

FOOD WASTE FOR COMPOSTING
CONTAMINATED WASTE - LANDFILL
PAPER
PLASTIC, CANS & GLASS

outline

- coordination in MAS
- adaptation to SOC
- smart infrastructure

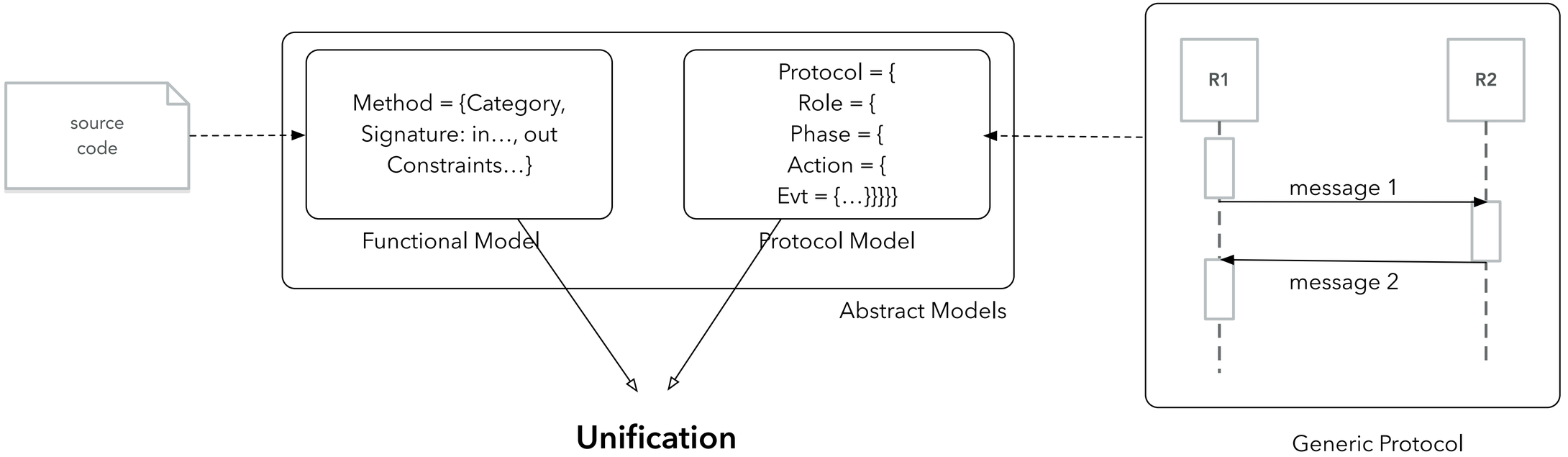
agent coordination


initiator
participant
request
refuse
agree
failure
inform done
inform result


P1
P3
p:1.0
T1
P4
p:1.0
T2
p:1.0
T4
P6
P5
p:1.0
T3
P8
P10
P7
p:1.0
T6
p:1.0
T7
p:1.0
T8
P14
p:1.0
T5
P11
P12
P13
P9
p:1.0
T9
p:1.0
T10
p:1.0
T11
P15

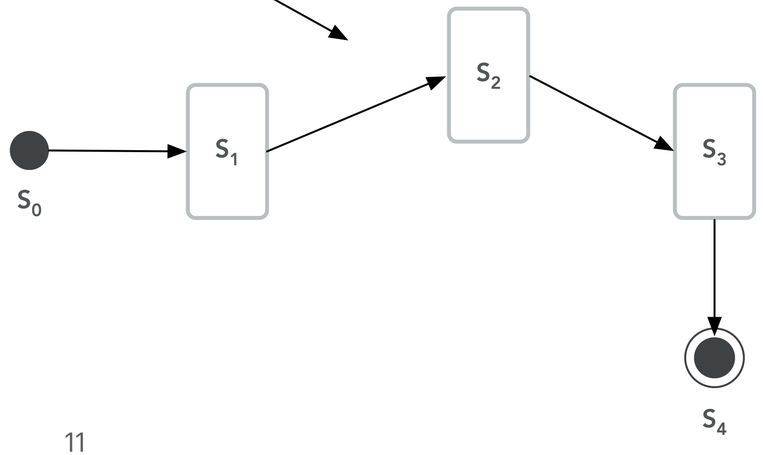
- automatic protocol configuration
  - [Quenum and Aknine, IJAOSE, 2010; Quenum, IAT 2007; Quenum IAT 2008]
- protocol selection
  - [Quenum and Aknine, CoRR 2005, Quenum and Aknine, CEEMAS 2005, Quenum et al., IAT 2006, Quenum et al. ICWS 2007]

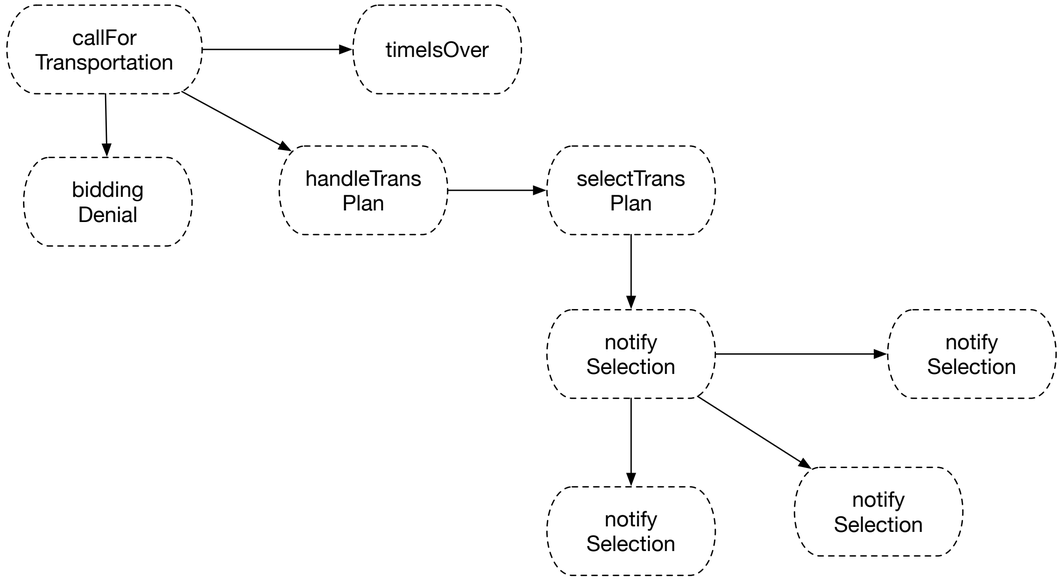
automatic protocol configuration

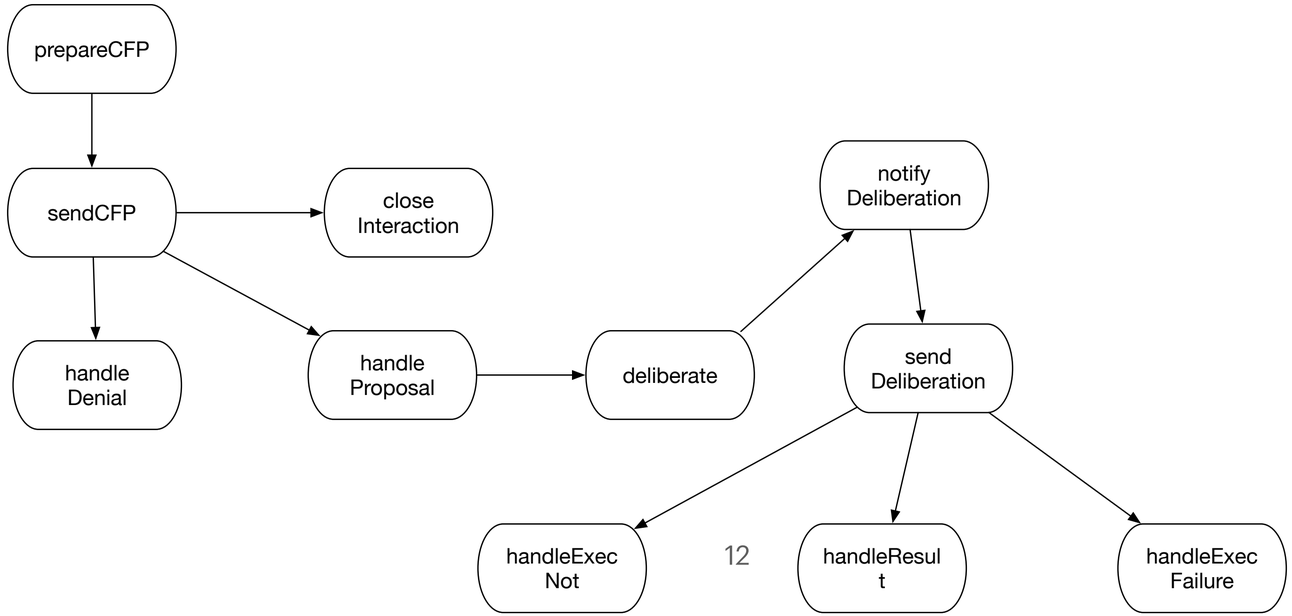


Unification

$$a \xrightarrow{unif} m \Leftrightarrow (\nu = \nu') \wedge (\text{Valid\_Instance}(\Sigma', \Sigma) \wedge (\text{Infer\_Constraints}(E, C)$$




callFor Transportation
timeIsOver
bidding Denial
handleTrans Plan
selectTrans Plan
notify Selection
notify Selection
notify Selection
notify Selection


prepareCFP
sendCFP
close Interaction
handle Denial
handle Proposal
deliberate
notify Deliberation
send Deliberation
handleExec Not
handleResul t
handleExec Failure

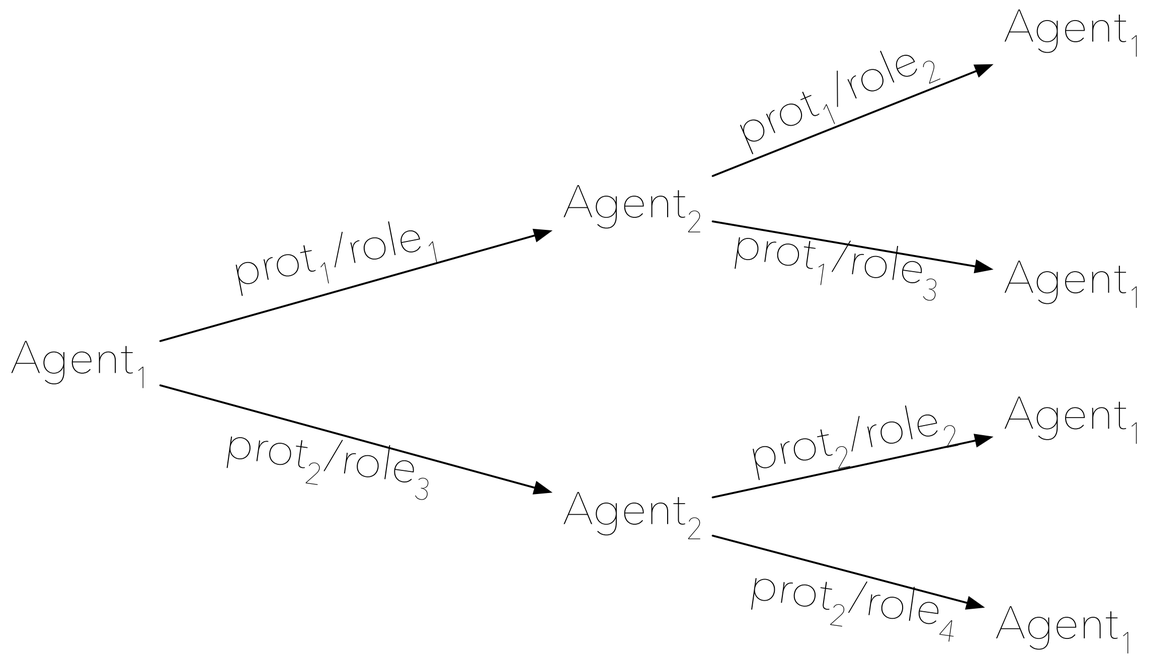
[dynamic] protocol selection

$$(\mathbf{t}, \mathscr{A}) \vdash \{\mathbf{r}_i\}$$

joint selection

trust among agents

- perfect information
- negotiation based on optimality
  - equilibrium found with backward induction
  - protocol compatibility



individual selection

lack of trust

- imperfect information
- repeated simultaneous games
  - rollback in case of error
  - protocol compatibility

		$Agent_2$	
		$prot_1/role_2$	$prot_2/role_2$
$Agent_1$	$prot_1/role_1$	P_{1A}, P_{2A}	P_{1A}, P_{2B}
	$prot_2/role_3$	P_{1B}, P_{2A}	P_{1B}, P_{2B}

in short

- automate protocol configuration
- enable agents to select the interaction protocols

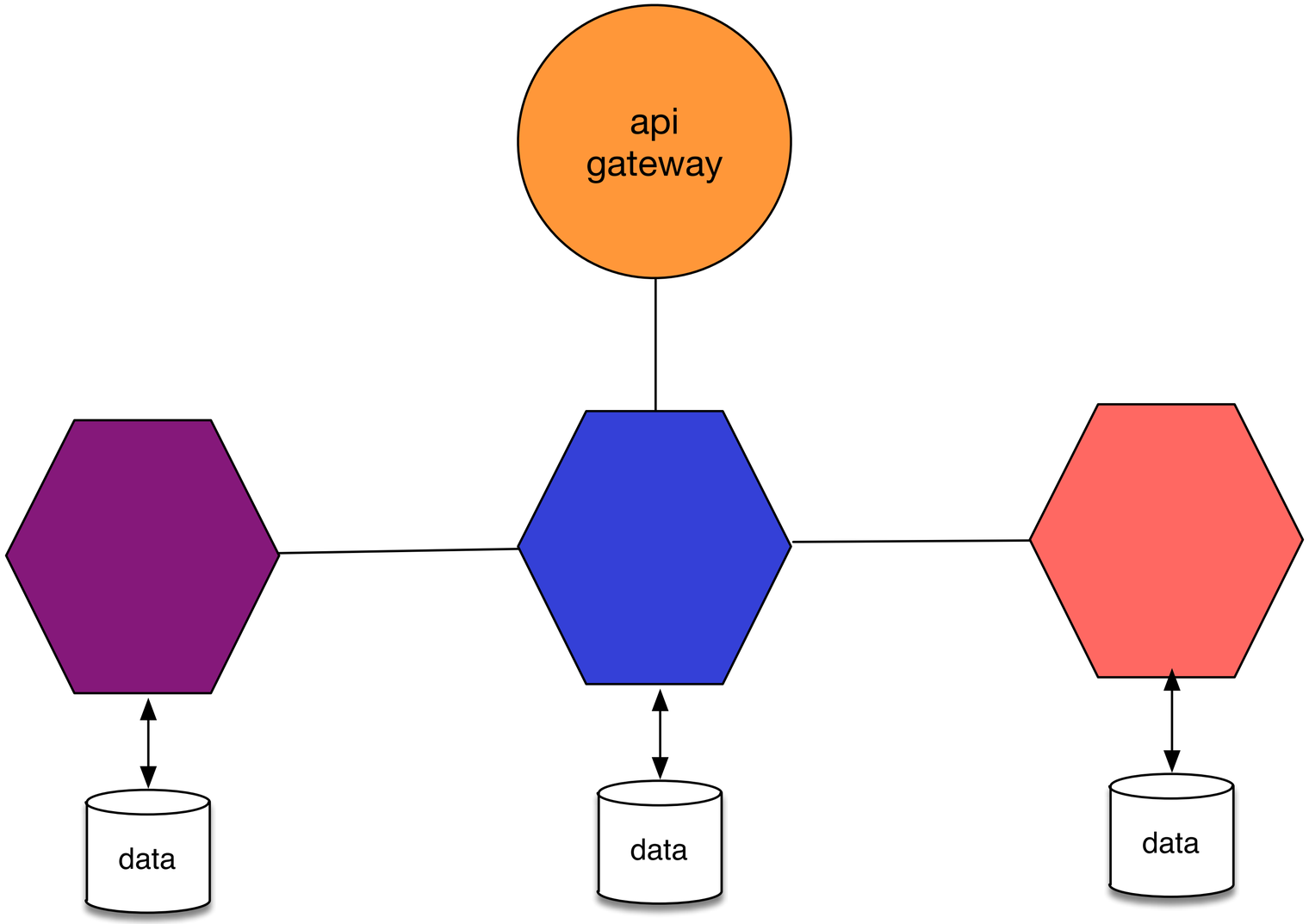
agents in SOC

- service **selection** and **composition**
  - [Quenum et al., ICWS 2007, Lumala and Quenum, Services 2009]

- micro-services and serverless
  - executable specifications for micro-services [Quenum and Aknine, SCC, 2018]
  - multi-cloud serverless function composition [Quenum and Josua, UCC, 2021]
  - containerisation and orchestration abstraction for cloud-native applications [Quenum and Ishuuwa, Cloud 2020]

executable specifications

- fine-grained interfaces
  - independent units with single responsibility principle
- business-driven development
- multiple computing paradigms
  - polyglot programming and persistence strategy


api
gateway
data
data
data

- micro-services and intelligent agents are
  - situated
  - goal-oriented (reactive agents)
  - autonomous

workflow agent

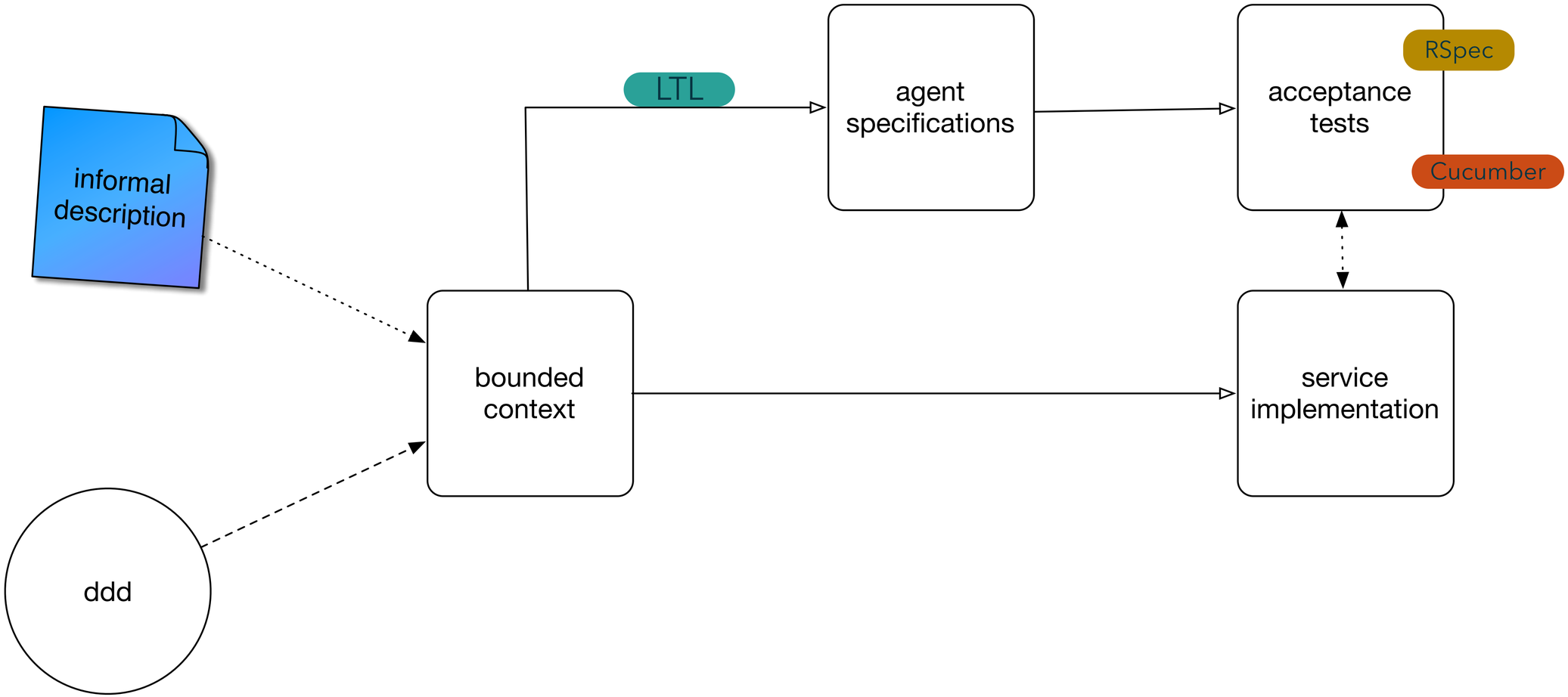
$$\Box(\rho \vee \phi)$$

$$\rho \stackrel{def}{=} \text{"}instantiate\ a\ new\ workflow\text{"}$$

$$\rho \equiv receive(e) \wedge (\forall \mathrm{w}_i \in \mathcal{U}, \neg bound(\mathrm{w}_i, \mathrm{e}) \implies \circ(createw(e)))$$

$$\phi \stackrel{def}{=} \text{"}consume\ events\ bound\ to\ a\ specific\ workflow\text{"}$$

$$\phi \equiv receive(e) \wedge (\exists! \mathrm{w}_j \in \mathcal{U}, bound(\mathrm{w}_j, \mathrm{e}) \wedge \dot{\mathrm{S}}_j \xrightarrow{e} \mathrm{S}_\ell \implies \diamond(cons(e)))$$


informal description
LTL
agent specifications
acceptance tests
RSpec
Cucumber
bounded context
service implementation
ddd

in short...

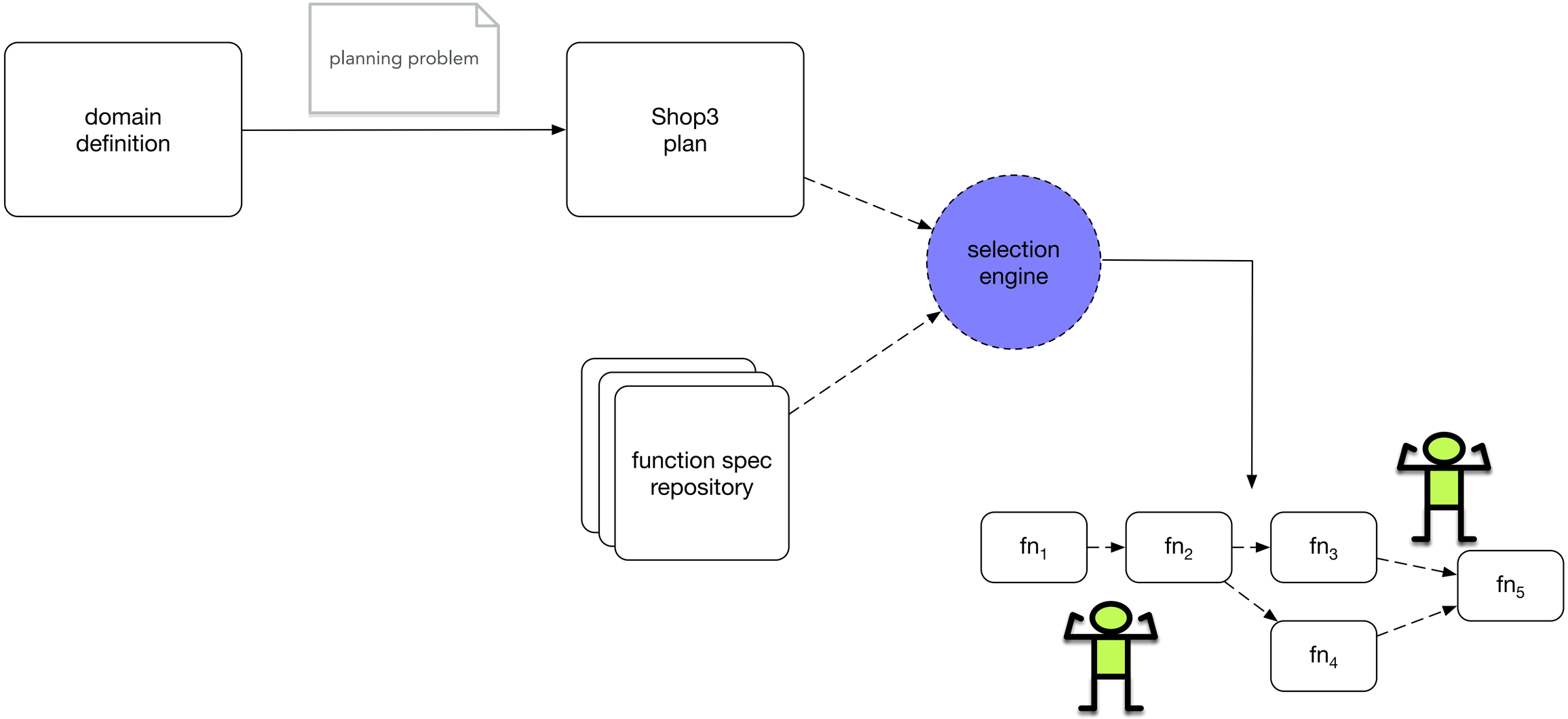
- combination of formal specifications and automated testing
- in the future
  - new DSL
  - integrated testing engines

serverless function composition

serverless computing

function as a service

- individual, ephemeral, event-driven **functions**
- highly scalable, available, fault-tolerant cloud **infrastructure**
- e.g.,
  - AWS lambda
  - Azure functions
  - openwhisk


domain definition
planning problem
Shop3 plan
selection engine
function spec repository
fn1
fn2
fn3
fn4
fn5

planning

- hierarchical task network (htn)
- shop3/pddl
  - domain definition
  - planning problem

```
(defdomain banking-example (
      (:op (!create_acc ?acc ?anumber)
       :precond (((not (existsacc ?acc)) (not (matchesn ?acc ?anumber))))
       :delete (((not (existsacc ?acc)) (not (matchesn ?acc ?anumber))))
       :add ( (existsacc ?acc) (matchesn ?acc ?anumber) (balance ?acc 0)))
```

```
(:method (transact ?a5 ?a6 ?anum5 ?anum6 ?depo_amount ?
tr_amount) () ((init_depost ?a6 ?anum6 ?depo_amount) (!
create_acc ?a5 ?anum5) (pay ?a5 ?anum5 0 ?tr_amount) (
withdraw ?a6 ?anum6 (call + 0 ?depo_amount) ?tr_amount)))
```

```
(defproblem trpb2 banking-example
  (((not (existsacc A3)) (not (existsacc A4)) (not (matchesn A3 251219)) (not (matchesn A4
291441)))) ((transact A3 A4 251219 291441 3000000 25000)))
```

```
------------------------------------------------------------------------
Problem #<SHOP3::PROBLEM TRPB1> with :WHICH = :ALL, :VERBOSE = :LONG-PLANS

Totals: Plans Mincost Maxcost Expansions Inferences  CPU time  Real time
           1     2.0     2.0          6          6     0.054      0.054

Plan trees:
((((TRANSFER A1 A2 312436 354789 7200 12500 4000)
   ((PAY A1 312436 7200 4000) (1.0 (!CREDIT_ACC A1 7200 4000) 0))
   ((WITHDRAW A2 354789 12500 4000) (1.0 (!DEBIT_ACC A2 12500 4000) 1)))))

------------------------------------------------------------------------
Problem #<SHOP3::PROBLEM TRPB2> with :WHICH = :ALL, :VERBOSE = :LONG-PLANS

Totals: Plans Mincost Maxcost Expansions Inferences  CPU time  Real time
           1     5.0     5.0         11         23     0.007      0.007

Plan trees:
((((TRANSACT A3 A4 251219 291441 3000000 25000)
   ((INIT_DEPOST A4 291441 3000000) (1.0 (!CREATE_ACC A4 291441) 0)
    ((PAY A4 291441 0 3000000) (1.0 (!CREDIT_ACC A4 0 3000000) 1)))
   (1.0 (!CREATE_ACC A3 251219) 2)
   ((PAY A3 251219 0 25000) (1.0 (!CREDIT_ACC A3 0 25000) 3))
   ((WITHDRAW A4 291441 3000000 25000) (1.0 (!DEBIT_ACC A4 3000000 25000) 4)))))
```

fn discovery and selection

$$\text{select}(\phi_\ell) \cdot \phi_\ell \in \Delta \quad \text{s.t. } \phi_\ell \sim \mathrm{a}_l \cdot \mathrm{a}_l \in \pi$$

- translation engine
  - generate
    - signature, pre and post conditions for operators
  - implemented in Julia

- match making
  - input-output/pre-conditions-effects (IOPE, see Ngan and Kanagasabai, 2013, Personal and Ubiquitous Computing Vol 17; Stollberg et al., ESWC 2007)
  - use decorator to assist in type equivalence

constraint satisfaction solver (1)

$$p = < X, D, C >$$

$X = \{x_\iota \cdot x_\iota$ is an action in the plan$\}$

$$D = \bigcup_{x_\iota} D_{x_\iota};\ D_{x_\iota} = \{\phi_\ell \cdot \phi_\ell \sim x_\iota\}$$

$$C = C^{u}_{x_\iota} \cup C^{b}_{x_\iota, x_J} \text{ with } x_\iota, x_J \in X \cdot \iota \neq J$$

constraint satisfaction solver (2)

- backtrack search
  - implemented in Julia
- qos requirements
  - latency
  - response time
  - throughput
  - success rate

composition

- composition refinement
  - enhance the plan with composition operators
    - sequence; parallel, conditional, loop
- generate execution plan

- orchestration
  - composition coordinator
  - agents coordinate with cloud platforms
  - fault-tolerance and error recovery

experiment

- banking domain
  - three functions: *create*; *increase*; *decrease*
  - implemented using serverless tool (https://github.com:joques/carthago)
  - implementation in node.js
  - deployed on **openwhisk**

```
serverless invoke --function create --data '{"account_number":
332146, name: "Mary Pickford", address: "Infinite Loop"}'
```

```
{result: "Ok!", account: {account_number: 332146, name: "Mary
Pickford", address: "Infinite Loop", balance: 0}}
```

in short...

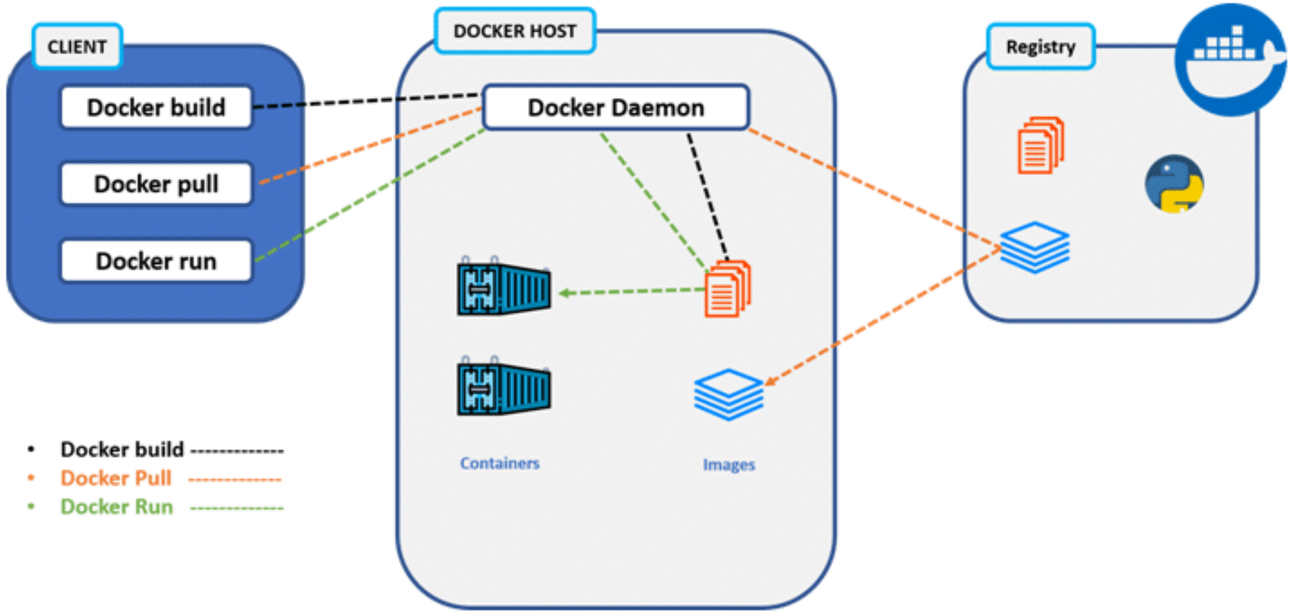
- composition and selection of serverless
  - hierarchical **planning**
  - **translation engine** and **constraint satisfaction**
- more
  - orchestration with **continuation-passing** style
  - more generic functions
  - type **inference** and type **equivalence**

velo 1.0

Dockferfile.dockerfile

Users > gervasius > Downloads > Dockferfile.dockerfile > ...

```
FROM node:latest
RUN apt update
RUN apt install -y git
RUN git clone http://196.216.167.206:3000/gervasius/usermanager.git
WORKDIR /usermanager
RUN npm install
EXPOSE 3000
CMD ["npm", "start"]
```



Welcome | docker-compose.yml

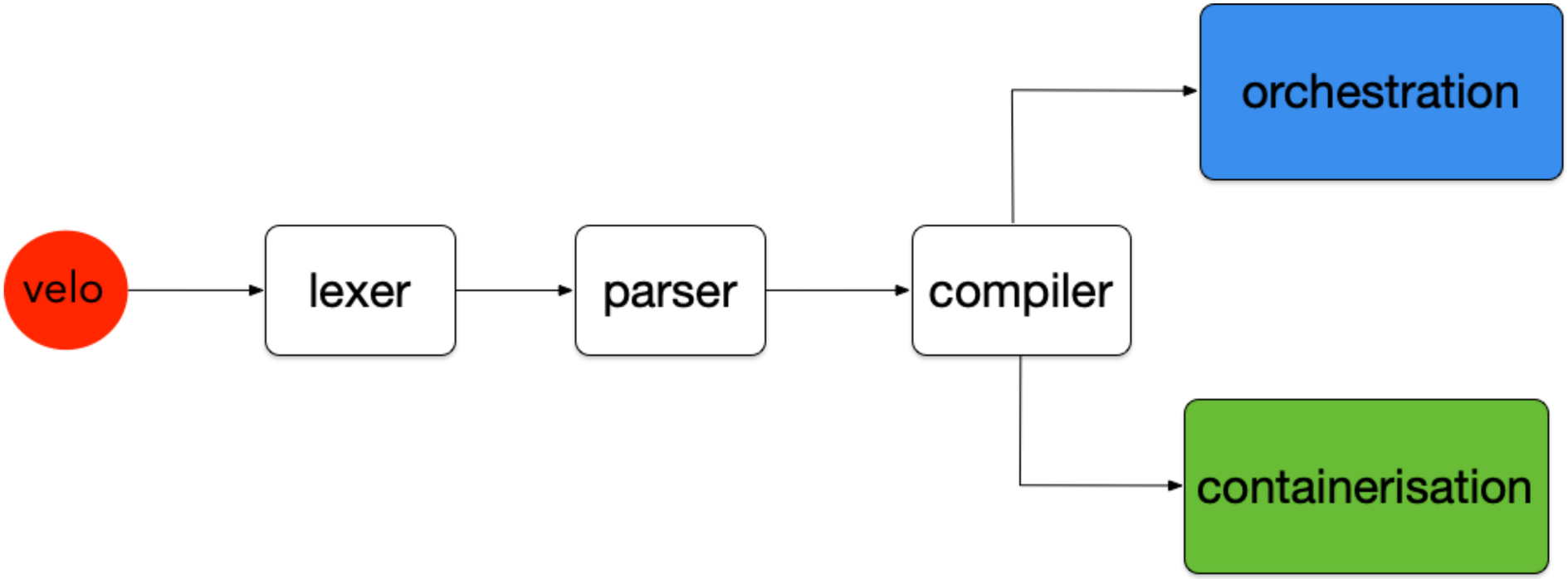
Users > gervasius > Downloads > docker-compose.yml > version

```
version: '3'
services:
  app:
    container_name: usermanager
    image: usermanager:latest
    restart: always
    ports:
      - '3000:3000'
    links:
      - mongo
  mongo:
    container_name: mongo
    image: mongo
    volumes:
      - ./data:/data/db
    ports:
      - '27017:27017'
```

- lack of **unifying** high-level abstraction
- no portability or inter-operability
- no mechanism to **enforce** best practices

velo 1.0

a domain-specific language to abstract containerisation and orchestration



the dsl

- key concepts
  - container
  - virtual bag
    - single or multiple
- mode
  - rich or light

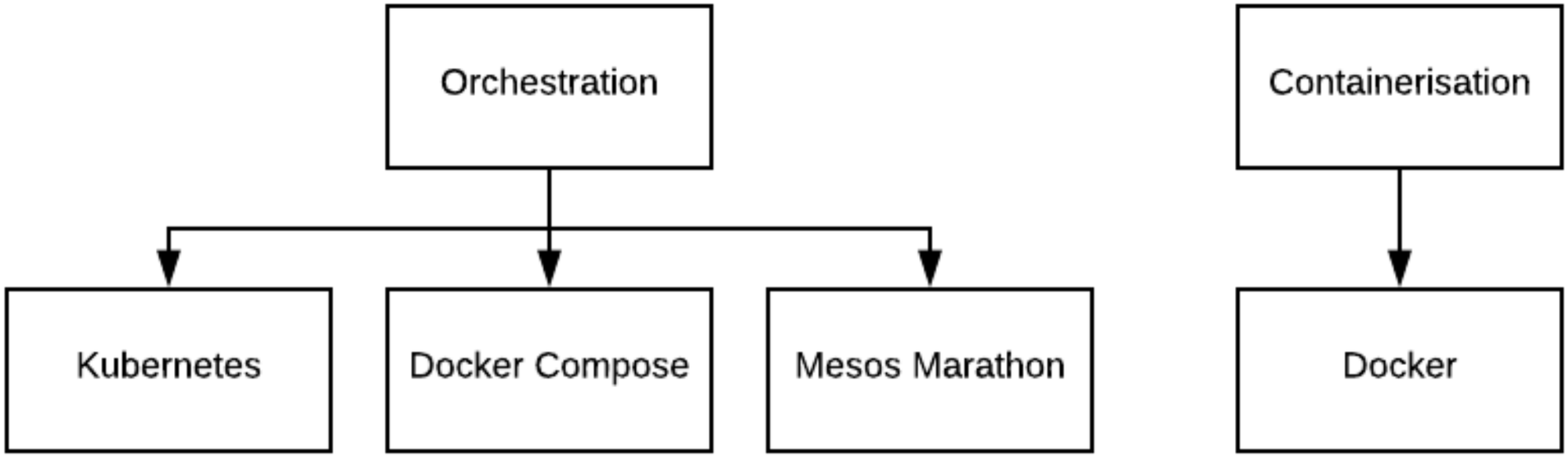
⟨*velo-app*⟩ ::= ⟨*velo_metadata**⟩ ⟨*orchestration*⟩
⟨*velo_metadata*⟩ ::= ⟨*author*⟩ | ⟨*title*⟩
⟨*author*⟩ ::= 'author' ⟨*string*⟩
⟨*title*⟩ ::= 'title' ⟨*string*⟩
⟨*orchestration*⟩ ::= ⟨*vbag*⟩
⟨*vbag*⟩ ::= ⟨*svbag*⟩|⟨*mvbag+*⟩
⟨*svbag*⟩ ::= ⟨*v_metadata**⟩ ⟨*labels?*⟩ ⟨*container*⟩
⟨*mvbag*⟩ ::= ⟨*v_metadata**⟩ ⟨*labels?*⟩ ⟨*container+*⟩

https://github.com:joques/velo-spec.git/grammar/Velo.g4

⟨*v_metadata*⟩ ::= ⟨*id*⟩ | ⟨*inst*⟩ | ⟨*nets*⟩ | ⟨*auth*⟩ | ⟨*version*⟩
⟨*labels*⟩ ::= ⟨*label+*⟩
⟨*label*⟩ ::= ⟨*label_name*⟩ '=' ⟨*string*⟩
⟨*label_name*⟩ ::= ⟨*string*⟩
⟨*container*⟩ ::= ⟨*c_metadata**⟩⟨*image*⟩⟨*labels?*⟩⟨*resources?*⟩
⟨*network*⟩⟨*volume*⟩
⟨*c_metadata*⟩ ::= ⟨*name*⟩ | ⟨*policy*⟩ | ⟨*auth*⟩
⟨*policy*⟩ ::= 'restart' '=' ⟨*policy_val*⟩
⟨*policy_val*⟩ ::= 'always' | 'on-failure' | 'no' | 'unless-stopped'
⟨*image*⟩ ::= ⟨*labels?*⟩ ⟨*commands*⟩

the transpiler

- source-to-source compiler
- automatically generate target container and orchestration specs


Orchestration
Containerisation
Kubernetes
Docker Compose
Mesos Marathon
Docker

in short...

- syntax testing
  - lexer and parser distinguish correct and incorrect rule sets
- compiler testing
  - the generated ast is correct
- valid description
  - generated files yield the expected containerisation and orchestration behaviour

in the future

- support more container tools
- make the language leaner and the compiler smarter
- provide direct integration with languages

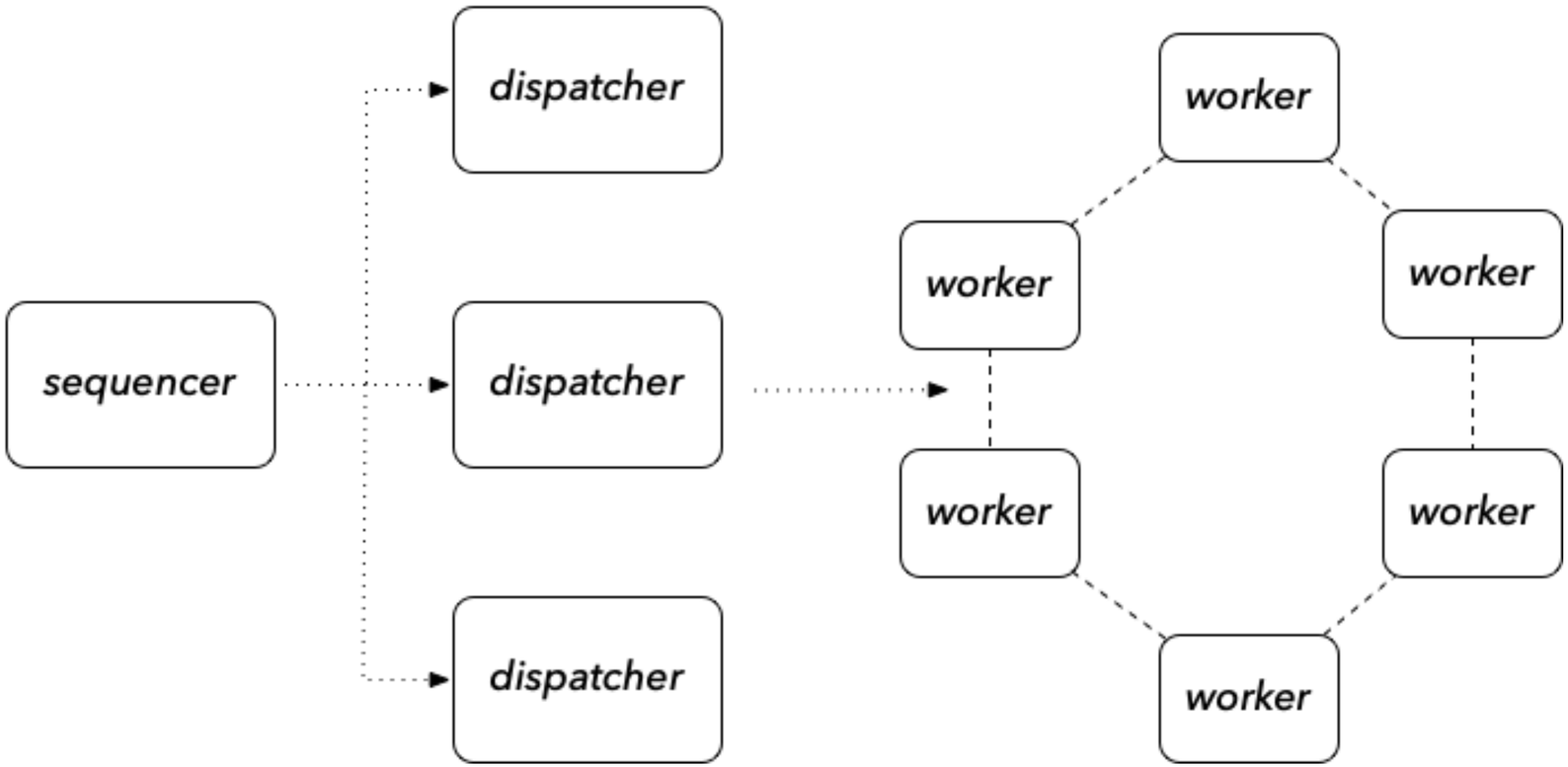
smart infrastructure

p2p storage engine for schemaless data — TaYo

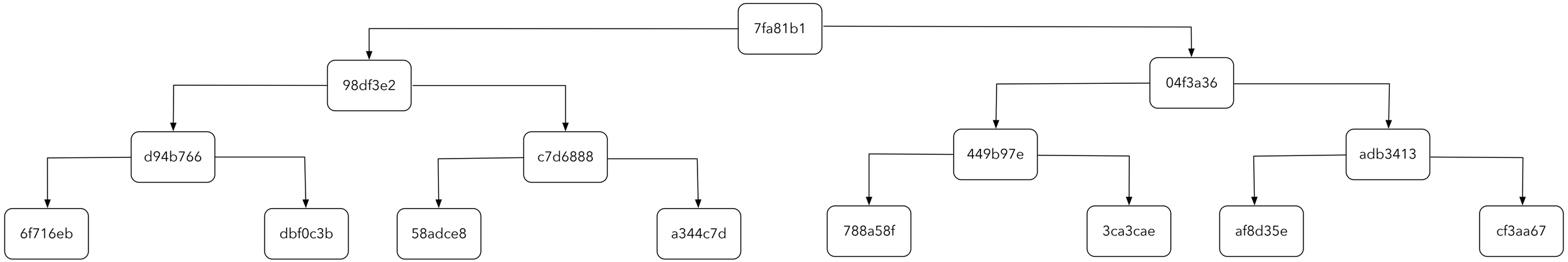
properties

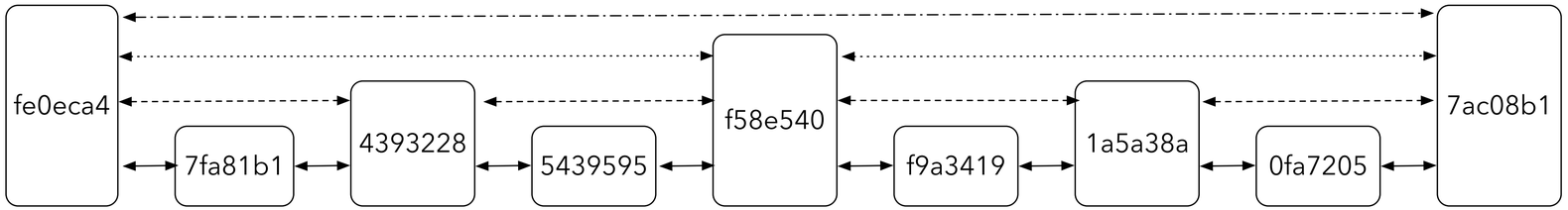
- membership and coordination
- replication and consensus
- consistency
- fault-tolerance

the architecture

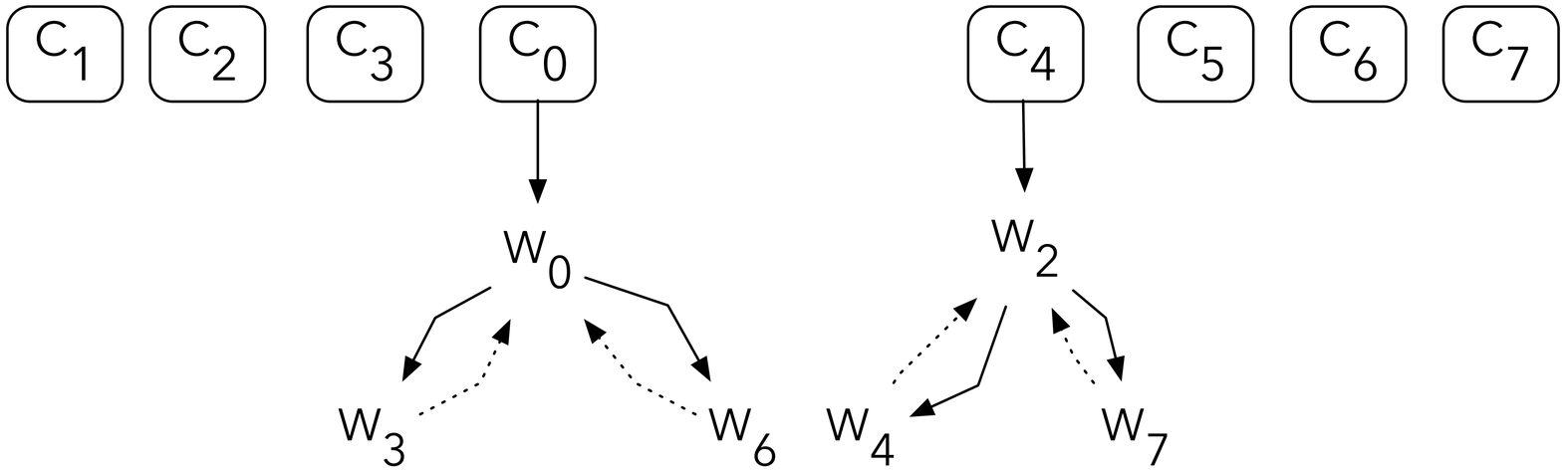


chunk index


7fa81b1
98df3e2
04f3a36
d94b766
c7d6888
449b97e
adb3413
6f716eb
dbf0c3b
58adce8
a344c7d
788a58f
3ca3cae
af8d35e
cf3aa67


fe0eca4
7fa81b1
4393228
5439595
f58e540
f9a3419
1a5a38a
0fa7205
7ac08b1

write operation



evaluation

object	I/O operation	size	execution time
O_1	W	10MB	1.142s
O_2	W	18MB	1.217s
O_3	W	37.83 MB	1.277s
O_1	R	10MB	1.188s
O_2	R	18MB	1.220s
O_3	R	37.83MB	1.296s
O_{17}	W	1GB	4.996s
O_{17}	R	1GB	5.871s

in short...

- accelerate r/w operations w/o os constraints
  - see [Quenum and Shipena, Int. J. of Cloud Computing, 2021]

cooperative vm placement

- map hardware resources to workloads
- slower execution with less resources
  - transfer VM for load balancing

hypervisor resource allocation

- VMWare ESX
  - resource commitment (memory over commit)
- Hyper V
  - dynamic memory

lease model

- spot instance (EC2)
- low-priority VM (Azure)
- pre-emptible and spot VMs (Google Computing Engine)

vm placement

goals

- maximize profitability (provider)
- guarantee performance following SLA (user)
- energy efficient

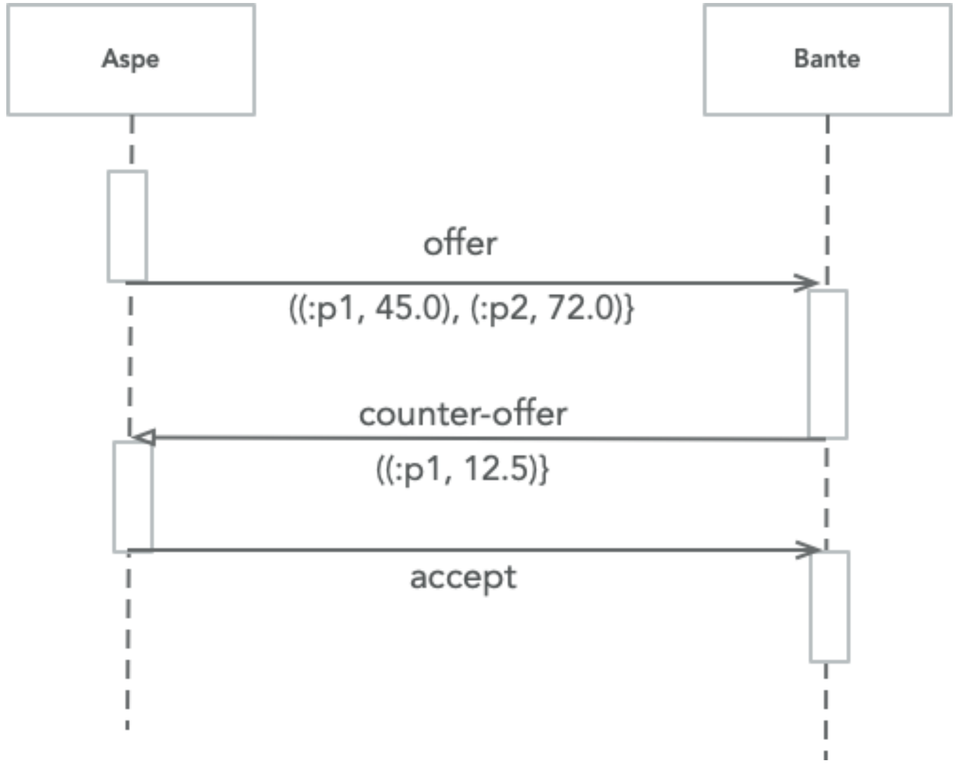
extend resource allocation to multiple hypervisors

automated negotiation

- 3 types of resources
  - cpu
  - memory
  - secondary storage

```
initiate_negotiation(Initiator, Agents)
  while(true) do
    handle_message(Iniator)

function initiate_negotiation(initator, agents)
    req := generate_request(initiator)
    pending := agents
    repeat
      send(req, select_provider(agetns, pending))
    until pending == {}
```

```
function handle_message(initator)
    msg := fetch message()
    agent := msg.destination
    if agent == initiator then
        if msg.type == accept then
            update agreements(msg)
        else
            if msg.type == counter_offer then
                evaluate_and_reply(msg)
    else
      if expired(msg) then
        send(reject, initiator)
      else
         if has_capacity_pref(msg, agent) then
            send(accept, initiator)
            update_agreements(msg)
        else
            c_offer := generate_counter_offer()
            send(c_offer,initiator)
```



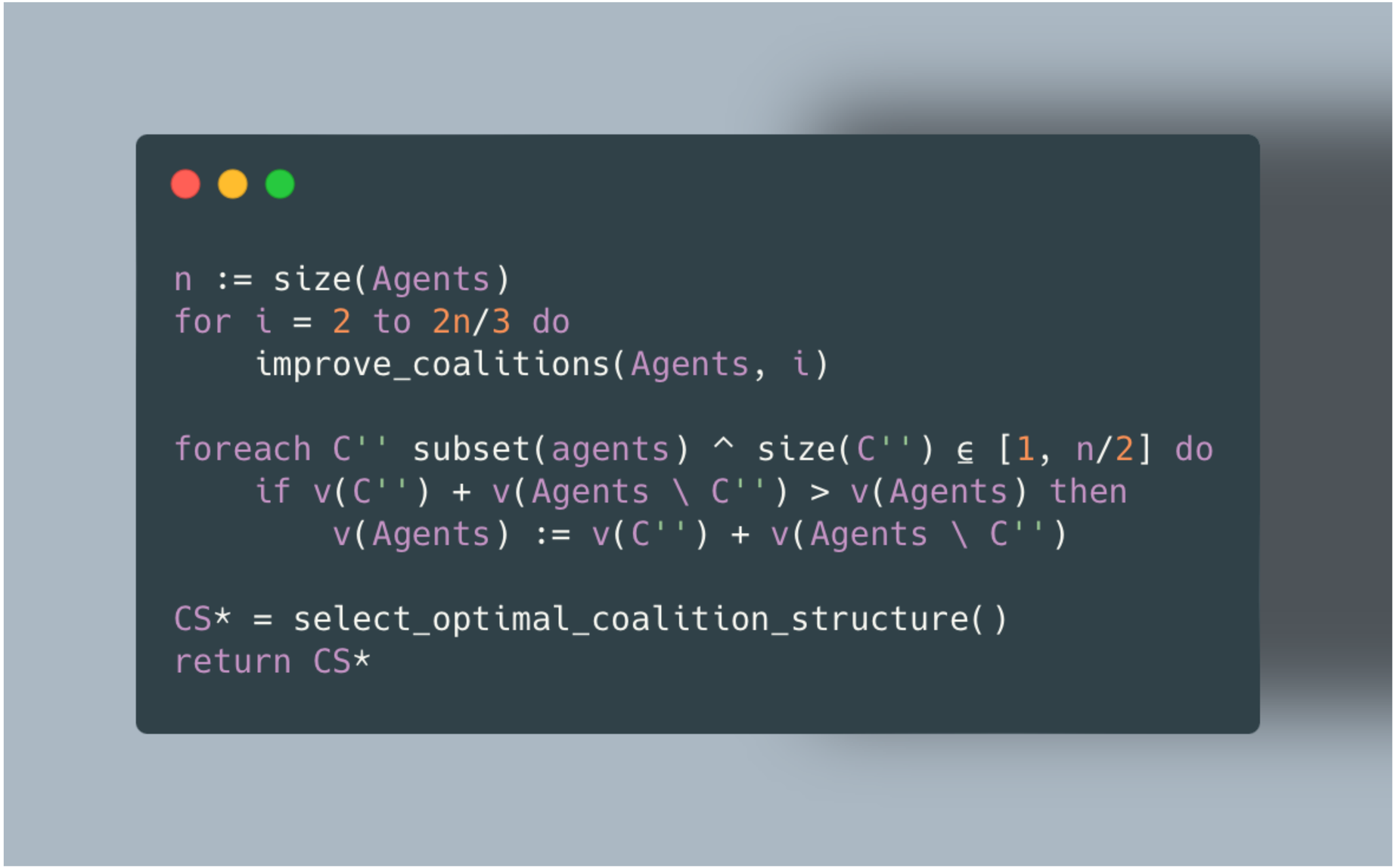

```
Agent Bante
--- As Provider ---
package p1
Agent: Kerou; quantity: 12.5

Agent Basque
--- As Provider ---                --- As Consumer ---
package p1                          package p1
Agent: Kerou; quantity: 12.0        Agent: Aspe; quantity: 20.0
Agent: Aspe; quantity: 12.5         Agent: Bearne; quantity: 20.3

Agent Aspe
--- As Provider ---                --- As Consumer ---
package p1                          package p1
Agent: Basque; quantity: 20.0       Agent: Basque; quantity: 12.5

Agent bearne
--- As Provider ---                --- As Consumer ---
package p1                          package p2
Agent: Basque; quantity: 20.3       Agent: Kerou; quantity: 10.0

Agent Kerou
--- As Provider ---                --- As Consumer ---
package p2                          package p1
Agent: Bearne; quantity: 10.0       Agent: Bante; quantity: 12.5
                                    Agent: Basque; quantity: 12.0
```

coalition formation

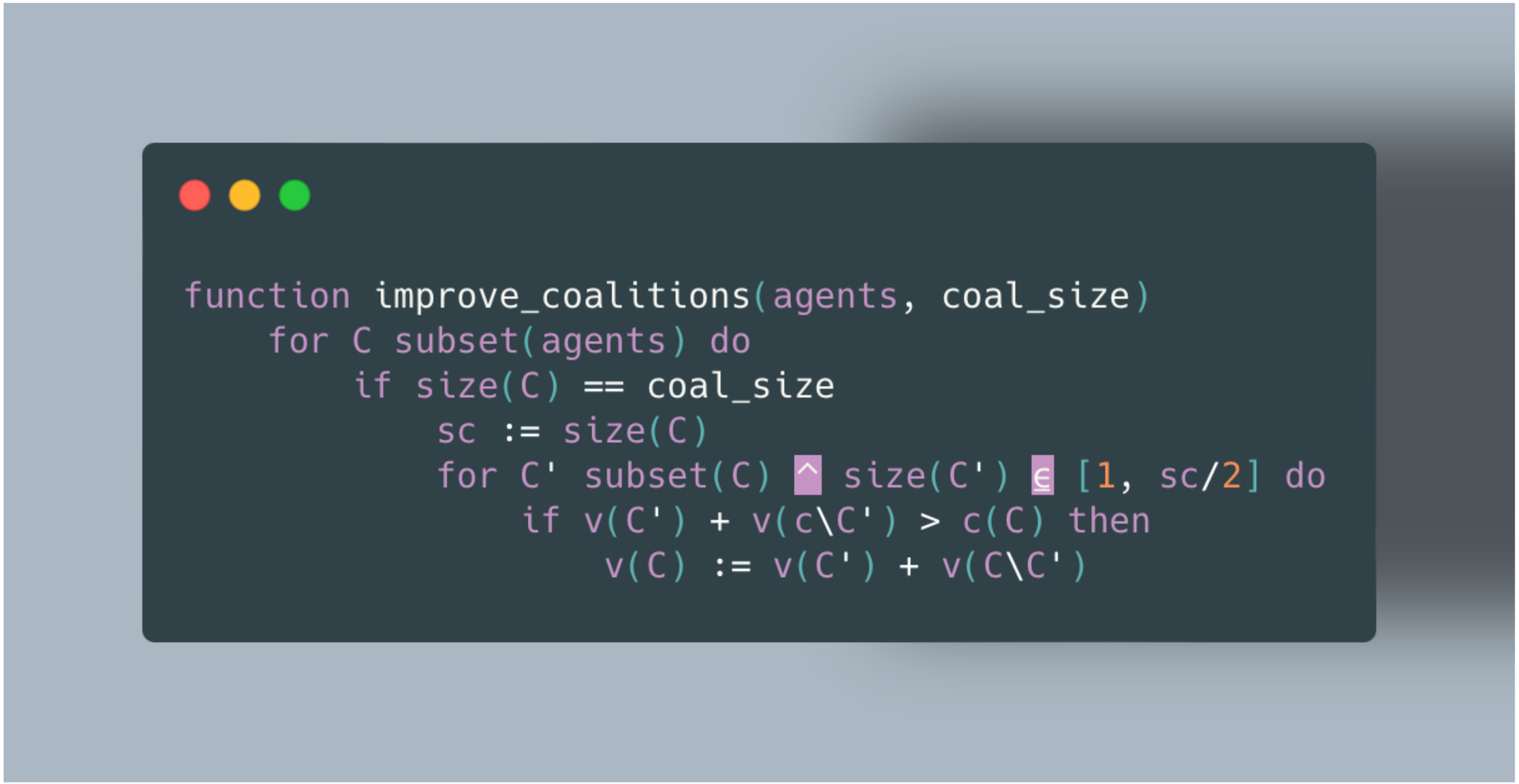
$$\mathscr{A} = \{\mathsf{A}_1, \mathsf{A}_2, \ldots, \mathsf{A}_n\} \qquad v : 2^{\mathscr{A}} \to \mathbb{R}$$

$$\{\mathsf{C}_1, \mathsf{C}_2, \ldots, \mathsf{C}_\ell\}$$

$$\mathsf{C}_\iota \neq \varnothing \,.\, \mathsf{C}_\iota \cap \mathsf{C}_J = \varnothing \,.\, \bigcup_{\iota=1}^{\ell} \mathsf{C}_\iota = \mathscr{A} \text{ with } \iota \text{ and } J \in \{1,2,\ldots,\ell\}$$



```
n := size(Agents)
for i = 2 to 2n/3 do
    improve_coalitions(Agents, i)

foreach C'' subset(agents) ^ size(C'') ⊆ [1, n/2] do
    if v(C'') + v(Agents \ C'') > v(Agents) then
        v(Agents) := v(C'') + v(Agents \ C'')

CS* = select_optimal_coalition_structure()
return CS*
```

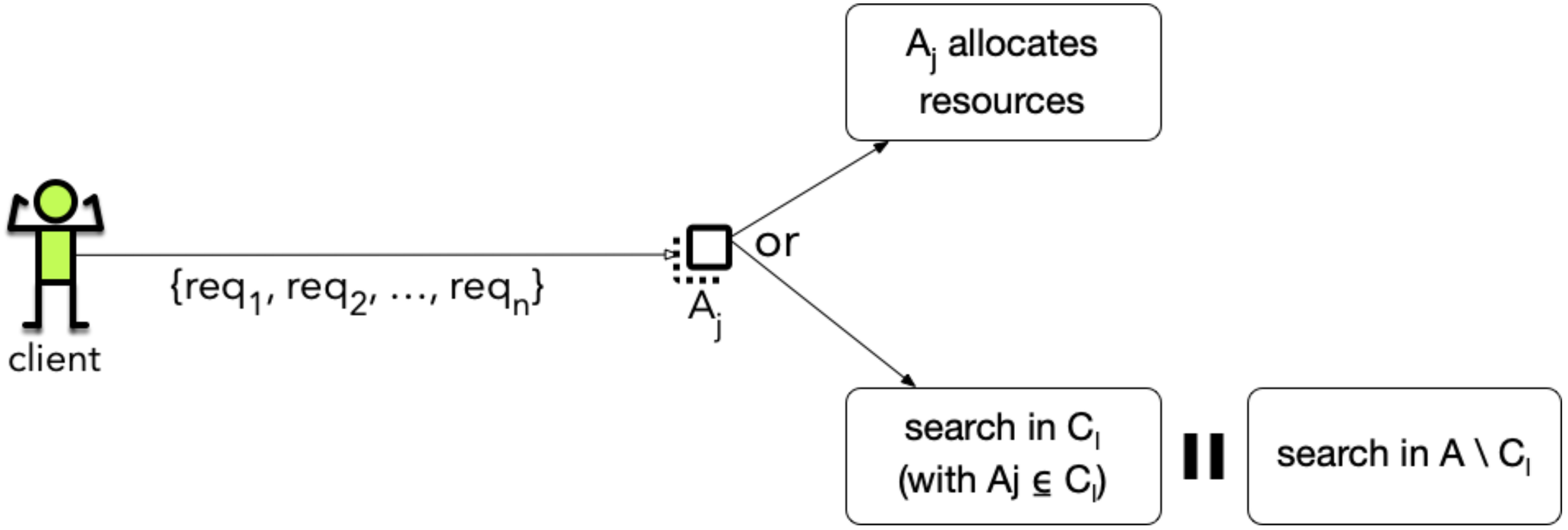


```
function improve_coalitions(agents, coal_size)
    for C subset(agents) do
        if size(C) == coal_size
            sc := size(C)
            for C' subset(C) ^ size(C') ϵ [1, sc/2] do
                if v(C') + v(c\C') > c(C) then
                    v(C) := v(C') + v(C\C')
```

$\{C_1, C_2\}$ with

$$C_1 = \{\texttt{Bante}, \texttt{Basque}, \texttt{Bearne}\} \text{ and } C_2 = \{\texttt{Kerou}, \texttt{Aspe}\}$$

resource allocation


client
{req1, req2, …, reqn}
Aj
or
Aj allocates resources
search in Cl (with Aj ∈ Cl)
search in A \ Cl

- beam search heuristic
  - sort requests from largest to smallest
  - select an agent (e.g., least tight agent)
  - define subsets of requests for each agent (capacity)
  - choose subset of highest cluster for the agent

$$\min \sum_{j=1}^{n} \sum_{\ell=1}^{m} \quad c_{\ell} \cdot y_{j\ell}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} x_{ij} = 1 \quad \text{for } i = 1,\ldots,n$$

$$\sum_{\ell=1}^{m} y_{j\ell} \leq 1 \quad \text{for } j = 1,\ldots,n$$

$$\sum_{i=1}^{n} \omega_i \,.\, x_{ij} \leq \sum_{\ell=1}^{m} W_{\ell} \,.\, y_{\ell j} \quad \text{for } j = 1,\ldots,n$$

$$x_{ij} \in \{0,1\} \quad \text{for } i, j = 1,\ldots,n$$

$$y_{j\ell} \in \{0,1\} \quad \text{for } j = 1,\ldots,n \text{ and } \ell = 1,\ldots,m$$

$$\mathsf{p}_{pl1}(t) = \vartheta + \eta \times \rho_p \times cpu\_load_{pl}(t)$$

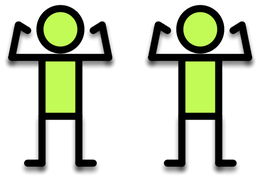
$$\mathsf{E}_{pl1} = \int_{start}^{end} \mathsf{p}_{pl1}(t)\, dt$$

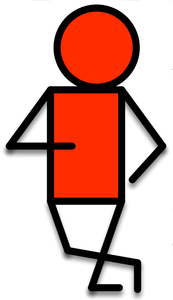
in short...

- reduce search scope using a coalition
  - see [Quenum and Aknine, ESOCC 2023]

in conclusion...

- automation
  - concrete solution to specific problems
- autonomy
  - decision-making in the solution





Dear Prot,
We recently embarked on a new project.
Its objectives are … We are turning to you for assistance in setting up the required infrastructure.
We look forward to your response.
Kind regards
Ben & Team

1

2

NLP techniques

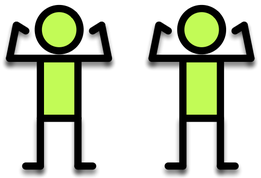
- text understanding
- need identification

3

- optimisation
- resource provisioning

4

- machine creation
- configuration
- setup



Dear Ben,
Please find below the credentials for the hosts you requested.
Machine 1
IP address: 196.216.167.002.
Accounts: ….
Please do not hesitate to contact me should have any further question. Best of luck in your project!
Kind regards
Prot

5

Jean-Pierre Briot

Onn Shehory

Shinichi Honiden

Samir Aknine

Aurelien Slodzian

Fuyuki Ishikawa

Agnes Lumala

Daniel Moldt

Bongani Shongwe

Shawulu Nggada

Anthony Iketchuwku

Jonas Josua

Guy-Alain Zodi-Lusilao

Gervasius Ishuuwa

Ahlonko Kuassi-Kpede

Emmanuel Ejembi

Valerie Garises

Odero Kenneth

Alexander B. Shipena

Nixon Ochara

Samuel Kapembe

references

- José Ghislain Quenum, Samir Aknine. A Unification-based Approach to Configure Generic Protocols into Agent Interaction Models. International Journal of Agent-Oriented Software Engineering, 2010, 4 (1), pp.32-78.
- José Ghislain Quenum. Generic Protocol Configuration and Execution within a Coalition. IAT 2008, pp 214-223
- José Ghislain Quenum. A Unification and Delegation Approach to Configure Generic Protocols for Agent Interactions. IAT 2007, pp 281-284
- José Ghislain Quenum, Samir Aknine. Enabling Agents to Dynamically Select Protocols for Interactions. CoRR, 2005.
- José Ghislain Quenum, Samir Aknine. A Dynamic Joint Protocols Selection Method to Perform Collaborative Tasks. CEEMAS 2005, pp 11-20.
- José Ghislain Quenum, Alexander Brown Shipena. Peer-to-peer storage engine for schemaless immutable data. Int. J. Cloud Comput. 10(5/6): 655-668 (2021)

references (cont’)

- José Ghislain Quenum, Samir Aknine, Onn Shehory, Shinichi Honiden. Dynamic Protocol Selection in Open and Heterogeneous Systems. IAT 2006, pp.333-341.
- José Ghislain Quenum, Fuyuki Ishikawa, Shinichi Honiden. Protocol Selection alongside Service Selection and Composition. ICWS 2007, pp. 719-726.
- José Ghislain Quenum, Samir Aknine. Towards Executable Specifications for Microservices. SCC 2018, pp. 41-48
- José G. Quenum, Jonas Josua. Multi-cloud serverless function composition. UCC 2021, pp. 9:1-9:10
- José Ghislain Quenum, Gervasius Ishuuwa. Abstracting Containerisation and Orchestration for Cloud-Native Applications. CLOUD 2020, pp. 164-180
- José G. Quenum, Samir Aknine. Cooperative Virtual Machine Placement. ESOCC 2023, pp. 136-150
- Agnes F. N. Lumala, José Ghislain Quenum. A Distributed Problem Solving Approach for Service-Oriented Computing Systems. SERVICES I 2009: 530-538

thank you!