An Operating System for the Real World

Why its pilot development needs OTP/Erlang

Paul Valckenaers

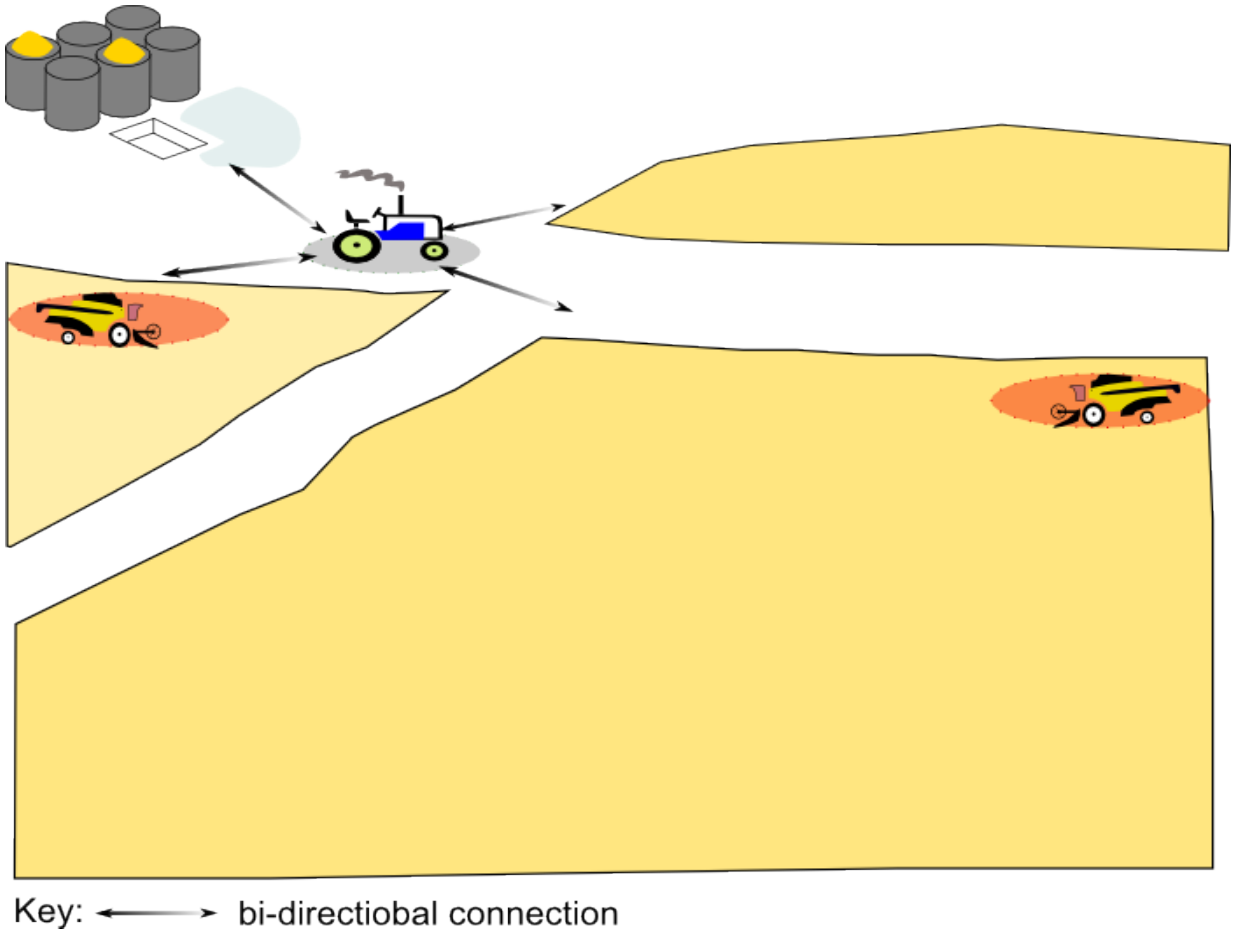
KU Leuven association
valckenaersp@acm.org

Personal History

- Late 1980's BYTE Magazine
  - Erlang : keeps the good & discards the bad
  - Loved it, but *"you can't have it all"*
- 3Q2010
  - CACM paper on Erlang
  - Demonstration of coordination of *combine harvester and tractor fleet operations*
  - Incentive to have a closer look

Demonstration 4Q2010

Activity Coordination



Research – Smart Systems

- Manufacturing Execution Systems (MES)
- Logistic Execution Systems (LES)
- *Robotic* fleets (hospitals, harvesting, road construction, open air mining …)
- Intelligent Traffic Systems (ITS) – FP7/MODUM
- Smart Grids (e.g. refrigerate when wind mills…)
- Electro-mobility (combines previous three)

Activities involving Valuable Resources

Computer OS

- Manages resources
  - CPU cores
  - Memory
  - …
- Used by activities
  - Processes
  - Threads
  - …
- Making minimal assumptions about …

Operating Systems

- **Explicit and mandatory/enforceable resource allocation**

The operating system is in charge of allocating resources such as time slices on processor cores, access and locks on data files, rights to use peripheral devices …

All activities need to acquire and release the resources that they need.

Operating Systems

- **Interaction facilities**

The operating system provides valuable services that the activities (processes) use to efficiently and effectively interact.

In particular, these services enable to communicate with the proper counterparts.

Importantly, these services allow activities to use the available resources efficiently (e.g. signal & wait avoids the need for polling).

Operating Systems

- **Application agnostic**

The operating system will have a scope – e.g. embedded or desktop applications – and will make some restrictive assumptions – e.g. concerning real time applications.

But, these constraints have a technological origin and are not reflecting specific expectations concerning the applications:

The modern operating system is designed for the unexpected

Differences

- Resources are orders of magnitude **more expensive and valuable**.
- The socio-economic impact of the activities is orders of magnitude more significant.

  Note: difference must not become so big that it becomes economical to have highly paid humans experts manage the activities and resources.

- Resources and activities are **more diverse and more complex**.

  They have state and state trajectories. Allocation and de-allocation cannot assume some (limited number of) reference states. A truck that is de-allocated in Barcelona cannot be allocated in Antwerp without executing an activity that transfers the truck to Antwerp first, which requires allocation of resources (truck, driver, fuel).

- **Imperfect enforceability** of the resource allocations.

  A truck may break down or get involved in a car crash.
  A truck may discover that the load is overweight during pick-up.

OS-4-RW Implications

- **Larger computational budget**

  OS4RW can perform complex computations and communications/ interactions to take optimized decisions and actions because the economic gains will cover the effort and investments.

- **More challenging tasks**

  Necessary for the OS4RW to be smarter than a computer OS because it is managing more complex and ***less predictable/ controllable*** *resources and activities.*

  Needs to be a model-driven design where models mirror real-world counterparts (e.g. trucks, overnight charging of car batteries …).

OS-4-RW Achievements

- *Software engineering and formal methods*
  by ..., **Michael Jackson**, ... in ACM vol. 51, no. 9

  "Software-intensive systems are intended to interact dependably with the human and physical problem world … non-formal problem world has many parts—human, natural, and engineered—whose properties and behavior are far less reliable … How can a dependable system be designed?"

  " … long-term community of engineers specializing in systems of the particular product class. For a critical software-intensive system, specialization is not optional."
- Move the starting point where this *specialization* is required closer to the final product… **relax this req**.

OS-4-RW Achievements

- Move the starting point where specialization is required closer to the final product
  - The knowhow of experienced specialists' is crystalized in the OS-4-RW
- Allow less experienced developers master the simpler tasks, projects… independently
  - The OS-4-RW handles the tricky parts
- Enable experienced developers to enter later in the project and wrap things up, undo, redo…
  - OS-4-RW resource allocation is key enabler

OS-4-RW Achievements

- Multi-party solutions
  - OS-4-RW resource allocation is key
- Allow co-existence of …
  - OS-4-RW resource allocation is key
- Enable evolutionary development…
  - OS-4-RW resource allocation is key enabler

Design for the unexpected

What about Erlang?

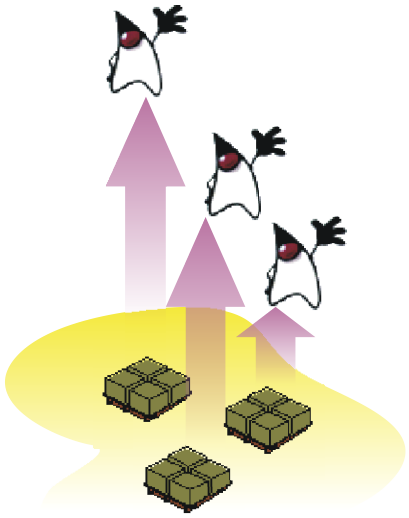
- Key enabling technology to make this feasible until OS-4-RW is recognized and broadly accepted (as ERP, Computer OS)
  - Addresses key concerns for pilot development and deployment
- Past developments: Java, dot-net, …
  - Academic demo = too far from industrial pilot
  - Industrial demo >> requires R&D funding
  - Time window is too small for …

What about Erlang?

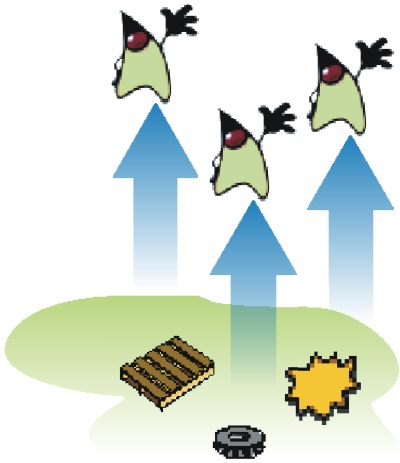
- Current developments
  - Tech transfer starts from existing software and involves end-user preferences (e.g. dot-net).
  - Recent research activities are using OTP/Erlang
  - MODUM – FP7 Project on ITS (4Q2011-3Q2014)
  - IFAC Technical Committee 5.1
    - Manufacturing Plant Control (on MES)
    - http://tc.ifac-control.org/5/1
    - http://www.linkedin.com/groups/IFAC-TC51-Manufacturing-Plant-Control-4235564

OS-4-RW Architecture

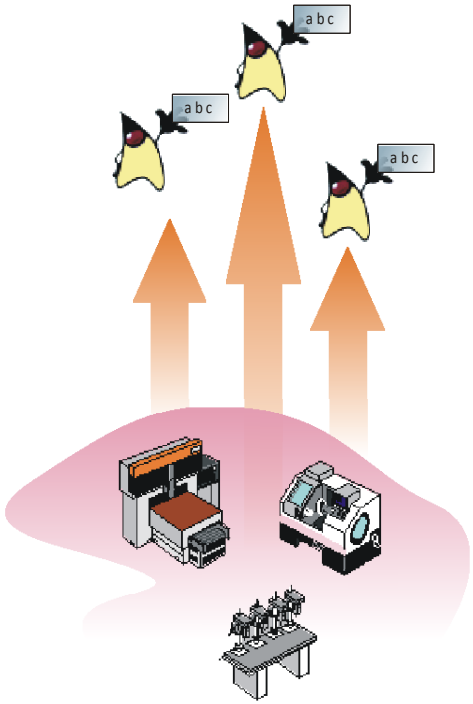
Mirror reality



Orders



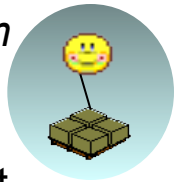
Product types



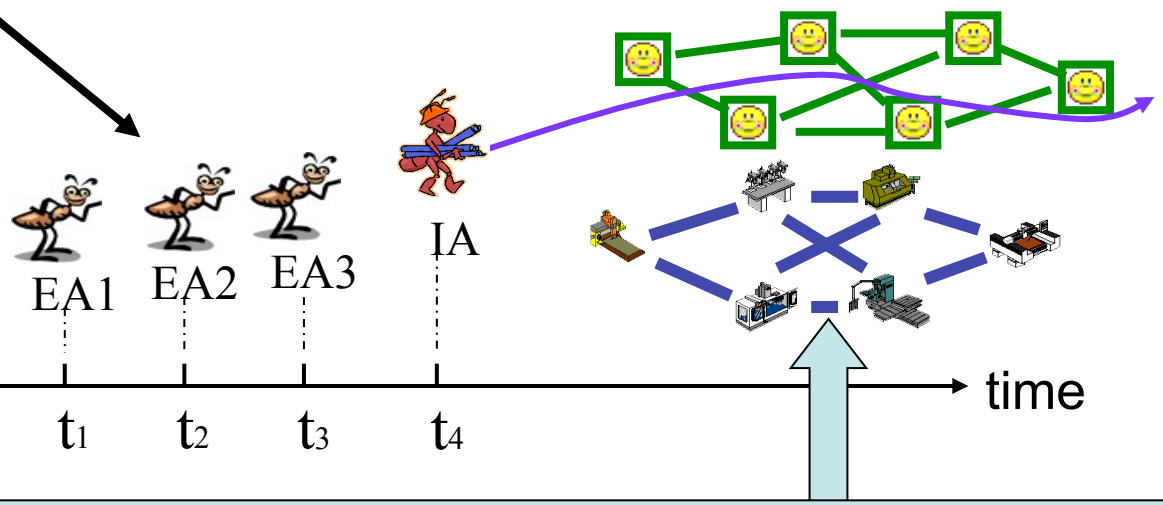
Resources

OS-4-RW “mirror” supports virtual navigation and execution – *explore*

Order agents create *swarm agents*, at regular time intervals, that virtually execute the remainder of their routing and report back how it performs.

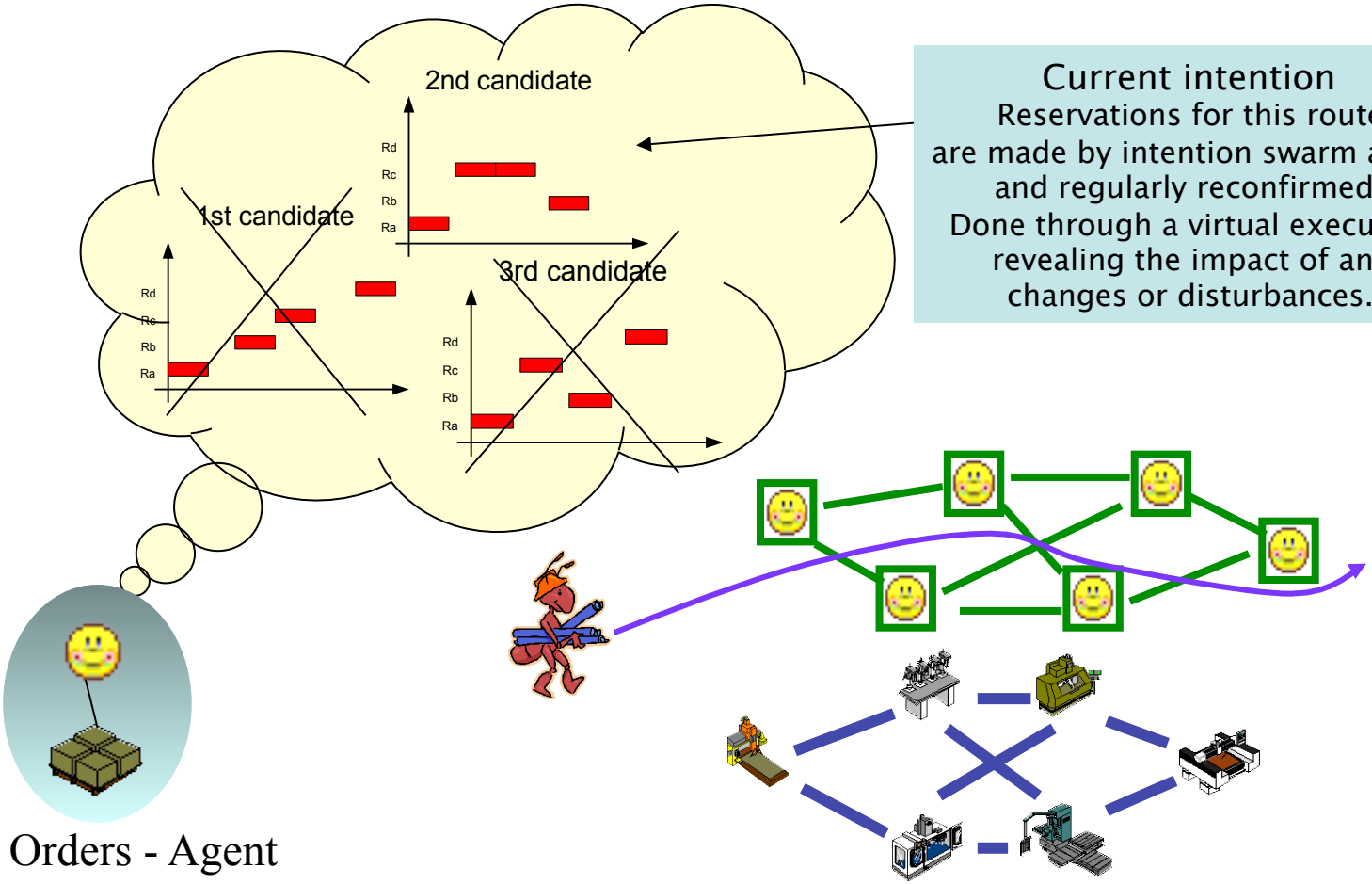


E.g. *swarm agents* virtually execute a possible routing

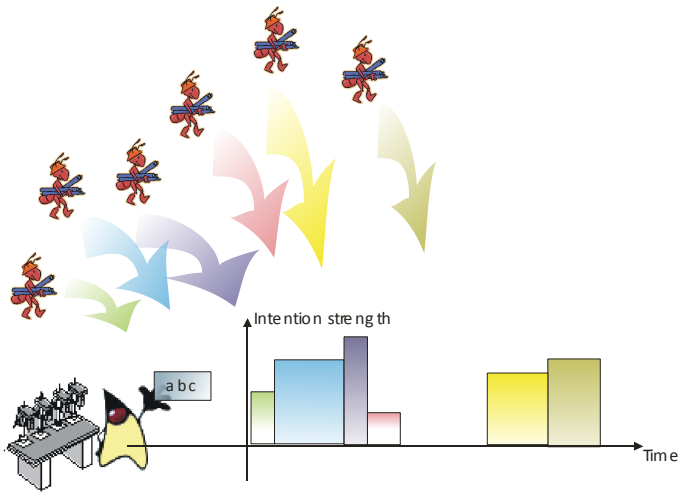


Software objects (processes) reflect reality and its structure

Intention swarm agent virtually execute the order intention in mirror



Intention propagation generates the forecast acounting for interaction



Intention swarms from orders that will visit this resource in future make (and regularly reconfirm) the corresponding reservations on the resource agent's agenda.

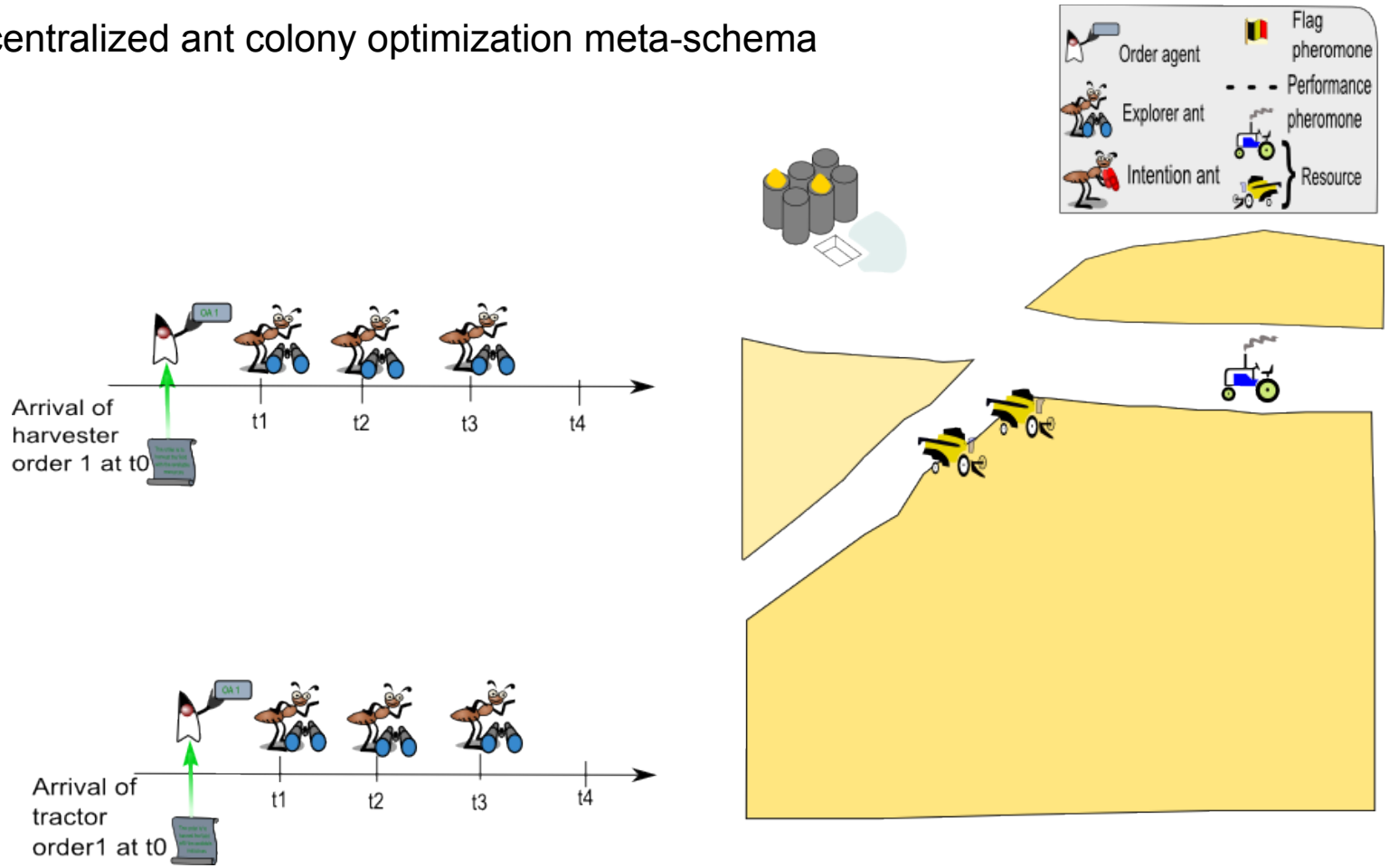
Resource agents are informed of expected visits by production orders and the processing steps that then will need to be performed.

This information is used in the self-model of the resource agent to support an accurate virtual execution on behalf of the visiting swarm agents.

A refresh-or-forget mechanism ensures regular updates and removes out-of-date information.

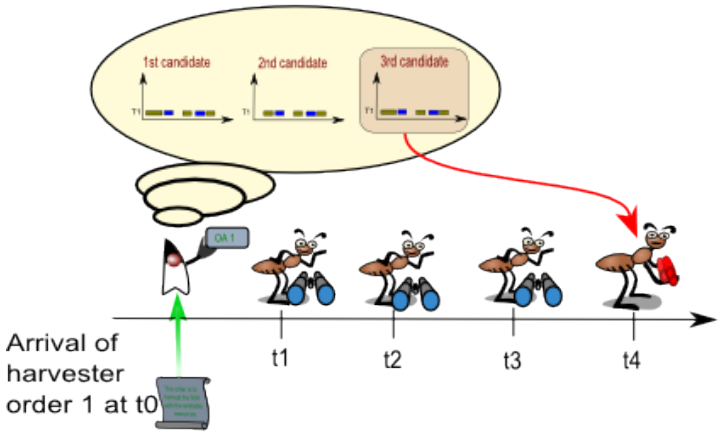
OS-4-RW – Harvesting

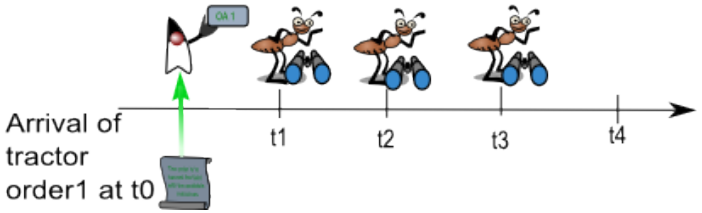
Decentralized ant colony optimization meta-schema

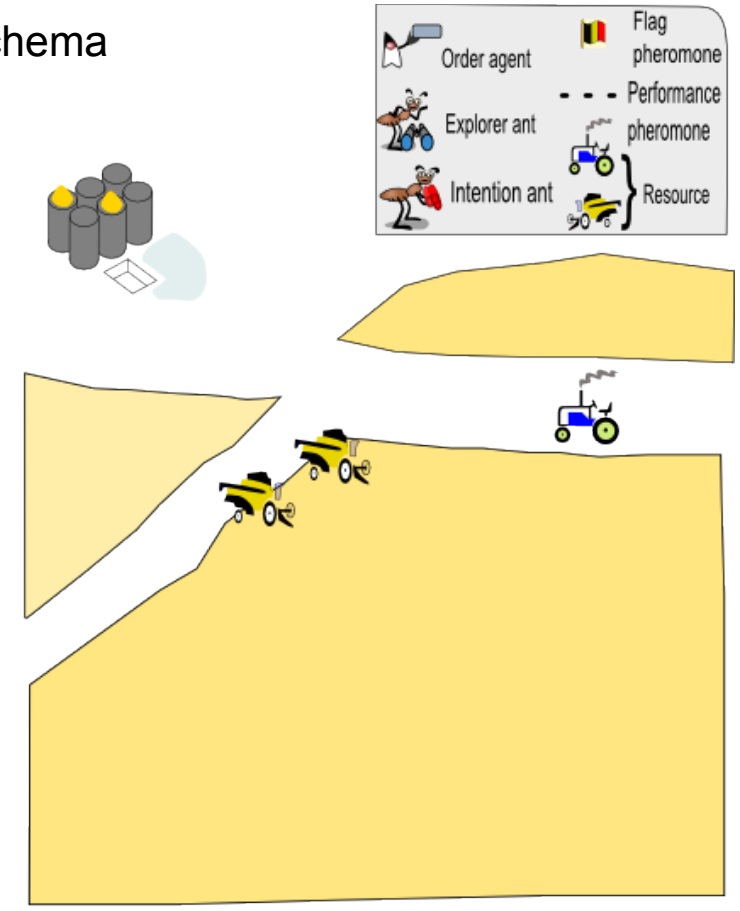


OS-4-RW – Harvesting

Decentralized ant colony optimization meta-schema

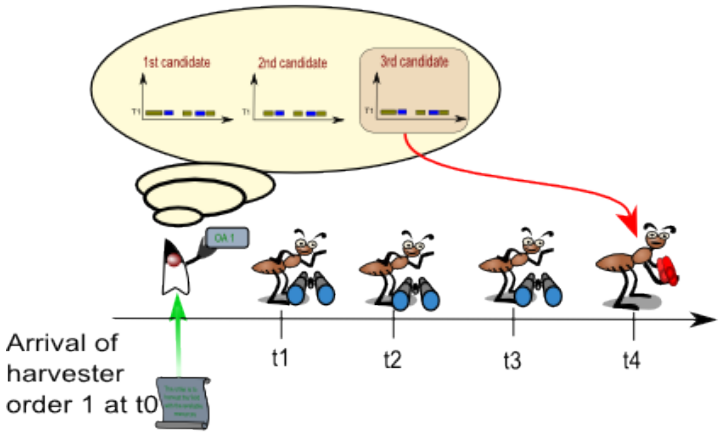


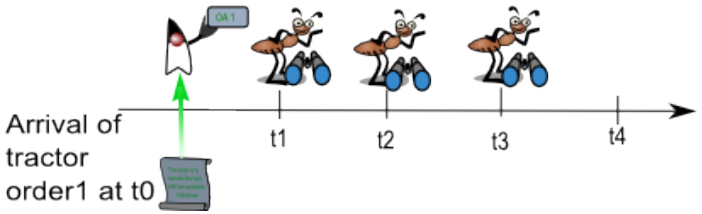


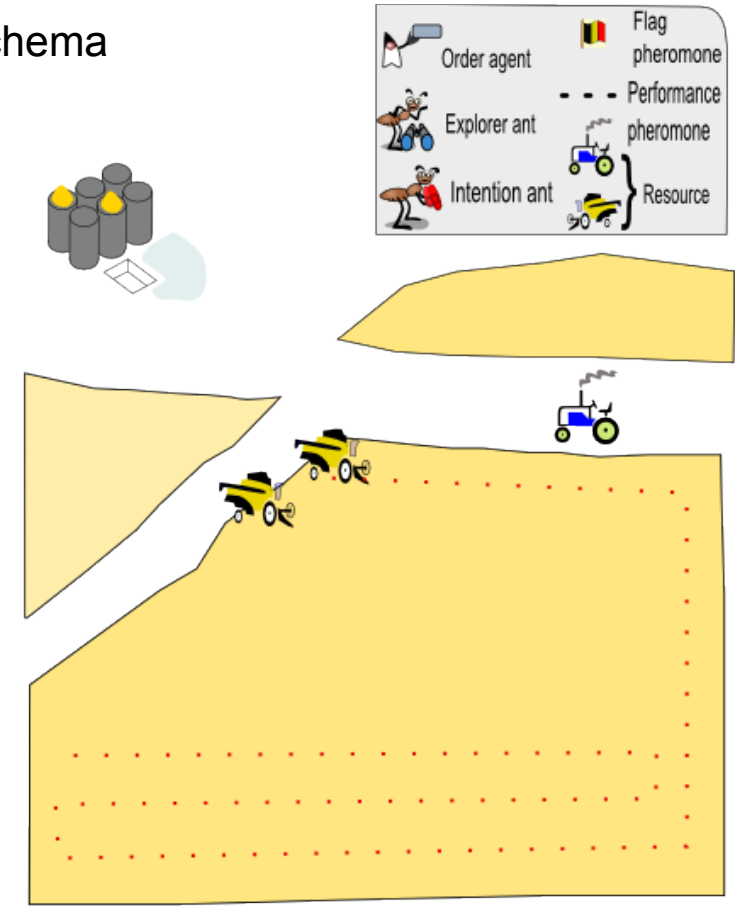


OS-4-RW – Harvesting

Decentralized ant colony optimization meta-schema

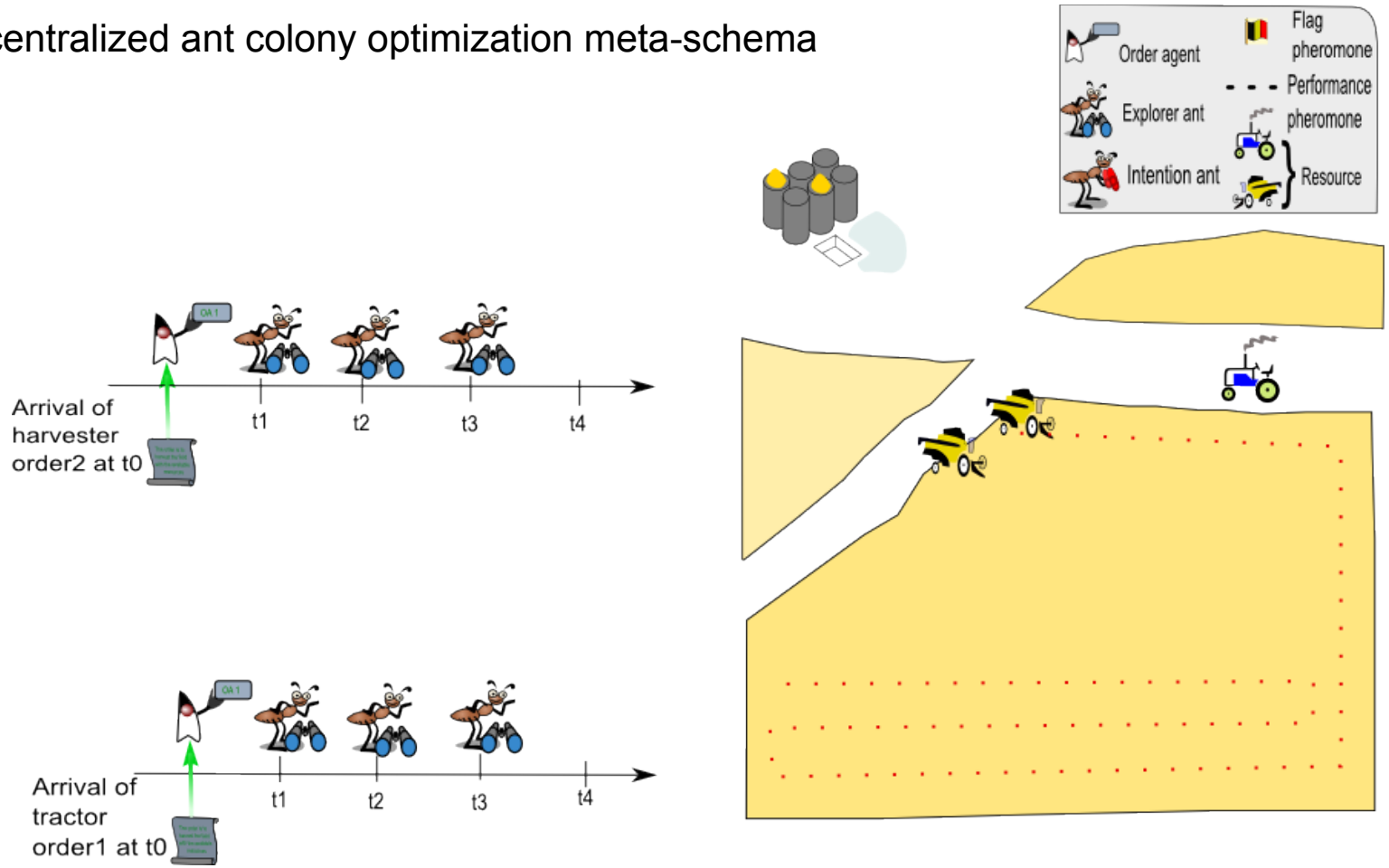






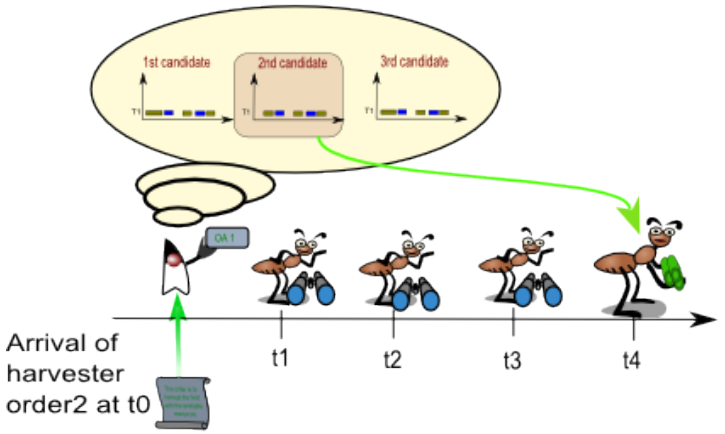
OS-4-RW – Harvesting

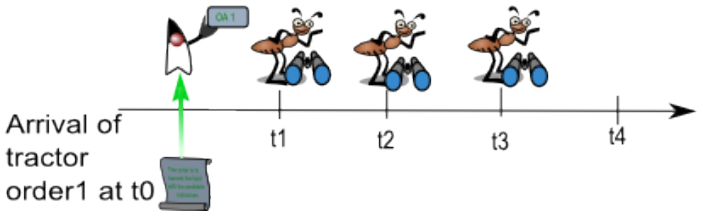
Decentralized ant colony optimization meta-schema

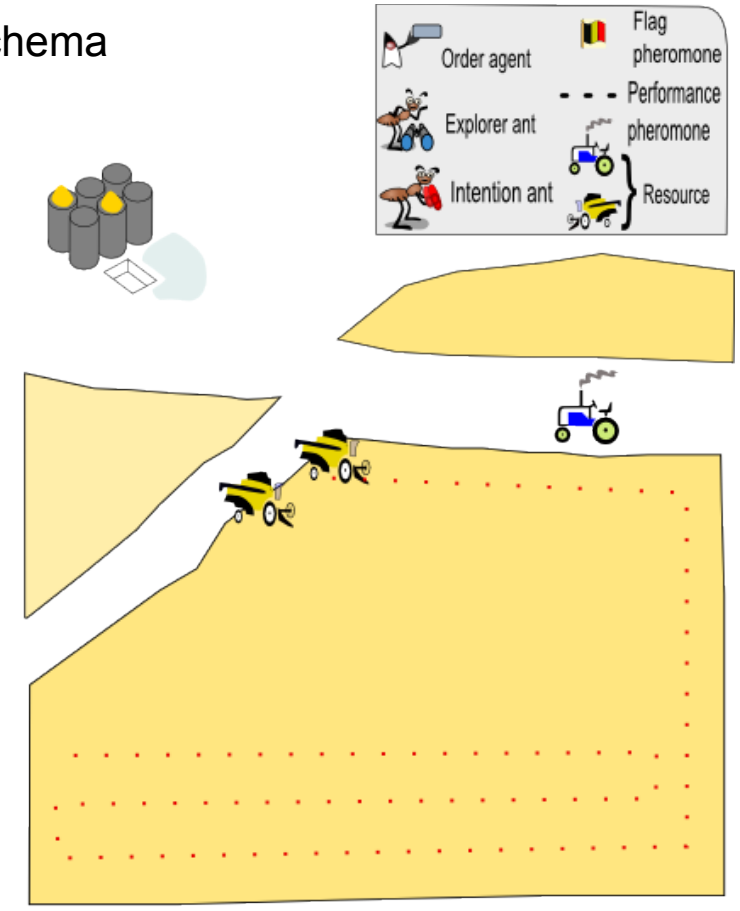


OS-4-RW – Harvesting

Decentralized ant colony optimization meta-schema

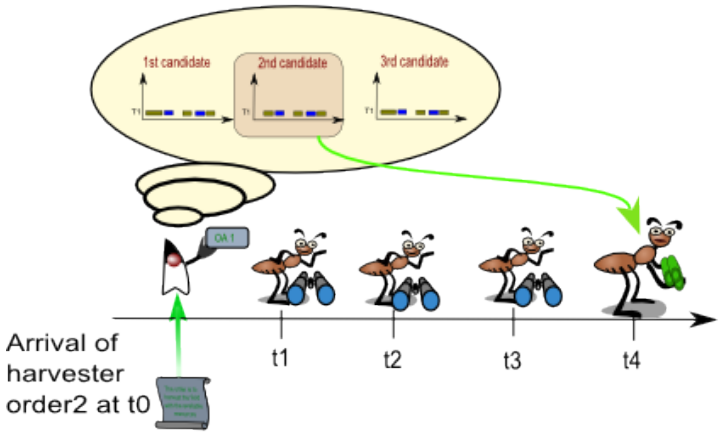


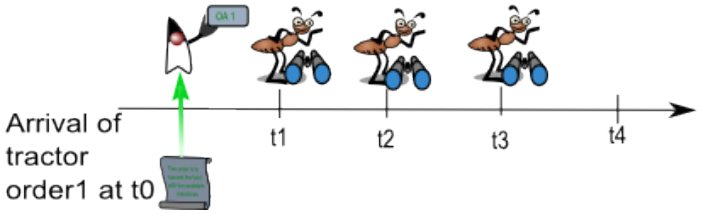


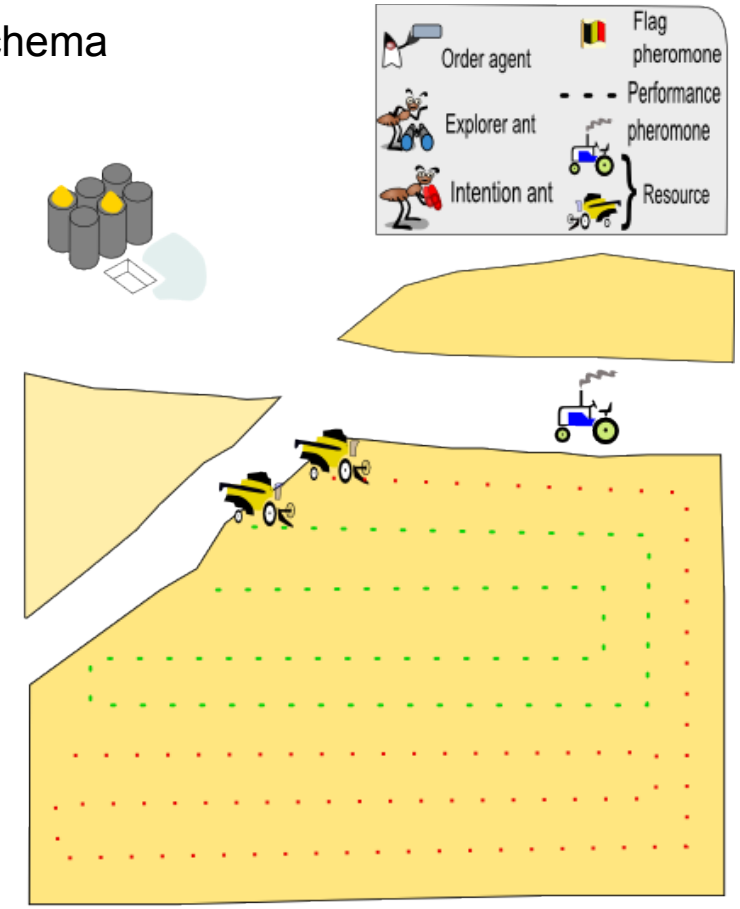


OS-4-RW – Harvesting

Decentralized ant colony optimization meta-schema

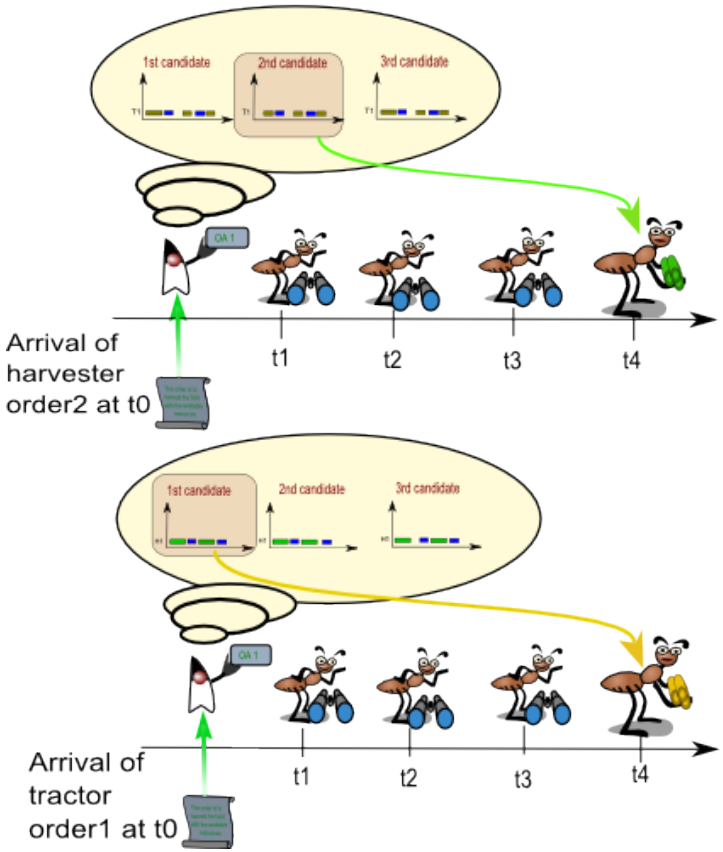


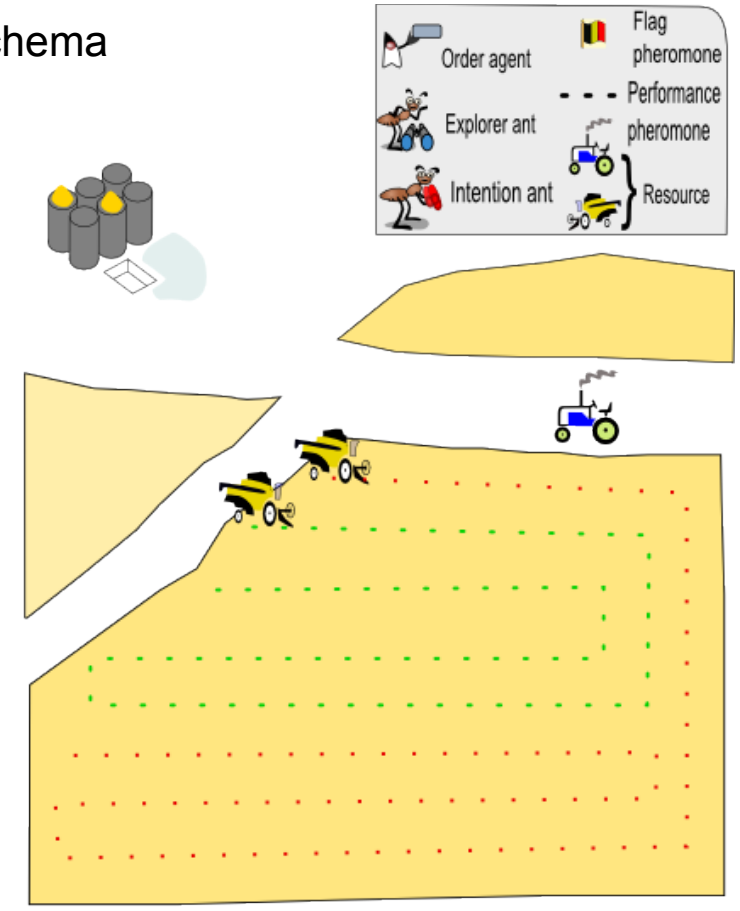




OS-4-RW – Harvesting

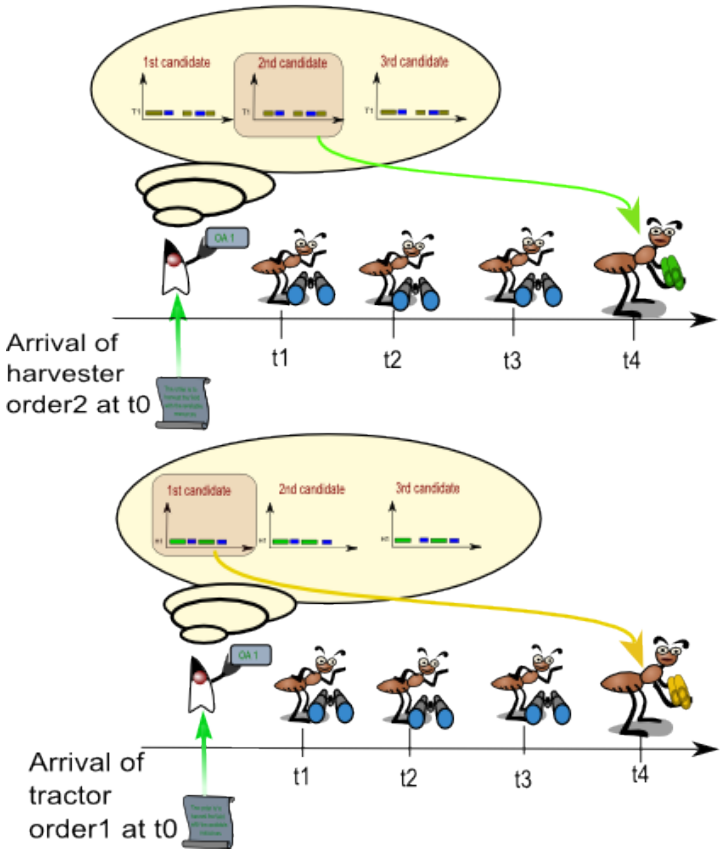
Decentralized ant colony optimization meta-schema

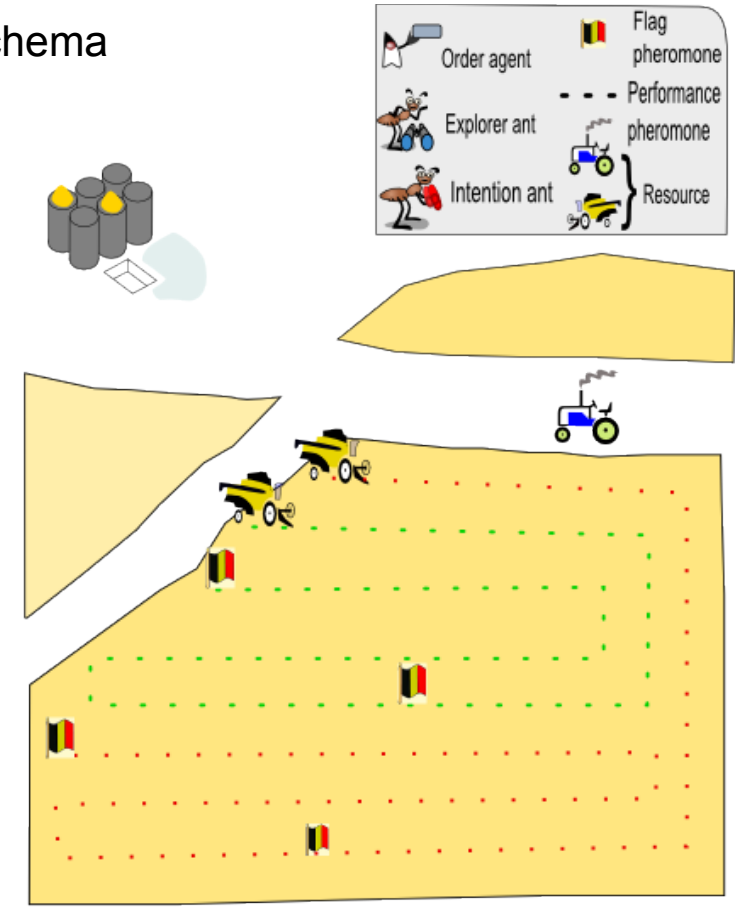




OS-4-RW – Harvesting

Decentralized ant colony optimization meta-schema





Emerging Online Planning

- Online planning method
  - Detailed schedules for the coordinating

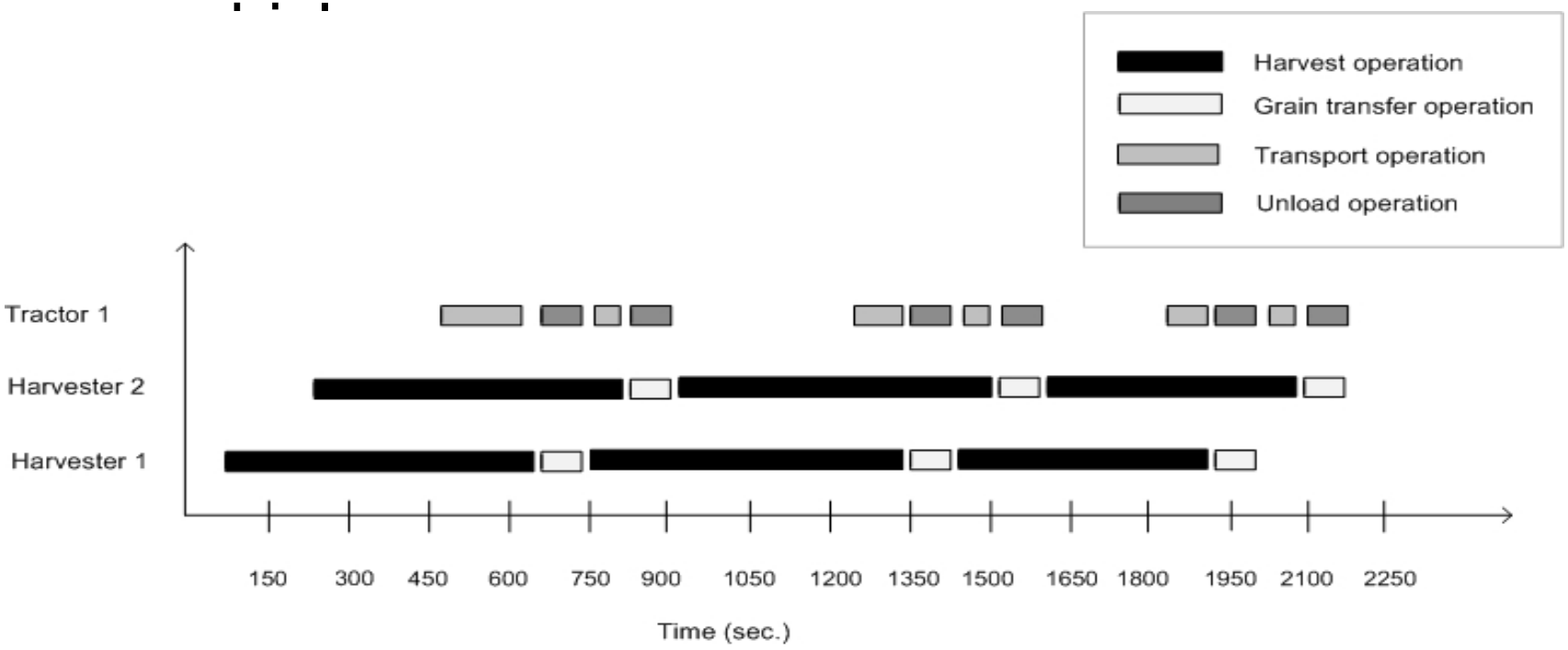
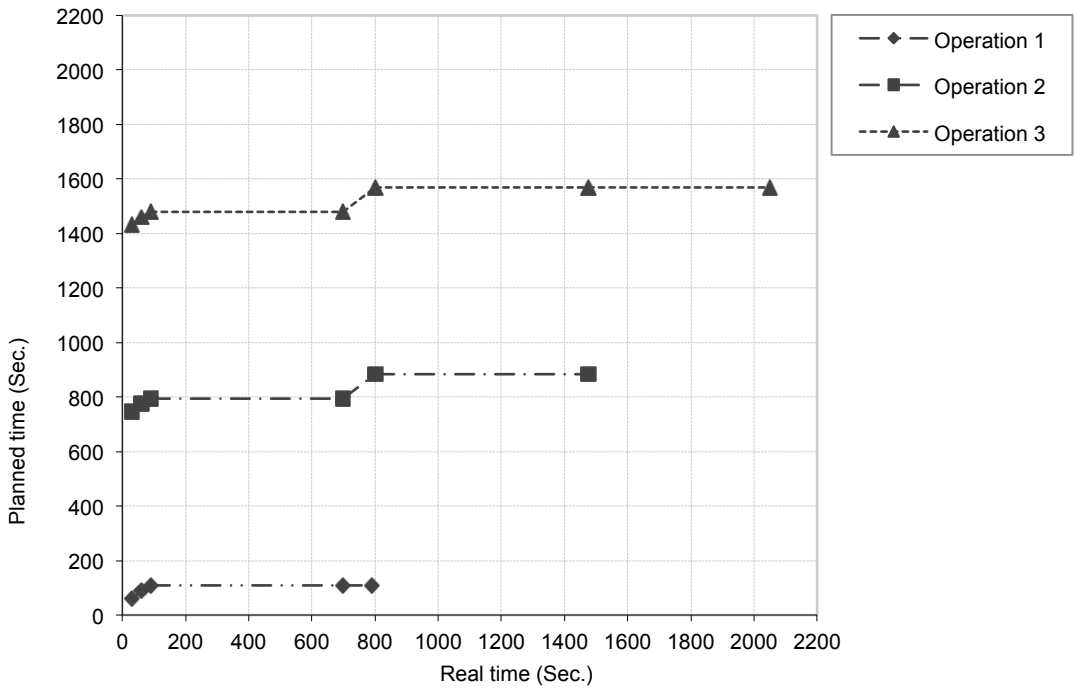


Fig. Local schedules of the harvesting vehicles

Emerging Online Planning

- Plan maintenance according to runtime conditions
  - Upc
  - Cor
    rem



the

Fig. Local schedules of the combine harvester

Real world deployment

- Connecting the system to the real world resources
  - Distributed setup of the system
  - Middleware for information exchange
  - Communication via GPRS/WiFi

Fig. Distributed setup of the online planning system

Real world deployment

- Planning and control of the real world harvesting vehicles
  - C…s:
    V…
  - R…s

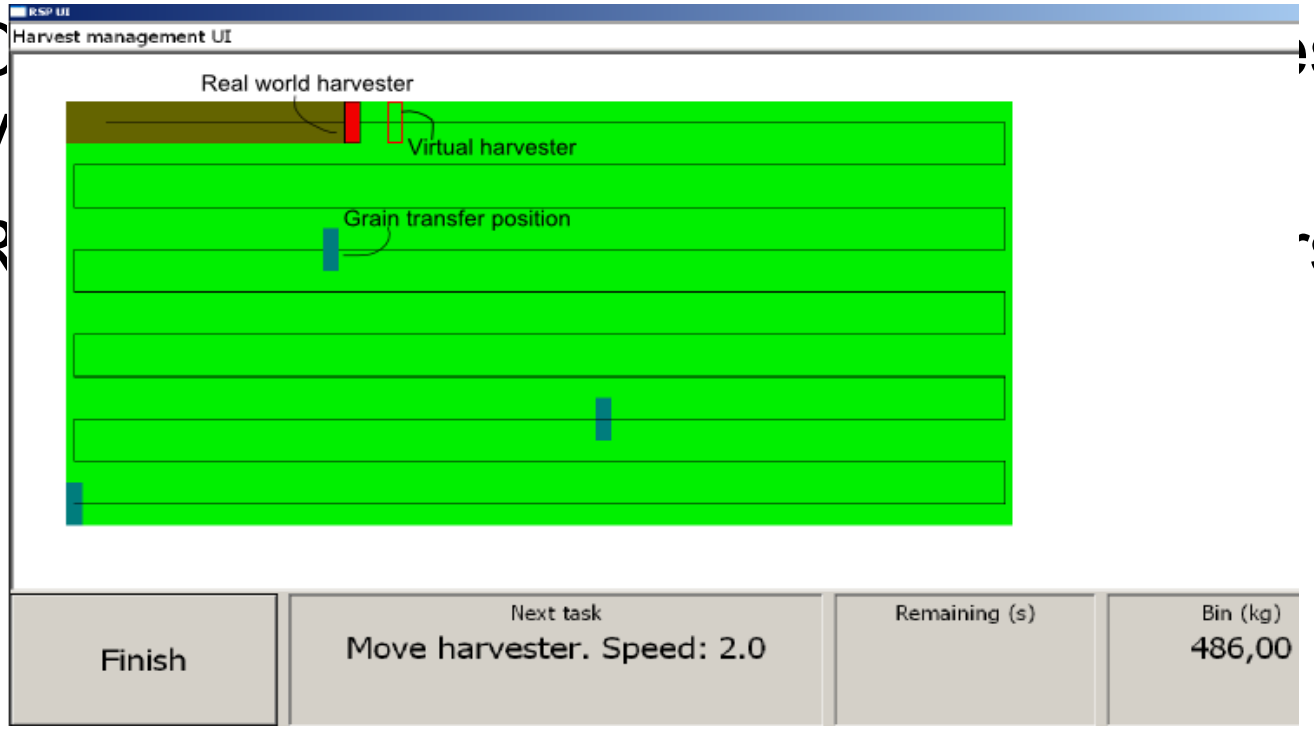
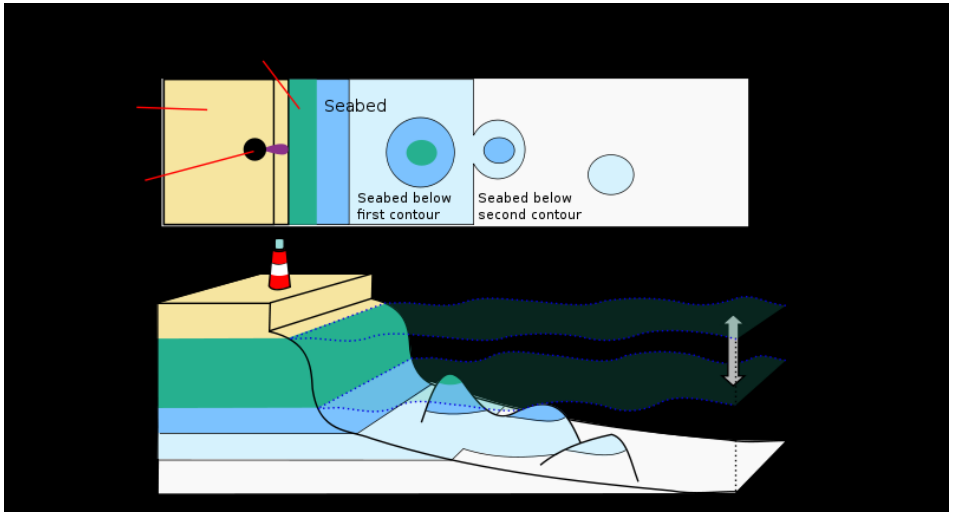
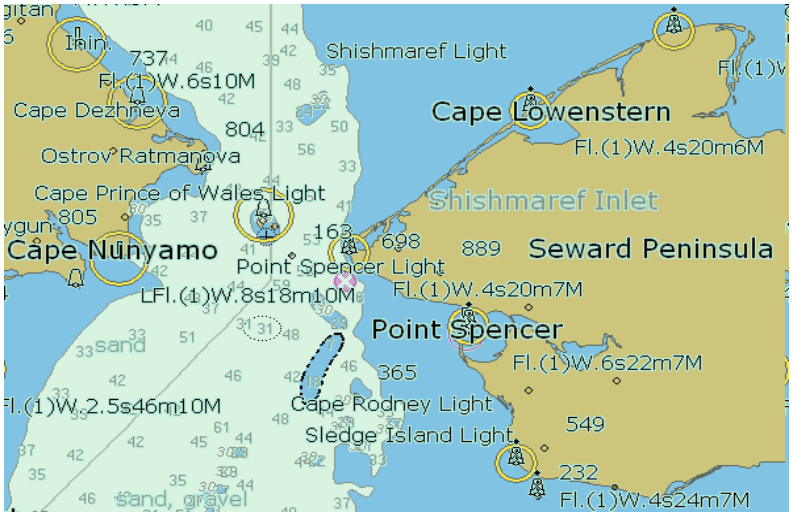


Fig. UI of the combine harvester

Example: resource = shipping lane

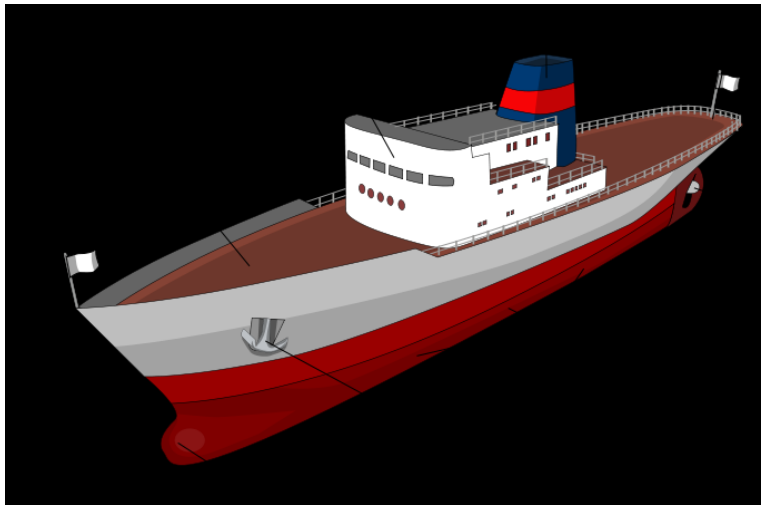
- Complicated: Intelligent lighthouse **(L)**
  - *Maps of the sea bottom, currents and tides*
  - *Status information and forecasts*
  - *D-GPS reference signal*
  - *RESOURCE ALLOCATION = EXPLICIT*





Example: resource = cargo ship

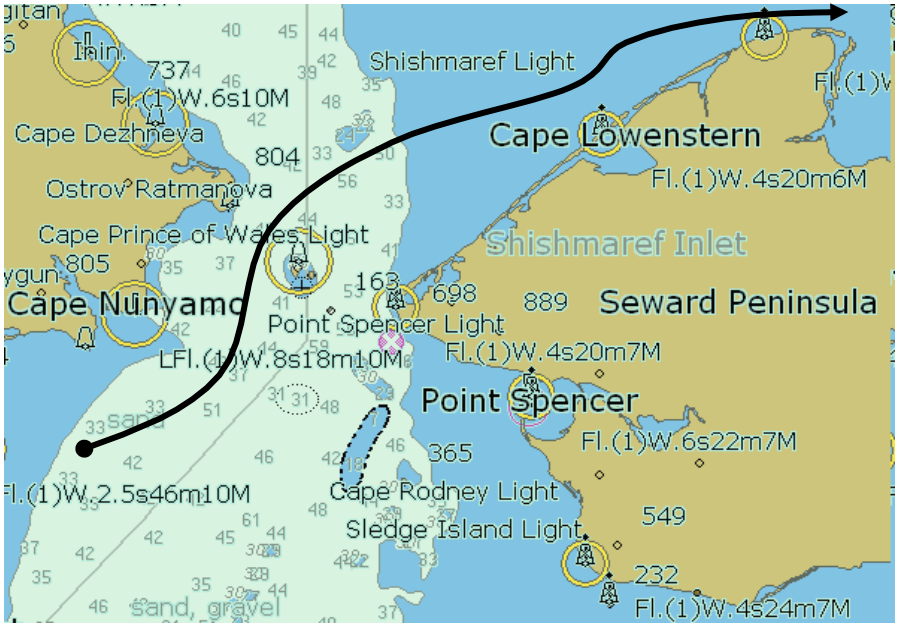
- Complicated **(B)**:
  - *Ship's technical data: power, size and shape, …*
  - *Status information: cargo, fuel, …*



Example: virtual execution

- Proactive coordination and control
  - *Have (A) steer the ship in predicted situations; (A) operates in simulation*
  - *Map ship trajectory relying on (B), (L) and (A)*

*Result is to **simulate and predict***



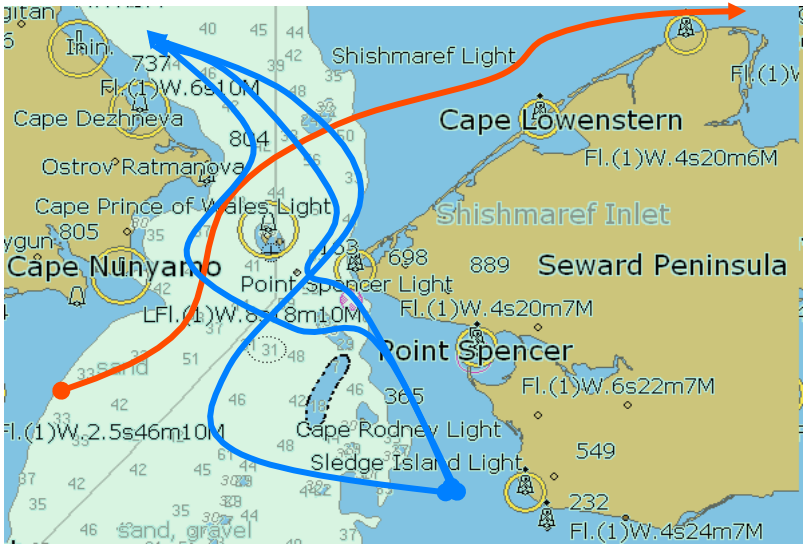
Example: forecast with interactions

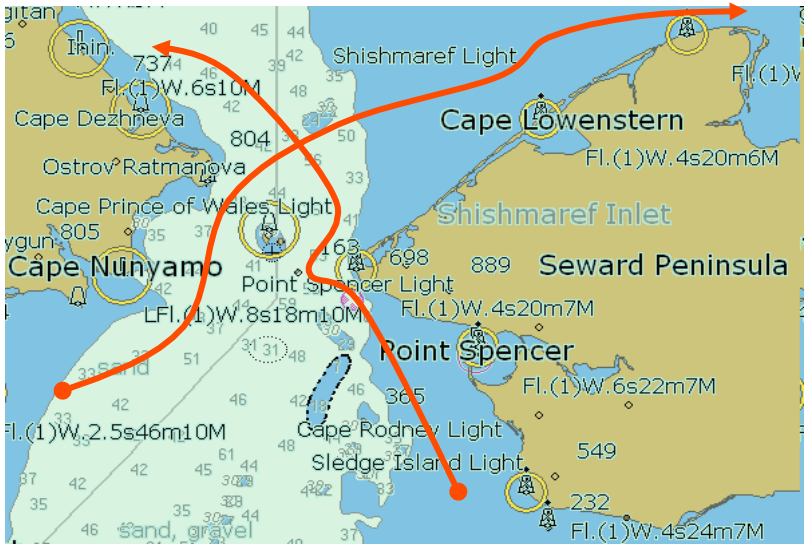
- Proactive coordination and control
  - *Map ship trajectory relying on (A), (B) and (L)*
  - *Make every mapped trajectory visible on (L)*

*Result is to simulate and **predict – individualized – Interactions***

Example: explore with interactions

- PROSA and Delegate MAS
  - Digital world & virtual execution
  - Exploring ant agents, intention ant agents for every *order*
  - Emergent self-organising forecasting and coordination





CONCLUSION

- OS-4-RW
  - Targeting ITS, MES, LES, … (not info-centric)
  - Valuable (e.g. get a personalized green wave)
  - Computationally efficient executable models
- OTP/Erlang is key enabler
- Recent activities are using OTP/Erlang