Agent-based Modelling and Simulation in Systems Biology

Emanuela Merelli

joint work with
Nicola Cannata and Luca Tesei
University of Camerino

Mini W/S on "Computational Approach to Biology"
Pisa - 25 October 2007

Outline

- Modelling and simulation biological systems: the challenges
- Agent and multi-agent paradigm: the historical overview
- From local interactions to global behaviour: coordination models
- Multilevel agent-based modelling in systems biology
  - our experience: Orion

The challenge of the 21st century will be to understand **how** biological molecules in living systems **integrate** to complex systems, how these systems **function** and their **evolution**.

We are scaling up from computational molecular biology (i.e. genomes, transcriptomes, proteomes, metabolomes) to **computational systems biology**

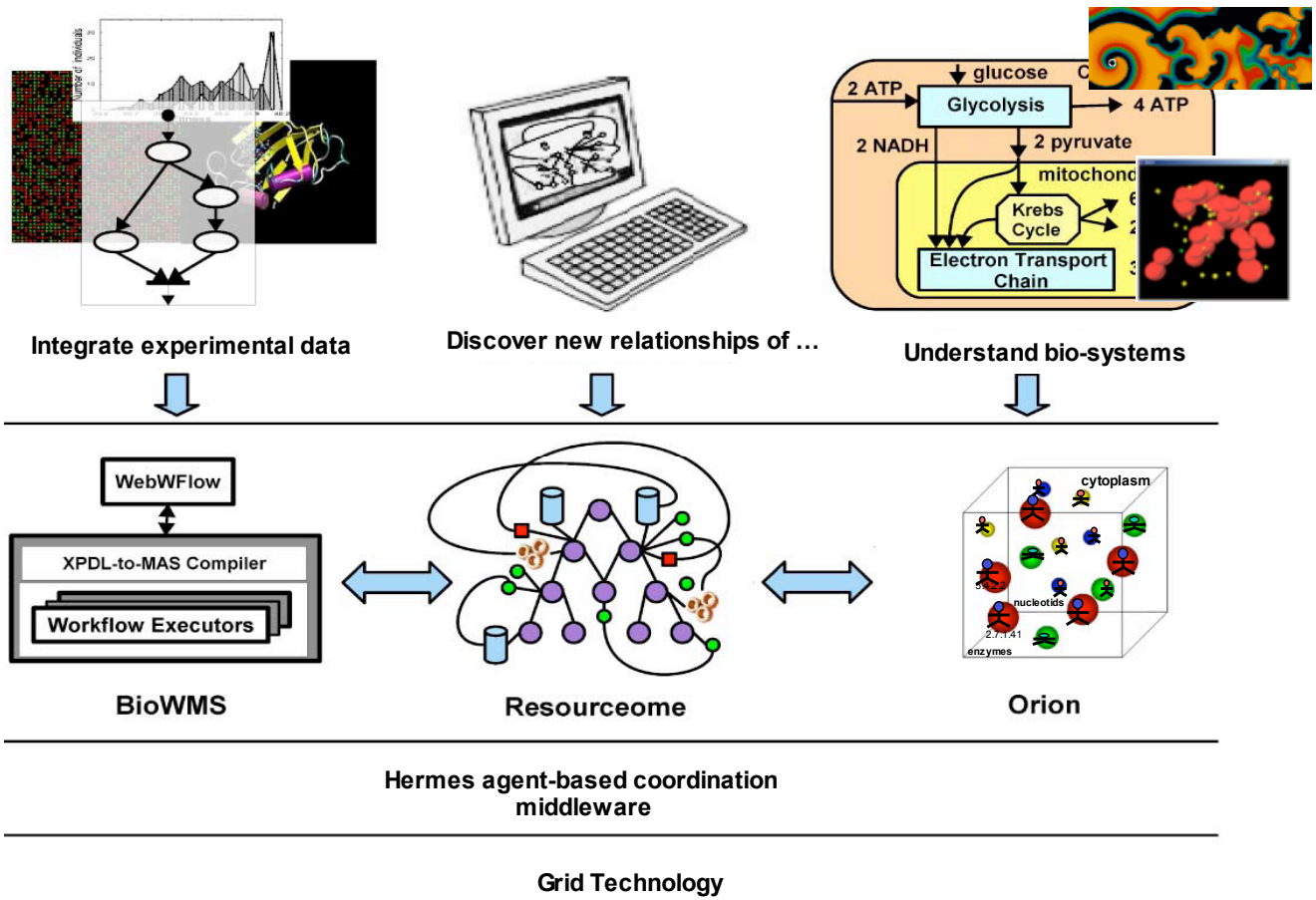
Computational systems biology is an emerging field in biological simulation that attempts to model or simulate intracellular and intercellular events using data gathered from genomic, transcriptomic, proteomic or metabolomic experiments.

Models (from cells to organisms) are used in **drug research** for understanding their effects and side-effects and for building personalized medicine

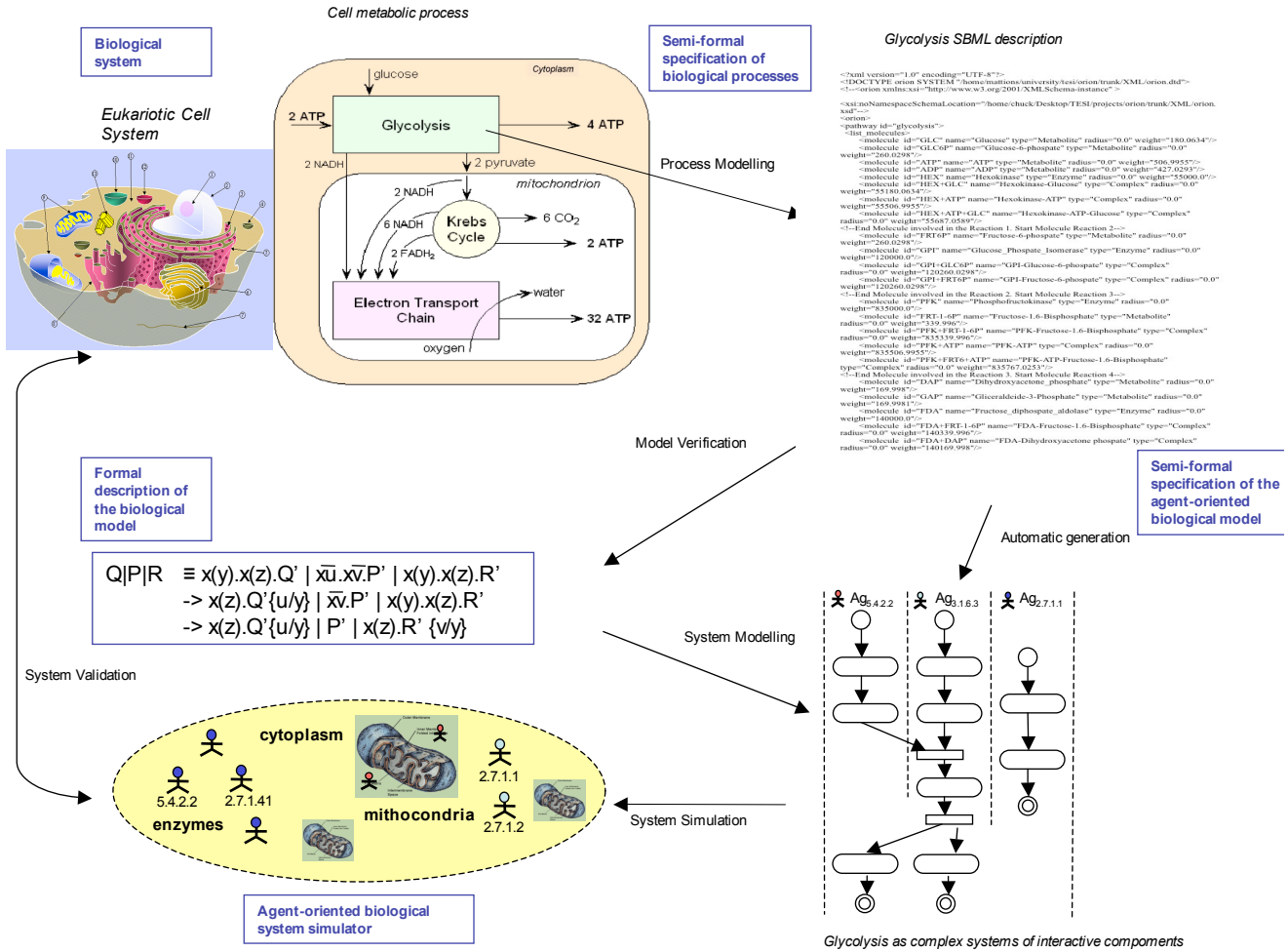
Computational Systems Biology typically requires tools to

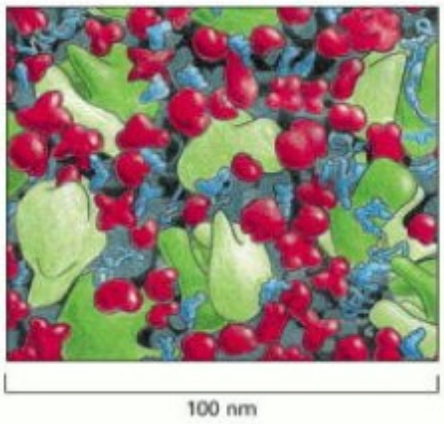
- **integrate** the huge amount of experimental data
- **understand** the behaviour of biological systems
  - by creating the model
  - by simulating the model
- **discover** new relationships between the various parts of a biological system, such as organelles, cells, organs, organisms
  - by perturbing the model of a biological system (biologically, genetically, or chemically)

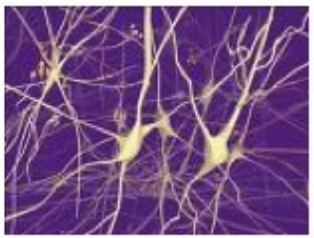
LITBIO project architecture

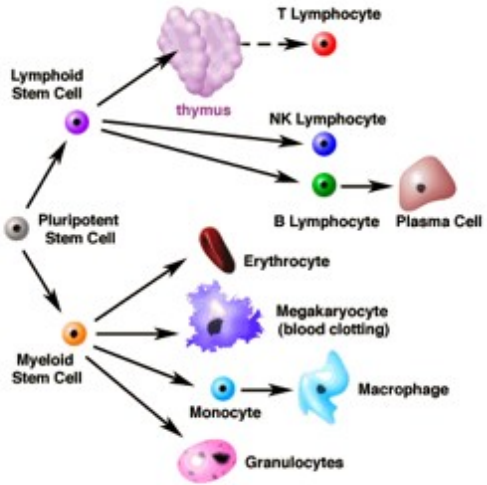


LITBIO project methodology



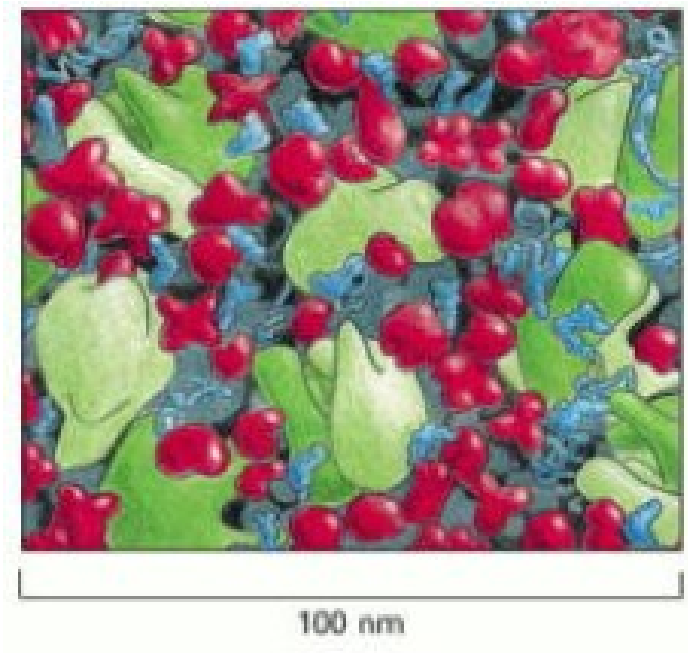


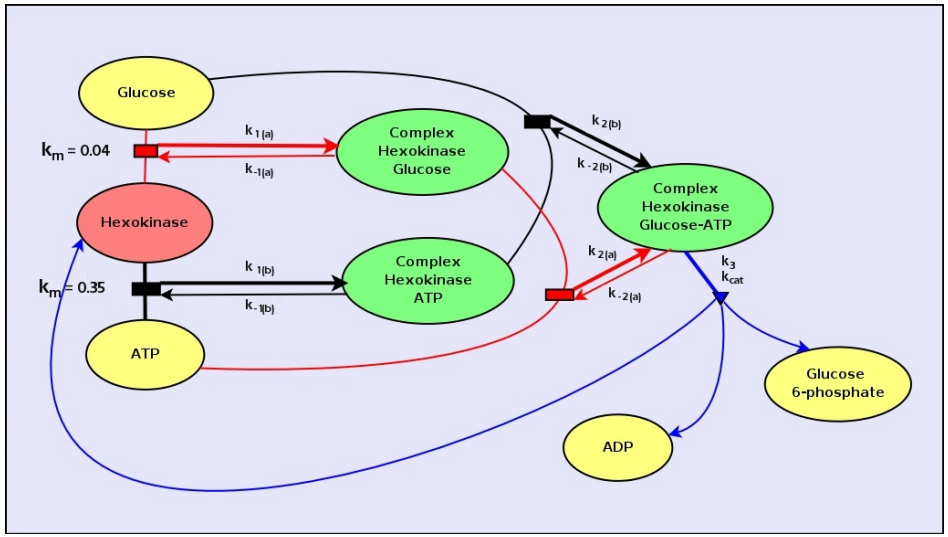




The global behaviour of the system can emerge from lower-level interactions among components

To model a portion (10^{-15} liters) of a **cytoplasm**, we want to simulate actions (**the movement** and the **interactions**) of about **2.7 millions of autonomous entities**.





How the combined actions of a large number of agents can lead to a coordinated behavior on the global scale?

- can a **Multiagent System** (MAS) help in designing these class of models?
- is MAS a promising approach for **multi-level** modelling?

  [*Multi-Level Modeling in Systems Biology by Discrete Event Approaches* A. Uhrmacher, C. Kuttler, IT-2006]
- is MAS a promising approach for **zooming-in** and **zooming-out** through models?

  [*Concurrency in Biological Modeling: Behavior, Execution and Visualization* D. Harel et al., to appear]

An agent is a computer system that is capable of *independent* action on behalf of its user or owner (figuring out what needs to be done to satisfy design objectives, rather than constantly being told)

[*An Introduction to MultiAgent Systems* by Michael Wooldridge, John Wiley & Sons, 2002]

A multiagent system consists of a number of agents interacting one-another.

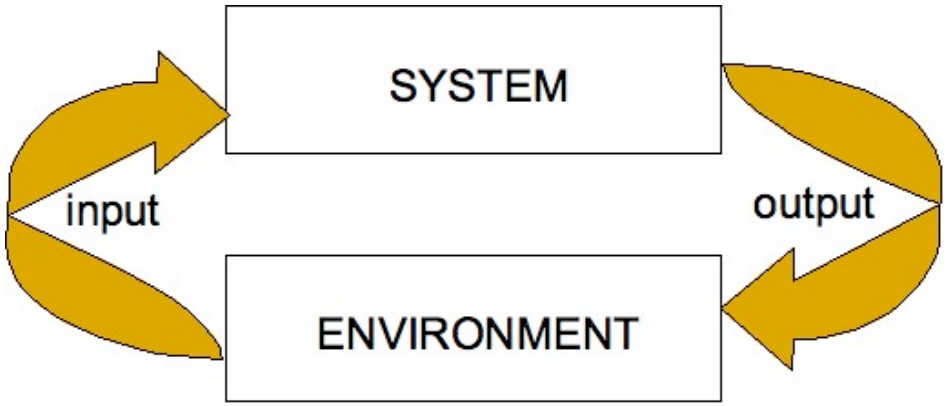
- In the most general, **agents will be acting on behalf of their users** with their different goals and motivations.
- To successfully interact, agents will require **the ability to cooperate, coordinate, and negotiate** with each other, much as people do.

[*An Introduction to MultiAgent Systems* by Michael Wooldridge, John Wiley & Sons, 2002]

A multi-agent system represents a new way of analysing, designing, and implementing complex software systems, . . . as those in biology.

An **agent** is a computer system **situated** in some environment and capable of **flexible**, **autonomous** action in that environment in order to meet its design objectives

[M. Wooldridge, *Agent-based software engineering*. In IEE Proceedings of Software Engineering, 1997]



is an agent an object?

is an agent an expert system?

is MAS just an AI project?

is MAS just distributed/concurrent systems?

An autonomous agent is a computer system capable of flexible, **autonomous** (problem-solving) action, situated in dynamic, open, unpredictable and typically multi-agent domains.

The agent has the control over its internal state and over its own behaviour

A situated agent is a computer system capable of flexible, autonomous (problem-solving) action, **situated** in dynamic, open, unpredictable and typically multi-agent domains.

The agent perceives the environment through sensors and acts through effectors

A flexible agent is computer system capable of **flexible**, autonomous (problem-solving) action, situated in dynamic, eventually open, unpredictable and typically multi-agent domains.

reactive: **the agent responds in timely fashion to environmental change**

adaptive: **the agent responds to the environmental change according to its internal state**

proactive: **the agent acts in anticipation of future goals**

A system is a pair containing an **agent** and an **environment**

Reactive agents can be defined by a 6-tuple

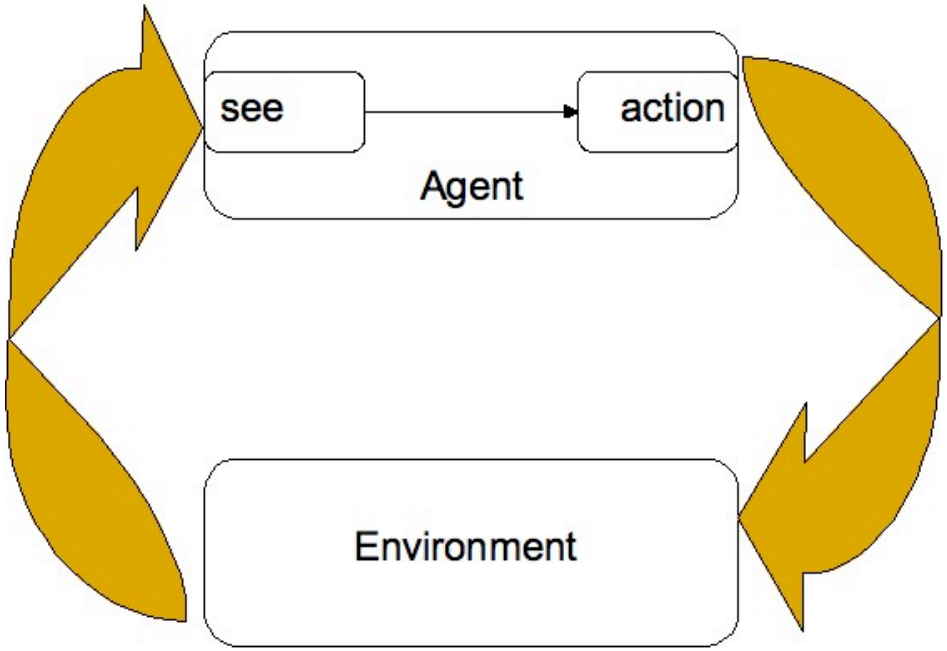
$$< E, P, A, see, do, action >$$

where

- E is the set of states for the environment
- P is a partition of E (representing the perception of the environment from the agent's point of view)
- A is a set of actions
- $see : E \rightarrow P$
- $action : P \rightarrow A$
- $do : A \times E \rightarrow E$

These agents observe the environment (*see*), find the appropriate action (*action*), and act (*do*) on the environment.

Reactive agent



can **a reactive agent** be considered as an ordinary Markov process?

and . . . can he be described by a finite state automaton (FSA)?

Adaptive agents can be defined by a 9-tuple

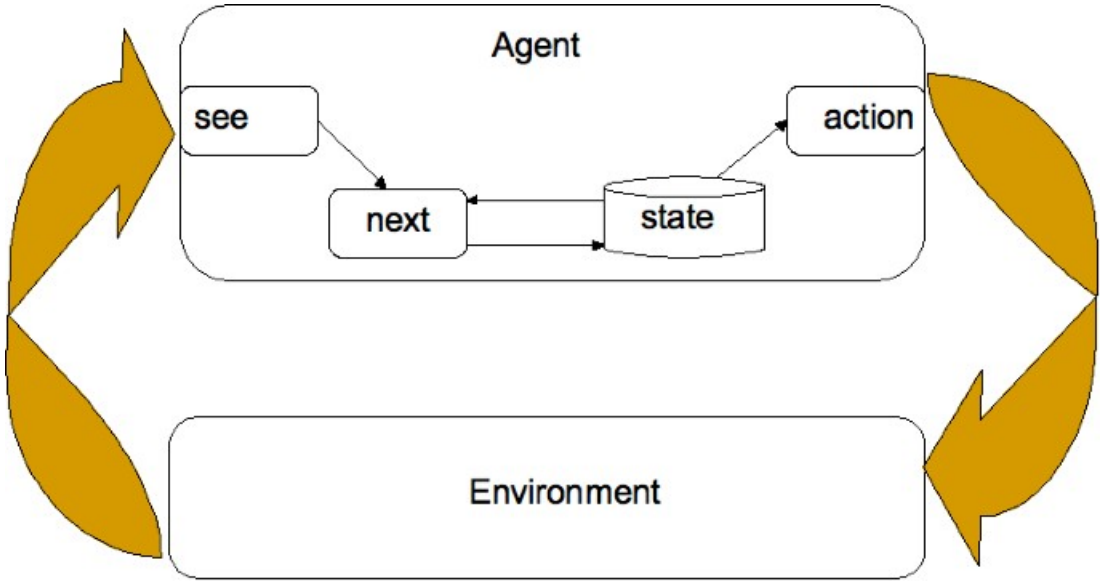
$$< I, E, P, A, i_0, see, next, do, action >$$

where

- I is a set of internal states
- E, P, A, see and do are the same as for reactive agents
- i_0 is the initial internal state
- $action : I \times P \rightarrow A$
- $next : I \times P \rightarrow I$

These agents observe the environment (see), choose their next action according to their internal state ($action$), act (do), and update their internal state (next).

Adaptive/proactive agent



Thus, a **simple adaptive agent** uses its memory to store observations of the state of the system and then it uses these observations to change its behavior.

can an **adaptive agent** be described as a push down automaton (PDA)?

and . . . can he be described as an Hidden Markov Model?

Proactive agents can be defined by a 9-tuple

$$< D, E, P, A, d_0, see, update, do, action >$$

where

- D is a set of predicate calculus (knowledge base)
- E, P, A, see and do are the same as for reactive agents
- d_0 is the initial predicate
- $action : D \times P \rightarrow A$
- $update : D \times P \rightarrow D$

These agents predicate observe the environment (*see*), choose their next action according to some kind of **reasoning** on their internal state (*action*), act (*do*), and update their knowledge base (*update*).

Agents vs process

1. Robin Milner in *CCS* and π-*calculus*
2. Rocco De Nicola in *Klaim*
3. David Harel in the *Statecharts*
4. Vincent Danos in *k*
5. ... and many others

Milner's agent point of view

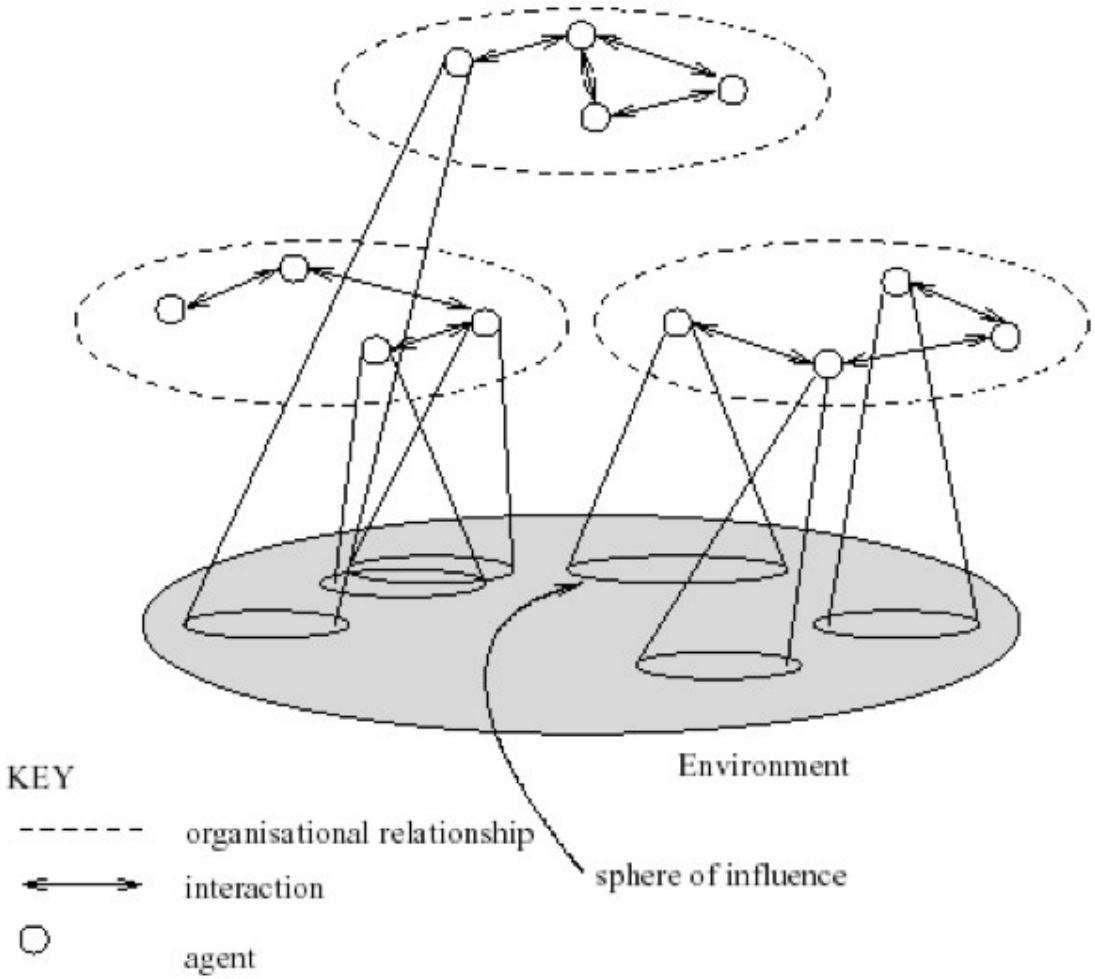
In 1989 Milner wrote ...

- ... Communication and concurrency notions, both essential in understanding complex dynamic systems. On the one hand such a system has diversity, being composed of **several parts** each acting concurrently with, and independently of, other parts; on the other hand a complex system has **unity** (or else we shall not call it a system) achieved through communication among its parts.
- ... underling both these notions is the assumption that each of the several parts of such a system has its own identity, which persists through time; we shall call these parts **agents**
- ... This lead to use the term **agent** very broadly, to mean any system whose behaviour consists of discrete actions; an which for one purpose we take to be atomic may, for other purpose, be decomposed into sub-agents acting concurrently and interacting.

Multi-agent system

A multiagent system consists of a collection of **interactive agents** and a **set of coordination rules**.

An agent has a name and a set of behaviours. He can adopts **different behaviours** at differents times depending on the environmental conditions.



[*An Introduction to MultiAgent Systems* by Michael Wooldridge, John Wiley & Sons, 2002]

A well-designed MAS is one that achieves a global task through the actions and interaction of its agents.

Main design steps consist:

- decomposing a global task into distributable subcomponents, yielding tractable tasks for each agent;
- establishing communication channels that provide sufficient information to each agent to achieve its task; and
- **coordinating the agents** in a way they cooperate on the global task, or at the very least, does not allow them to pursue conflicting strategies in trying to achieve their tasks.

The fundamental problem, in analyzing/designing such MAS, is in determining **how the combined actions** of a large number of agents, leads to **coordinated behavior** on the global task.

"... Coordination is the process of building programs by gluing together active pieces"

[*Coordination languages and their significance,* Carriero and Gelernter, Communication of ACM, 1992]

"... Coordination is managing dependencies between activities"

[*The interdisciplinary study of coordination,* Malone and Crowston, ACM Computing Survey, 1994]

"**A coordination model** is the glue that binds separate activities into an ensemble"

Coordination models

- **Message passing models** allow to coordinate processes that can communicate with other processes through channels or ports on which messages are sent and received
- **Tuple space models** allow to coordinate processes through a common space used to communicate in an temporal and referential uncoupled manner

LINDA: a coordination model based generative communication

Coordination languages

- A coordination language is the linguistic support that includes the coordination model
- A coordination language includes clearly defined mechanisms for communication, synchronization, distribution, and concurrency control

KLAIM: a kernel language for agents interaction and mobility

LINDA model

Based on generative communication: processes (agents) communicate by inserting or retrieving data objects (called Tuples) from shared Data Space (called Tuple space)

Operations

- out(t) insert a tuple t into the Tuple space (non-blocking)
- in(t) find and remove a *matching* tuple from the tuple space; block until a matching tuple is found
- rd(t) like in(t) except that the tuple is not removed
- eval(t) add the active tuple t to the tuple space

The model is structurally recursive: a tuple space can be one field of a tuple.

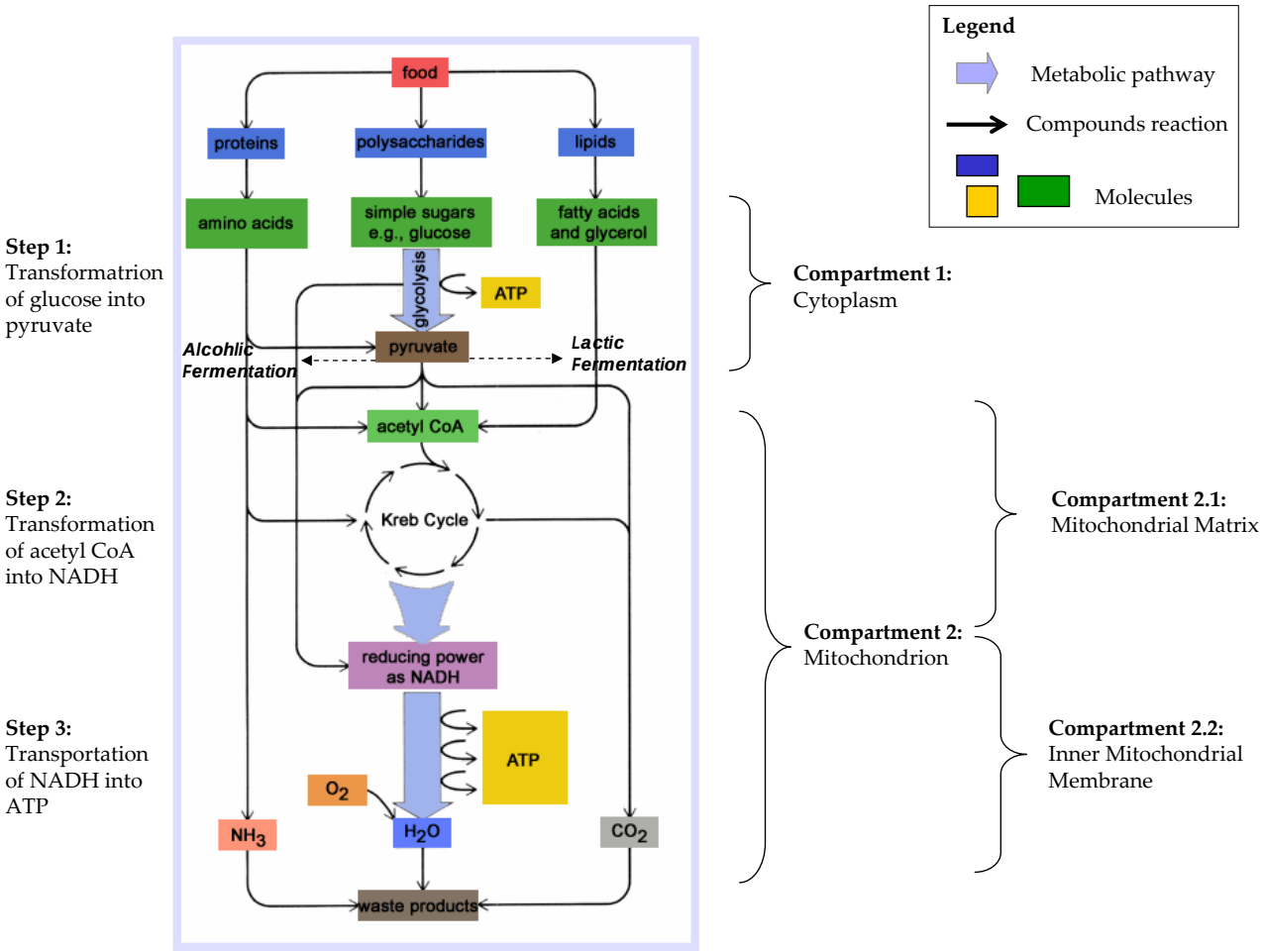
KLAIM language

Klaim is a simple formalism that can be used to model and program mobile agent-based systems:

- Klaim allows programmers to concentrate on the distributed structure of programs while ignoring their precise physical allocations
- Klaim is based on process algebras, but makes use of Linda-like asynchronous communication and models distribution via multiple shared tuple spaces
- Klaim Tuple Spaces and agents are distributed over different localities and the classical Linda operations are indexed with the location of the tuple space they operate on

Our experience

Carbohydrate Oxidation Cellular Process



Cell metabolism is the process by which individual cells process nutrient molecules.

The series of chemical reactions within a cell that realise cell metabolism are called **metabolic pathways**.

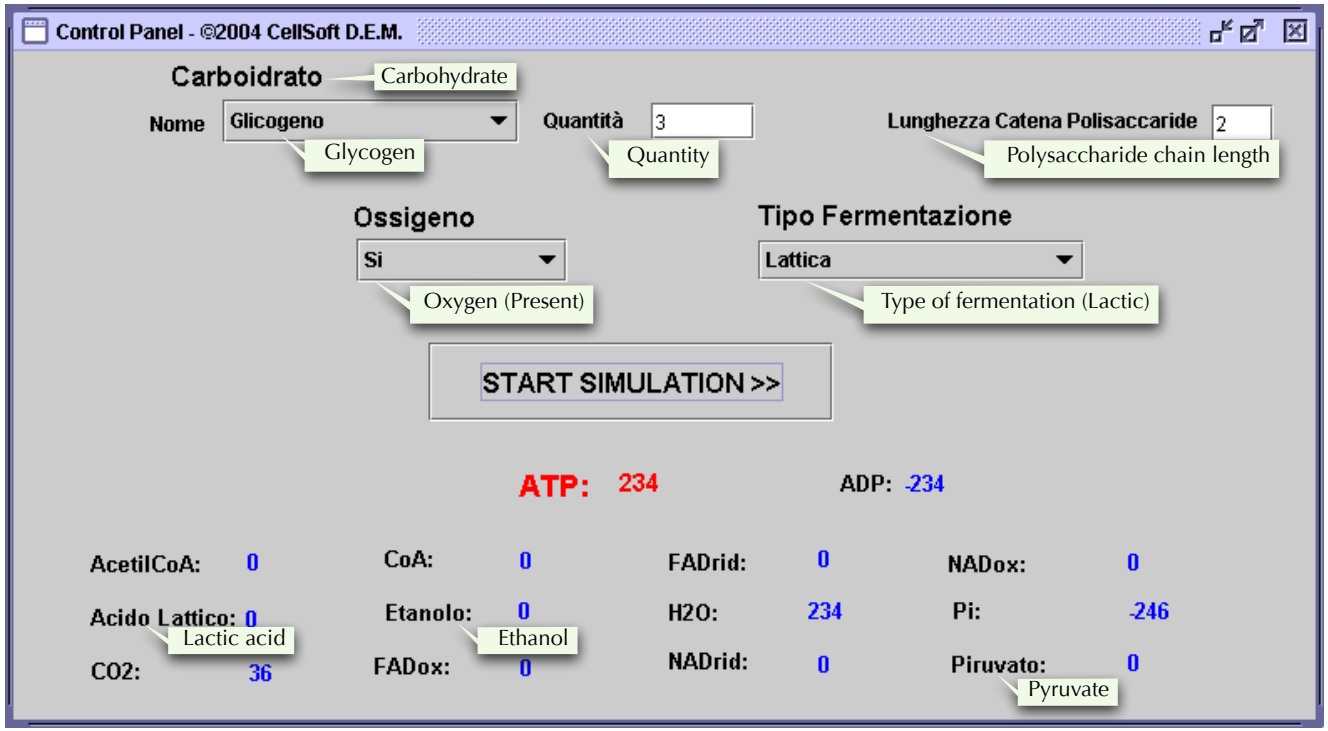
Carbohydrate oxidation is a set of metabolic pathways by which a cell produces energy through chemical transformation of carbohydrates like fructose, glucose, mannose, maltose, lactose, saccharose, glycogen and starch

During the carbohydrate oxidation process, three metabolic pathways are active:

- **Glycolysis** converts one molecule of glucose into two molecules of pyruvate.
- **Lactic fermentation**, in absence of oxygen, reduces the pyruvate to lactate.
- **Alcoholic fermentation**, in the absence of oxygen, reduces pyruvate to ethanol.

Lactic fermentation occurs in anaerobic microorganisms (sour milk) and animal cells.

First Simulator: Carbohydrate Oxidation pathway

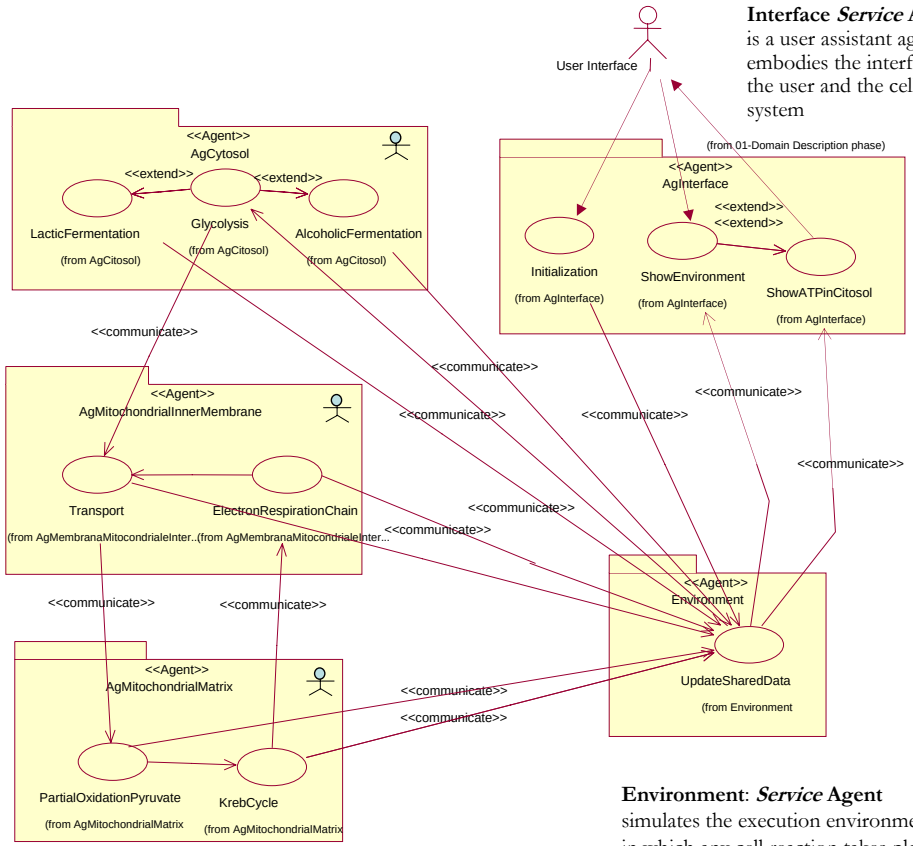


Carbohydrate oxidation: the agents identification UML diagram

Agent Citosol simulates the role of Cytoplasm in the Glycolysis, Lactic and Alcoholic Fermentations functions

Agent Inner Mithocondrial Membrane simulates the role of IMM in the Transport and Electron Respiration Chain functions

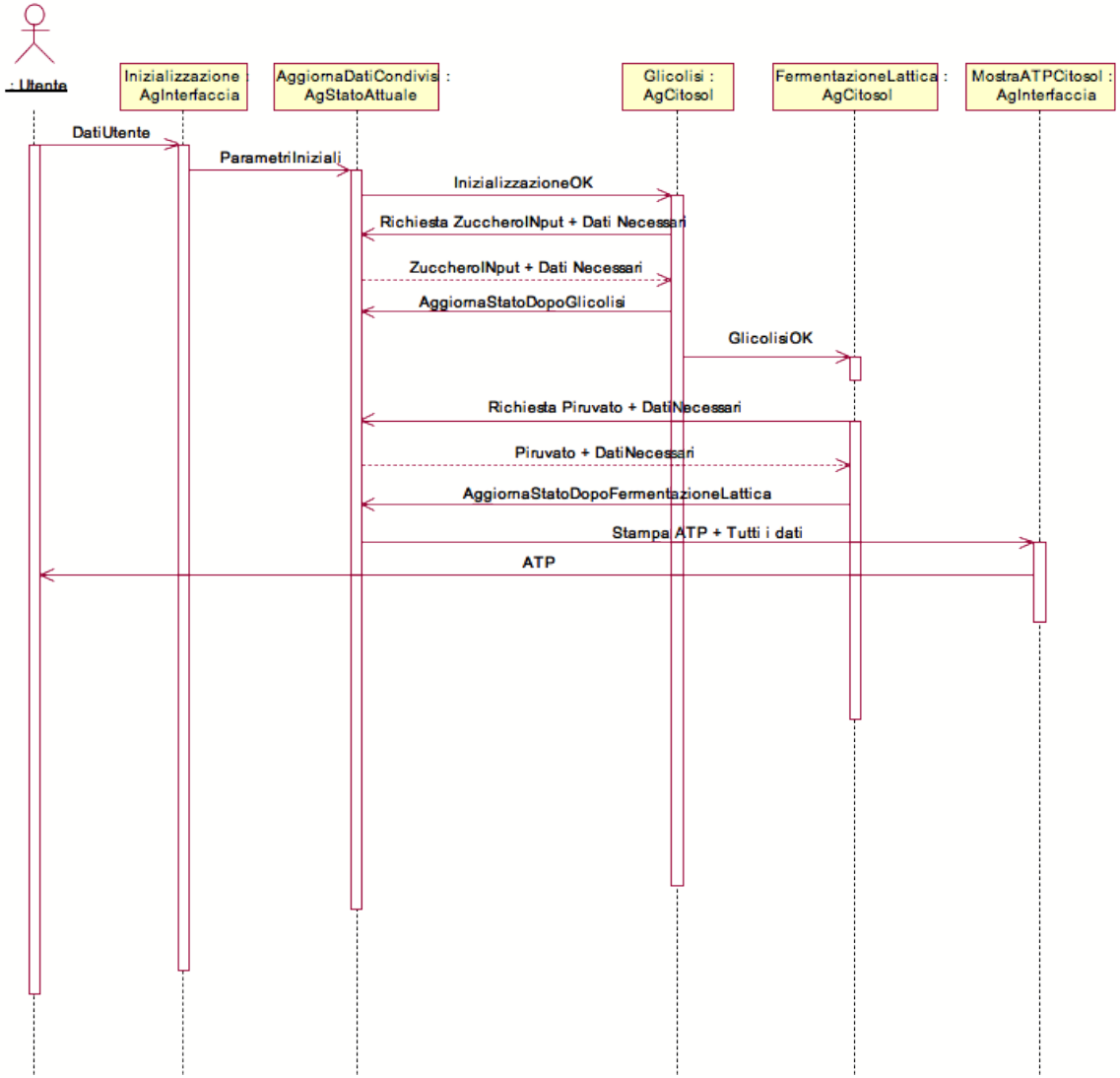
Agent Mithocondrial Matrix: simulates the role of MM in the Partial Oxidation of Pyruvate and Kreb's Cycle functions…



Interface *Service* Agent is a user assistant agent. It embodies the interface between the user and the cell simulation system

Environment: ***Service*** **Agent** simulates the execution environment in which any cell reaction takes place

Agent roles and interaction in the Lactic Fermentation function



Model validation

The **validation of the model**, based on **mutation testing**, imposes a new requirement on the model:

the model simulation should give reasonable results when the model is altered in ways that correspond to known or **plausible mutations**

MAS Valitation by mutation

Carbohydrate oxidation can take place within two different environmental conditions:

- in presence of oxygen (aerobic) or
- in its absence (anaerobic)

The former takes place in the mitochondria, the latter in the cytoplasm.

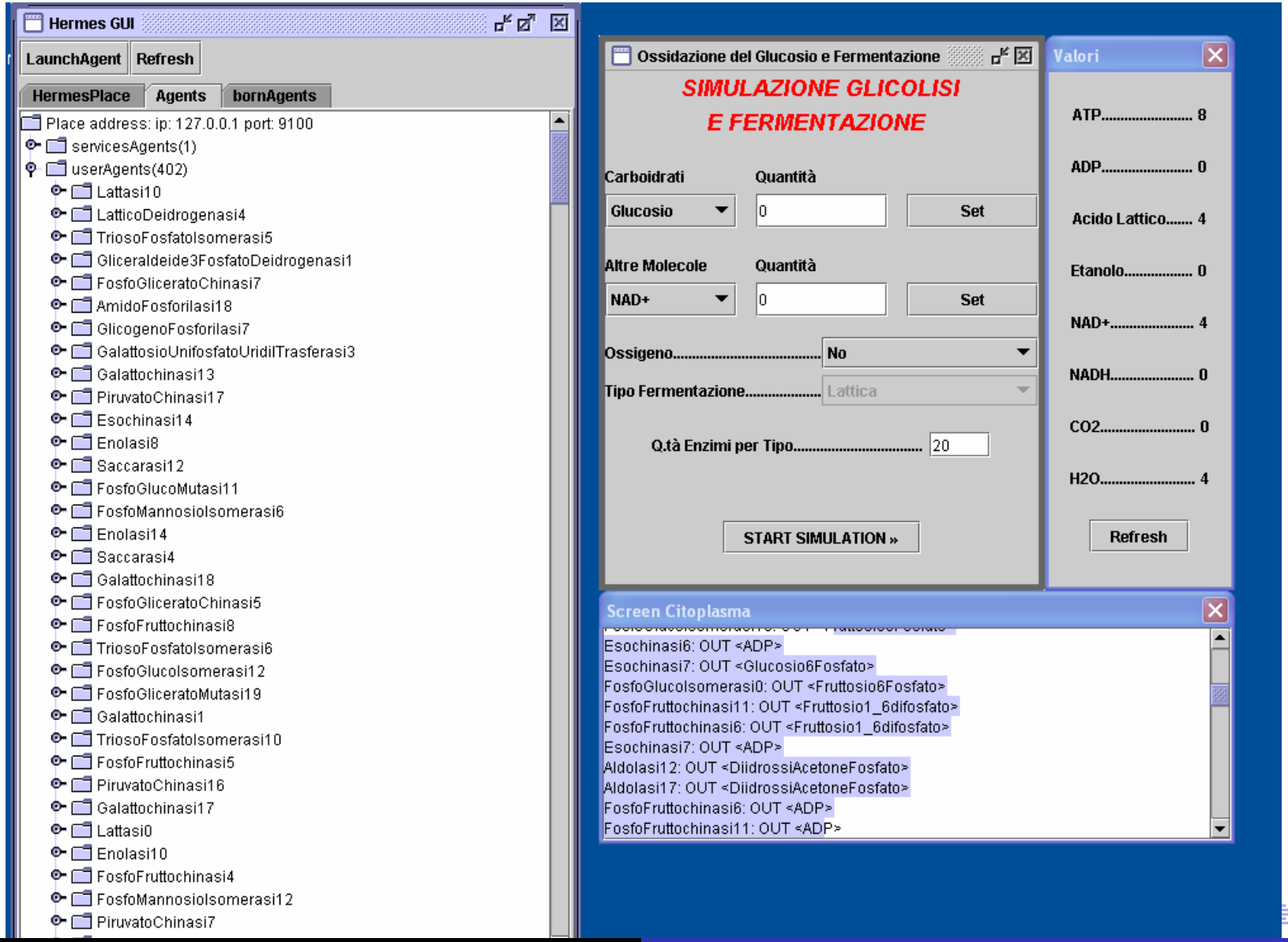
There are some cases where as consequence of malfunction due to **DNA mutation in the mitochondrion** the aerobic pathway is blocked and metabolism is forced to change behavior with respect to new condition.

A mitochondrion DNA mutation can lead to **a lack or disappearance of an enzyme** involved in a metabolic pathway. It can alter the pathway to producing ATP or, in the worst case, block the process.

In nature, an aerobic microorganism can become anaerobic as a consequence of a mutation which prevents use of cellular respiration in mitochondria

We need to model the Carbohydrate Oxidation at fine grain abstraction!

Second Simulator: Glycolysis pathway



Glycolysis Pathway

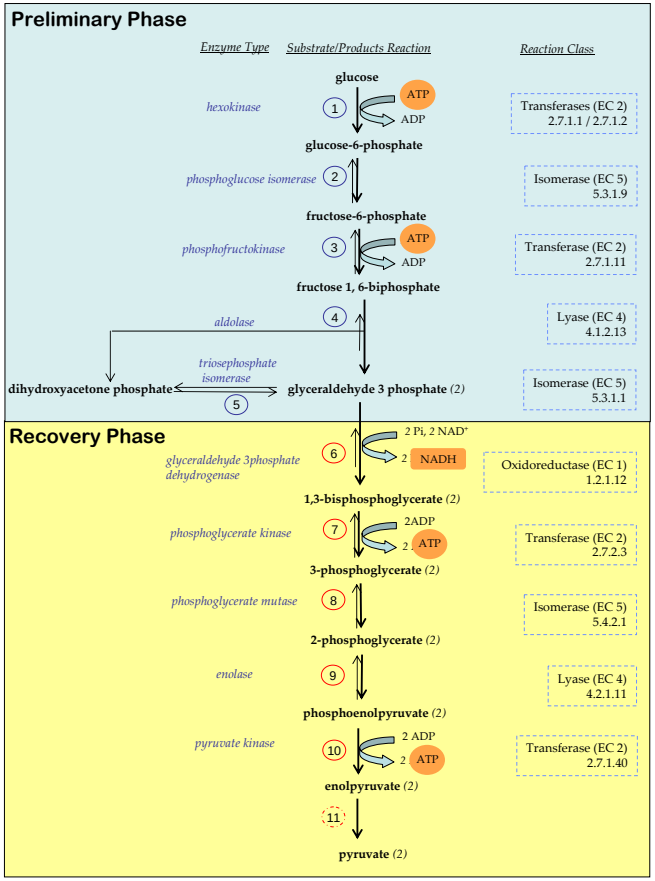
- An ubiquitous pathway in life: produce ATP from Glucose
- A pathway is composed by a series of reactions in wich **metabolites** and **enzymes** are involved

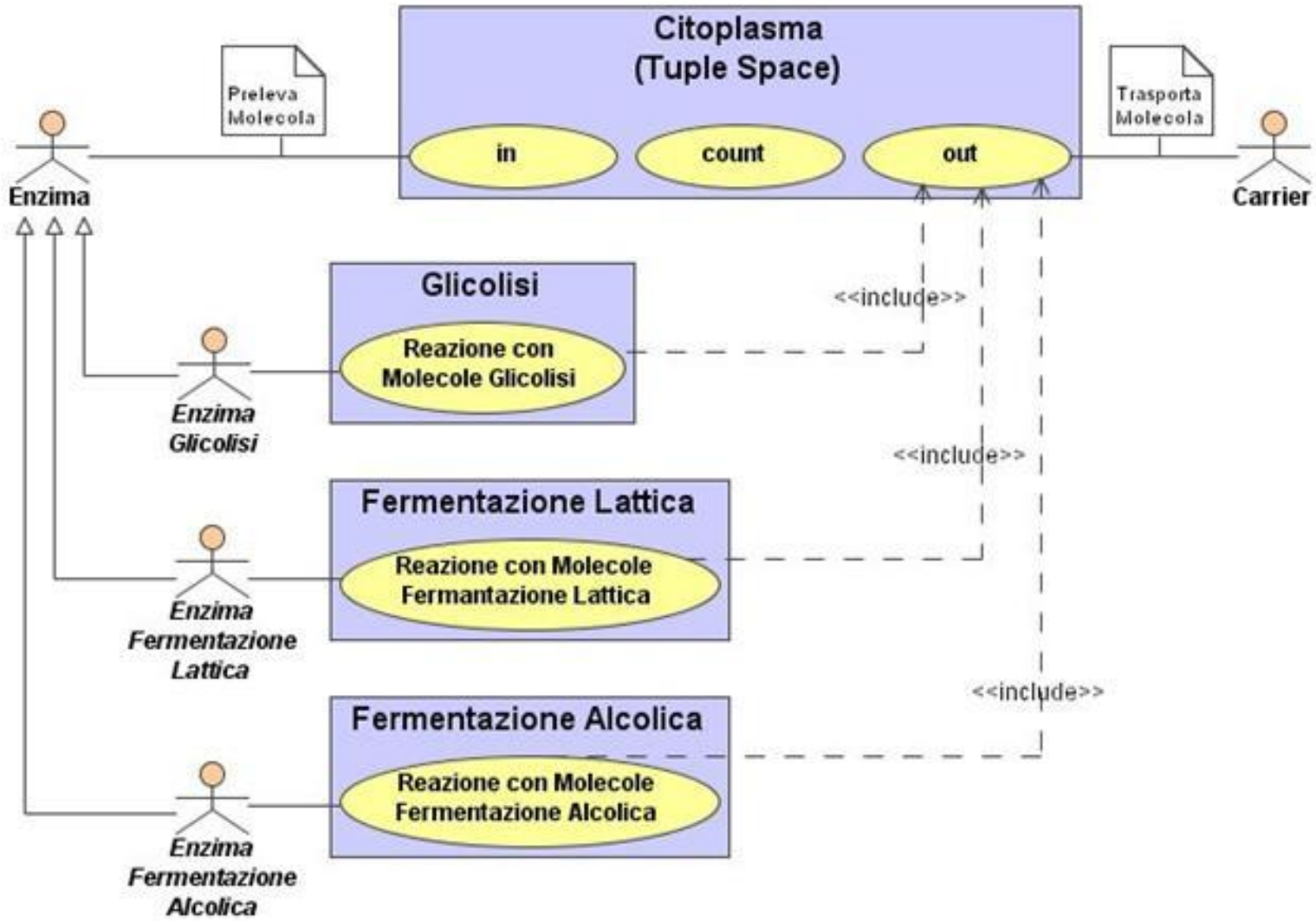
$$\text{Step 1: Hexokinase} + \text{Glucose} + \text{ATP} \longrightarrow \text{Glucose 6-Phos.} + \text{ADP}$$

- Hexokinase is an enzyme which promotes the reaction between Glucose and ATP

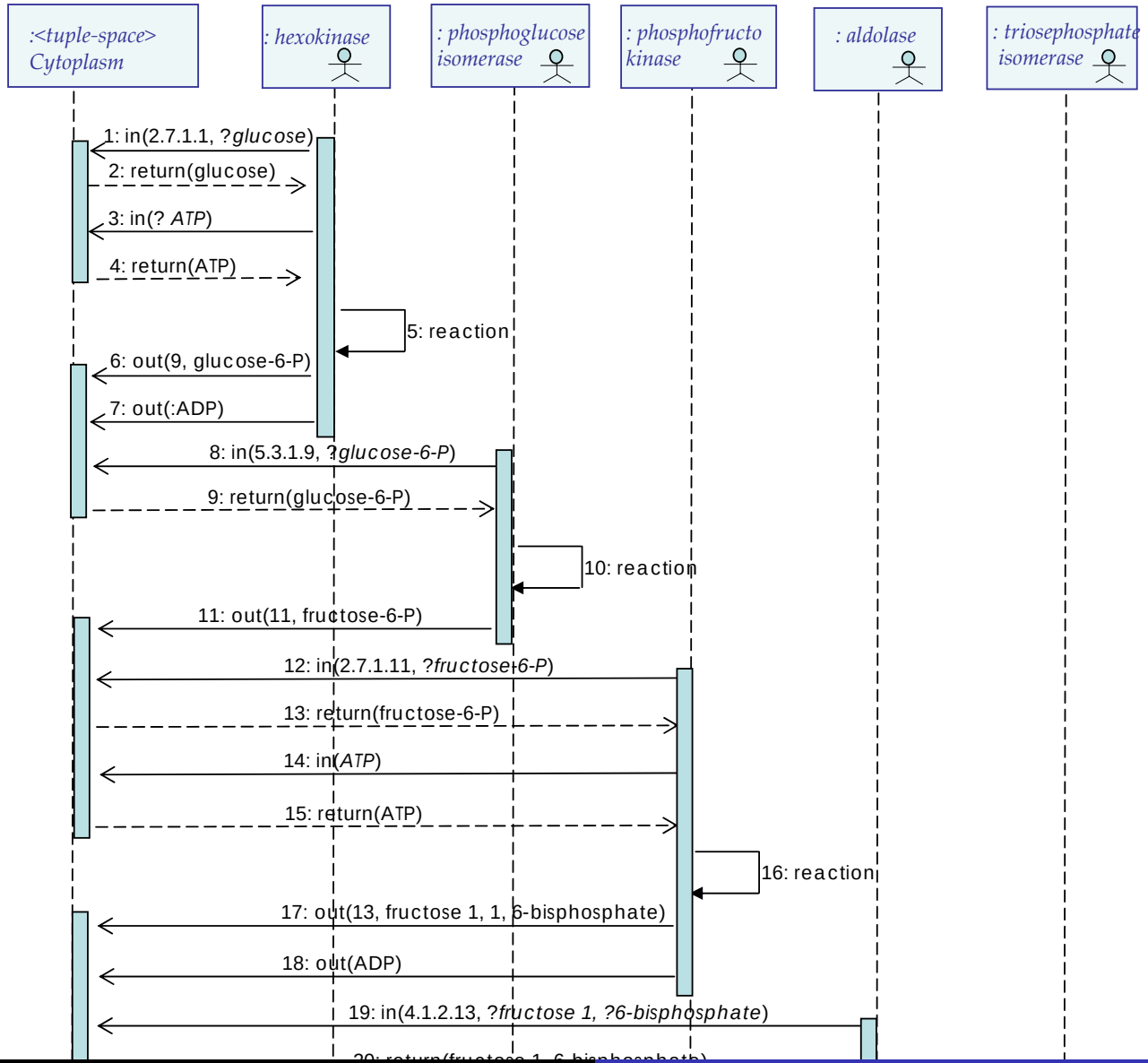
Glycolysis metabolic pathway

$$glucose + 2NAD^{+} + 2ATP + 2ADP + 2Pi \rightarrow 2pyruvate + 2NADH + 4ATP$$



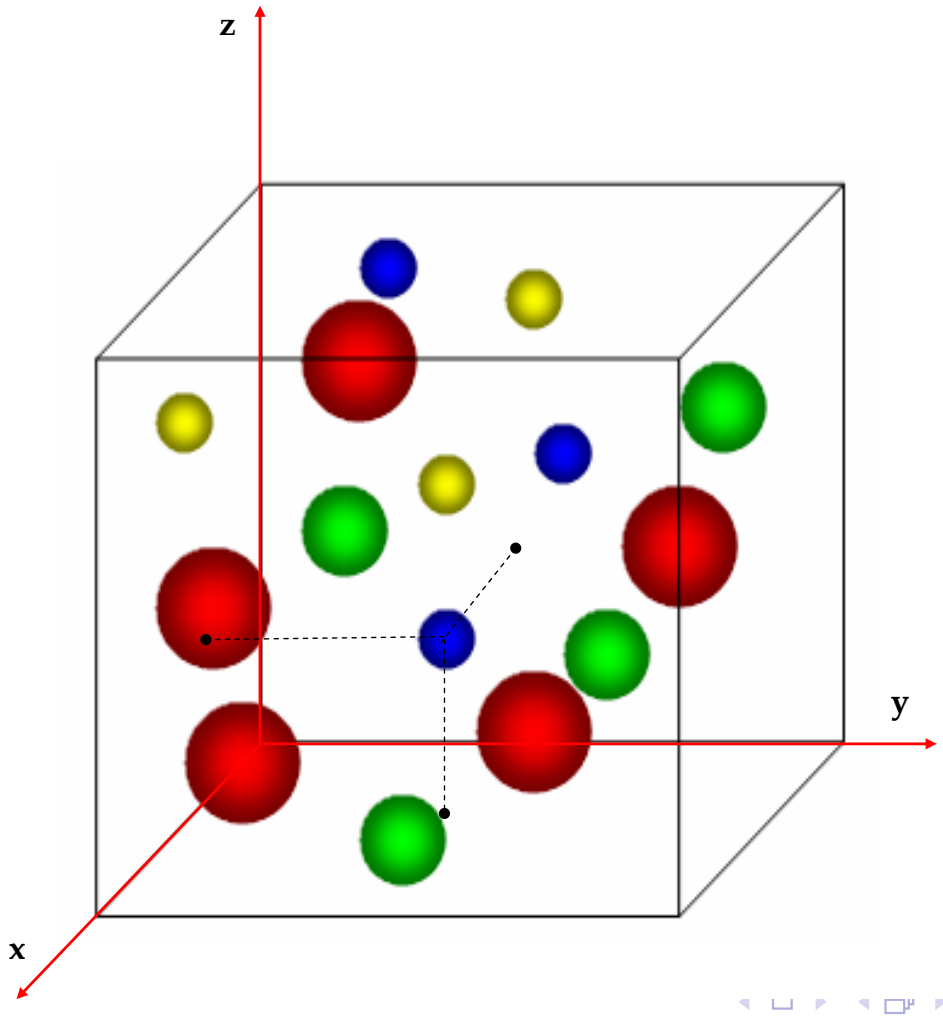

Citoplasma
(Tuple Space)
in
count
out
Preleva
Molecola
Trasporta
Molecola
Enzima
Carrier
Glicolisi
Reazione con
Molecole Glicolisi
<<include>>
Enzima
Glicolisi
Fermentazione Lattica
Reazione con Molecole
Fermantazione Lattica
<<include>>
Enzima
Fermentazione
Lattica
Fermentazione Alcolica
Reazione con Molecole
Fermentazione Alcolica
<<include>>
Enzima
Fermentazione
Alcolica

UML Sequence diagram of the preliminary phase of glycolysis

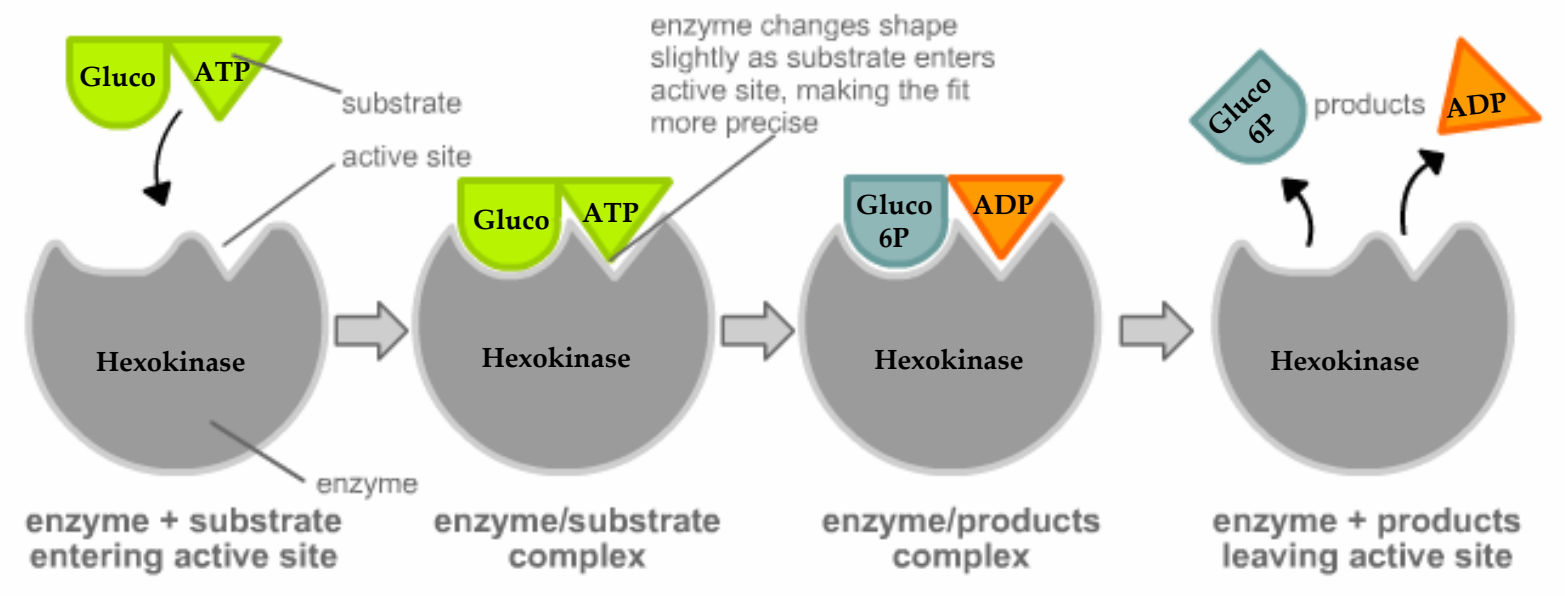


Model validation

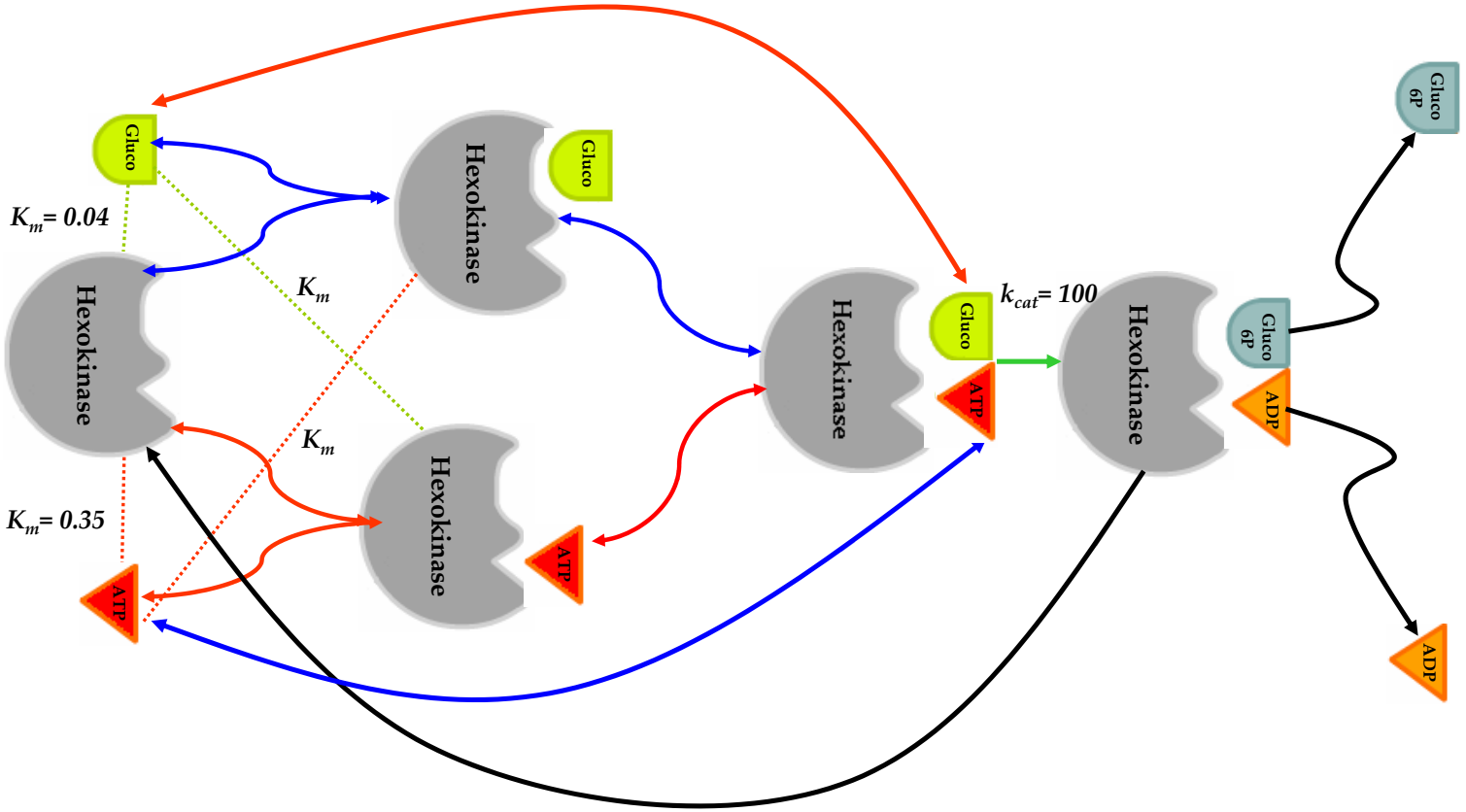
But ... molecules move



... and then interact by vicinity



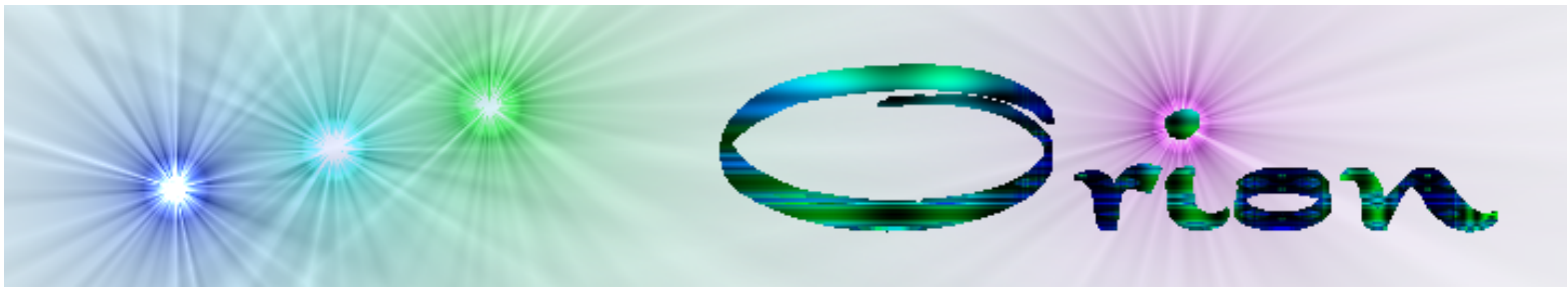
... and then interact by affinity



and . . . molecules have also their volume and many other parameters . . .

Third simulator: Orion

- Orion is a MAS simulator over Hermes (an agent-based coordination middleware)
- Orion agents define a virtual **space** and **time** where **biological** entities can **move** and **interact**



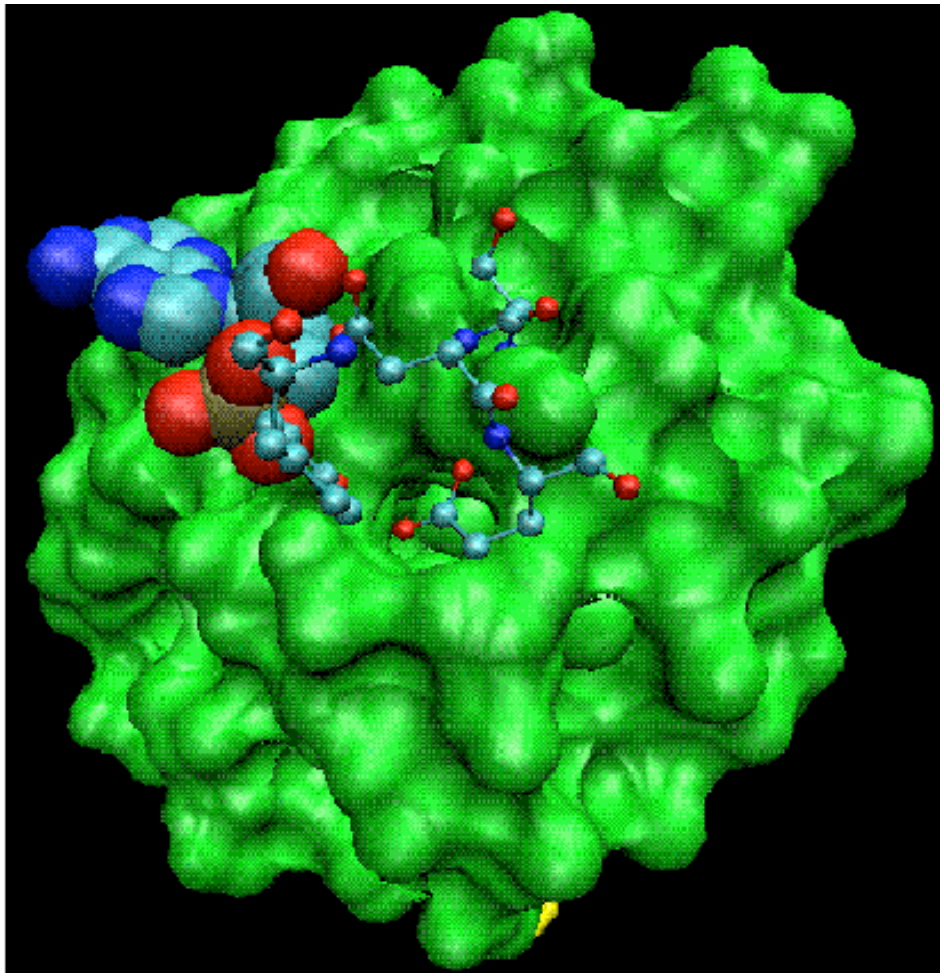
System behavior emerging from components behavior

- Orion adopt a bottom-up strategy: model small components as autonomous Agents and make them live in a **concurrent** environment with **random** interactions
- Orion supports experimental model validation: correctness of the model (through agents behavior)
- and . . . theoretical validation: accuracy of the approach (concurrent stochastic interactions and movements)

Glycolysis case study: complexes and reactions

- An enzyme binds its substrate with a given affinity, becoming a **Complex**
- When all of the substrate is bound, the reaction happens at the given velocity: products are released and the enzyme is ready for another binding
- Enzymes have **affinity constants** (K_m) for each substrate they can bind to
- The quantity of products in the time unit is a function of the concentrations and an **enzyme-constant** (K_{cat})

A Complex



Motion of the molecules

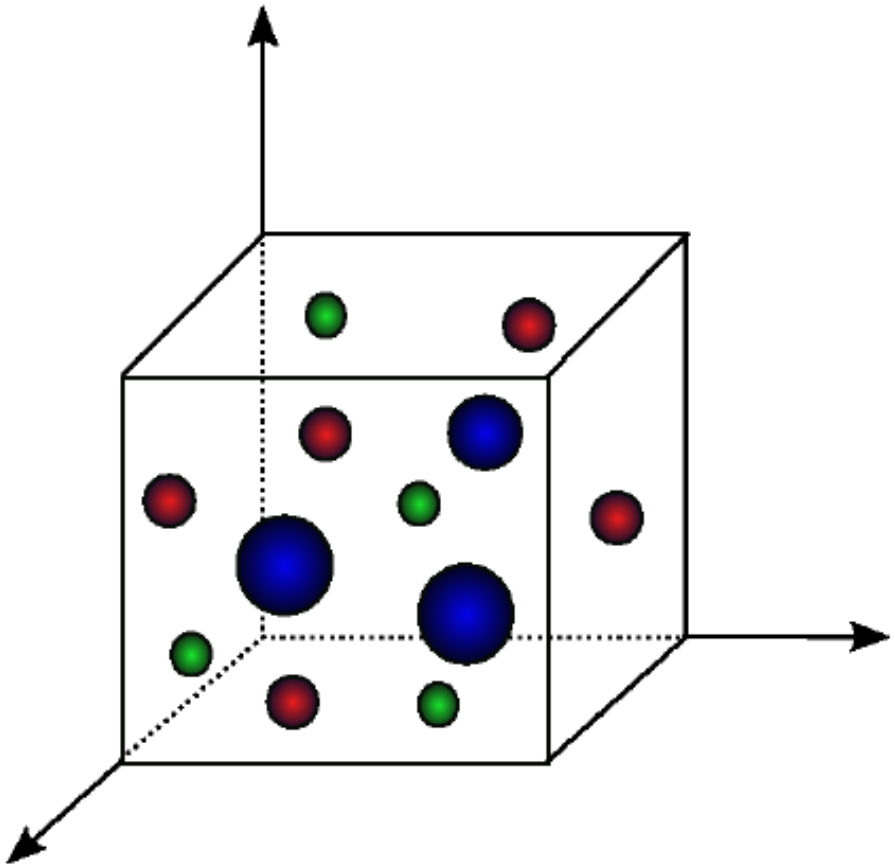
- We model the cell at mesoscale (10^{-8}) level
- At this scale the predominant force is Brownian Motion
- The molecules move in random directions with a calculated offset (Einstein relation)

See simulation... (brownian.html)

Volume of simulation

- We simulate the glycolysis pathway in a cube of cytoplasm of a yeast cell
- Temperature is considered constant at 25°C
- Cytoplasm viscosity is considered
- Uniform distribution of the molecules in the cytoplasm is assumed
- Molecules are represented as 3D spheres with a radius proportional to their weight
- A reasonable choice of volume is the femtolitre: $10^{-15}\ell = 10^{-18}m^3$ (the cube side is $10^{-6}m$)

Cube of simulation



Orion Project Design

- Model Enzymes and Complexes as Hermes Agents
- Model Metabolites as Java Objects
- Define ServiceAgents for managing space and time
- All move according to Brownian Motion
- Enzymes and Complexes perceive metabolites in their action radius
- Probabilistically (using the affinities K_m) Enzymes and Complexes bound metabolites
- Complexes react producing new metabolites in a given time (inverse of K_{cat}) and become Enzymes again

Getting biological data

- Orion uses an XML database (eXist) to store the biological data
- Biological data about the pathway and the molecules are stored in SBML: Systems Biology Markup Language - `www.sbml.org`
- Data for different pathways and molecules are easy to import

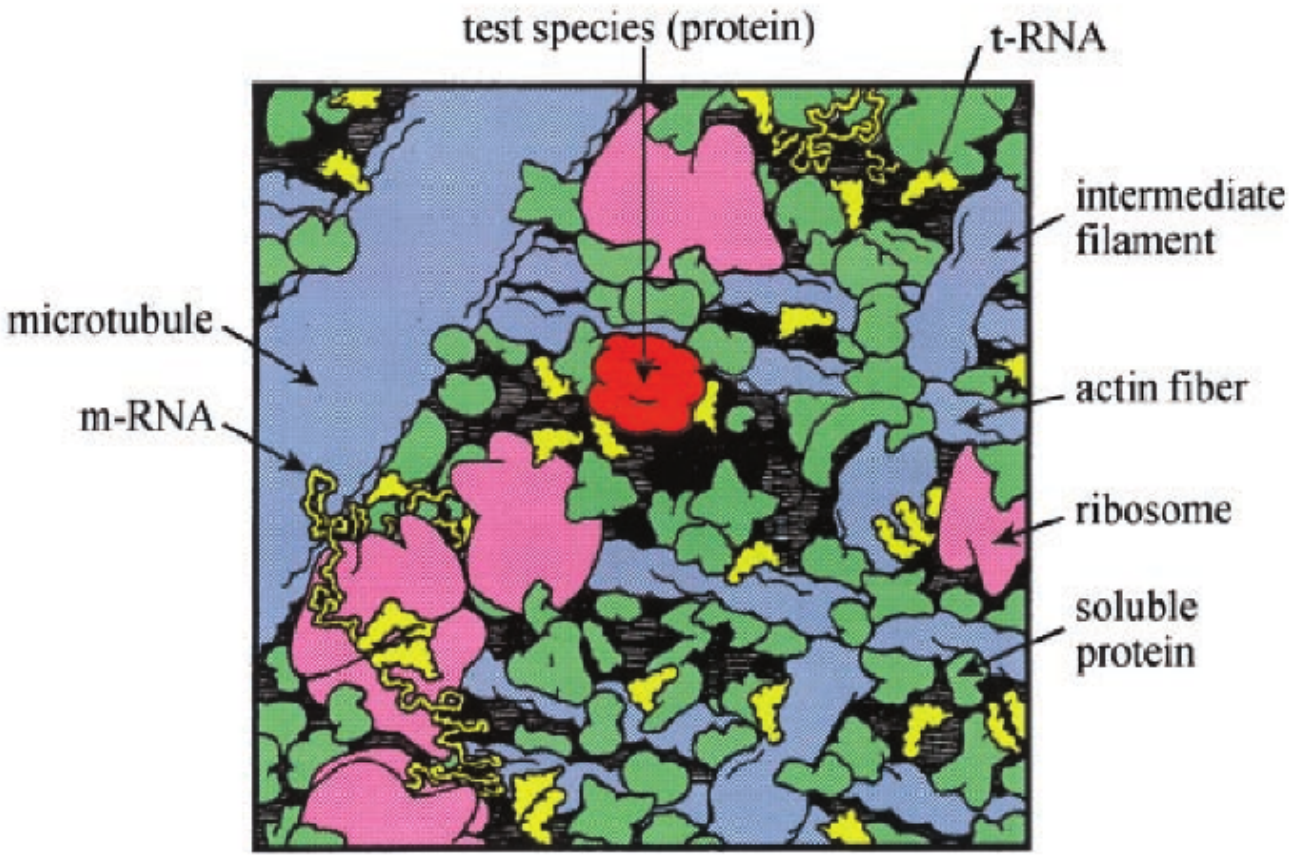
A sample of the SBML description

```
<pathway organism=Saccharomyces_cerevisiae
id=glycolysis>
 <list_molecules> .... </list_molecules>
 <list_reactions>
  <reaction kcat=50.0>
   <list_interactions>
    <interaction>
          <reactants>
           <reactant id=GLC/>
           <reactant id=HEX/>
          </reactants>
          <products>
        <product id=HEX+GLC/>
          </products>
     <Km_constant>0.028</Km_constant>
    </interaction>
    <interaction>
          <reactants>
           <reactant id=ATP/>
           <reactant id=HEX/>
          </reactants>
          <products>
        <product id=HEX+ATP/>
          </products>
     <Km_constant>0.05</Km_constant>
```

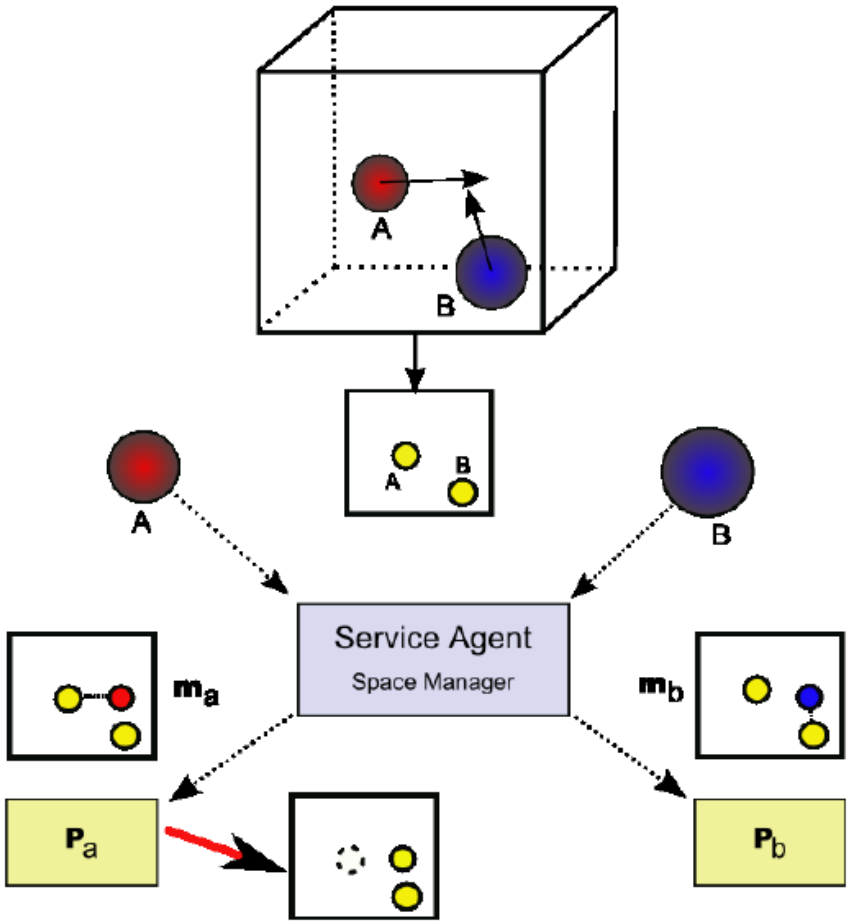
Agents

- Enzymes and Complexes are Agents
- Metabolites are Objects whose movement is managed by a ServiceAgent
- **Space** occupation and movements are managed by a ServiceAgent (bumps currently not permitted) which resolves conflicts randomly
- **Time** impulses (1 tick is 1 millisecond, for glycolysis) are given by a ServiceAgent when **all** the agents and metabolites have finished their moves and reactions in the current time slice

Intracellular Concentration



Space Manager



Dimensions of the simulation

- In the simulation volume the higher and dominant concentration is those of ATP
- Using bio-data we calculate approximatively 2 700 000 ATP molecules in the space
- They have to be maintained in memory and moved according to Brownian Motion in each time slice
- Moreover other metabolites have to be managed
- Enzymes and Complexes also compete for space
- A standalone workstation cannot perform the simulation in a reasonale space and time

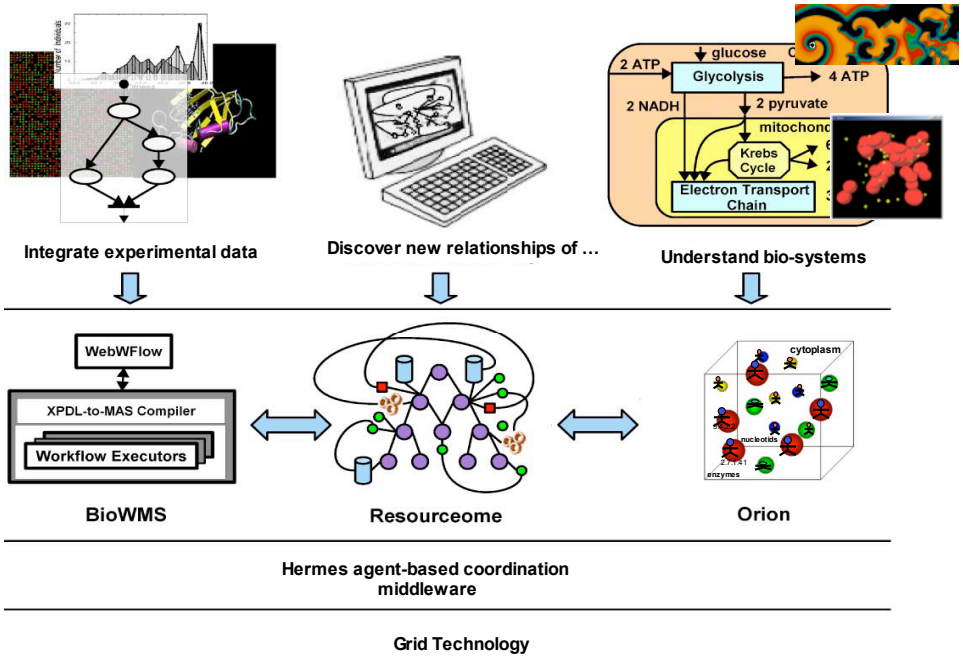
Cube splitting

- Our solution (work in progress) is to decompose the cube of simulation in smaller cubes
- Each cube manages a slice of the virtual space
- Cube splitting must be transparent to Agents
- Cubes boundaries are guarded by ServiceAgents who implement the transparent splitting
- Coordination models and languages are needed to achieve the goal

Deployment

- LitBio: Laboratory of Interdisciplinary Technologies in Bioinformatics - `litbio.cs.unicam.it`
- Collaborating with Cilea (`www.cilea.it`)
- Provides (work in progress) a **GRID** solution for Orion
- Each small cube is managed by an instance of Orion in a GRID node
- Coordination models guarantee the correctness and consistency of the simulation

LITBIO Project: Unicam Framework Architecture



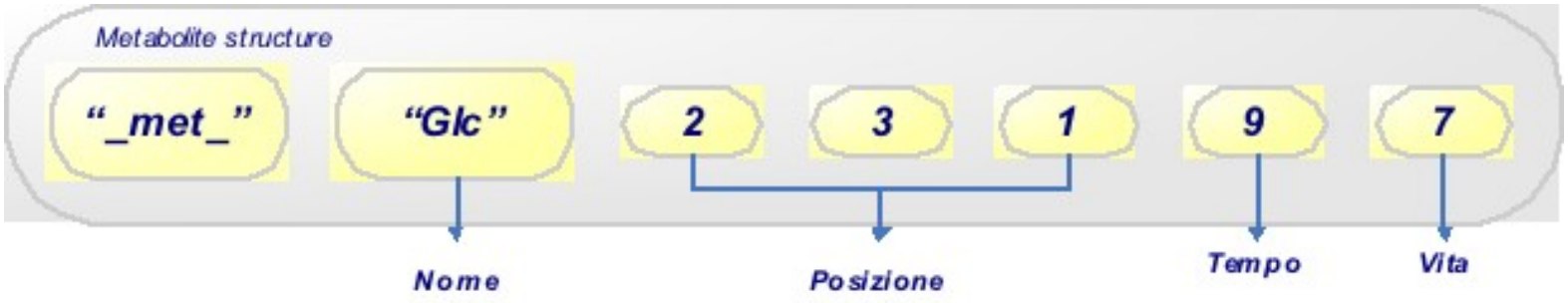
[litbio.cs.unicam.it]

Get another simulator . . . **Kalliope**

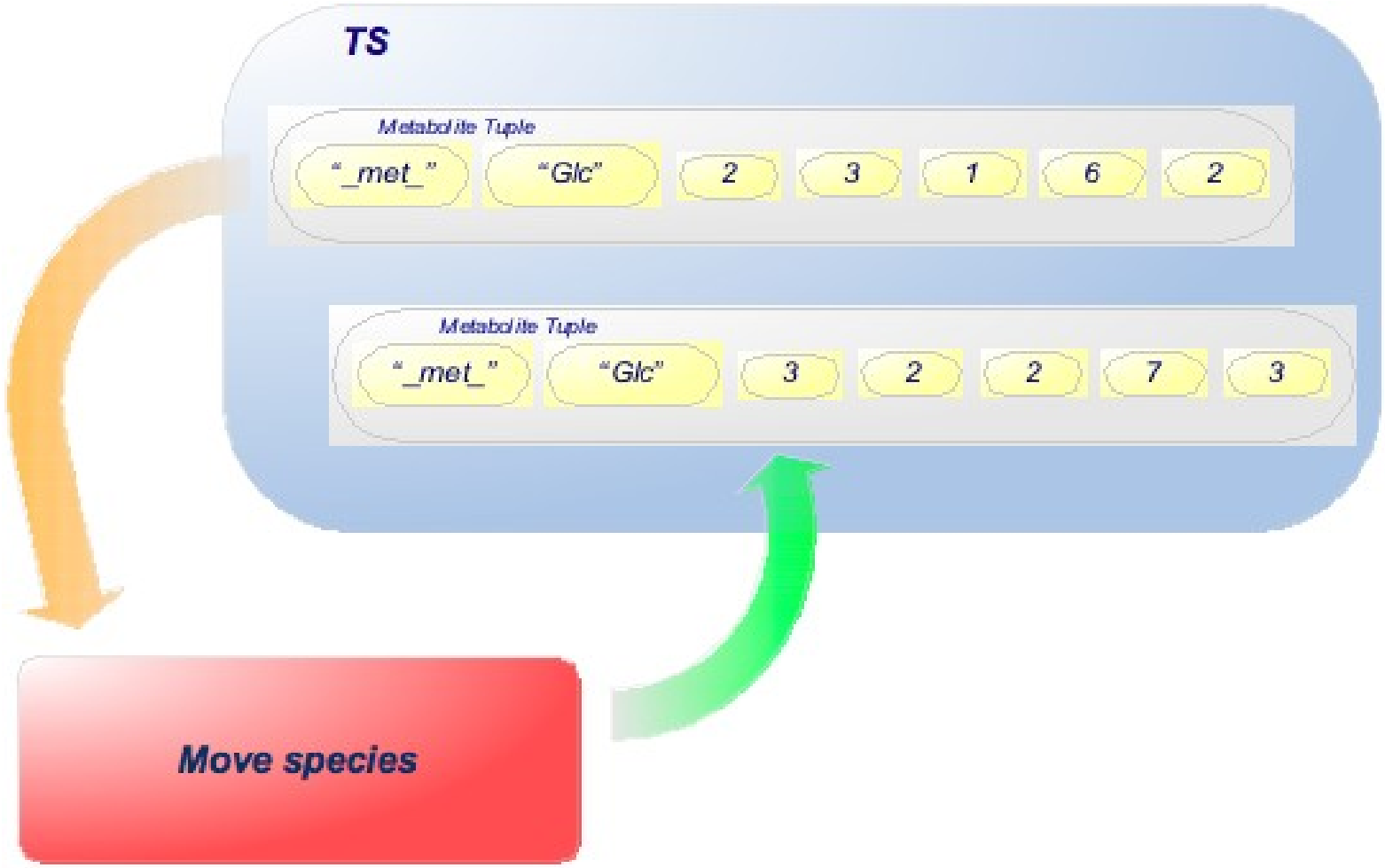
Kalliope has been developed in X-Klaim and run over Klava framework

- a Klaim Tuple Space (TS) represents the Cytoplasm
- the Cytoplasm molecules are tuples in the TS
- a set of Klaim agents manipulated the TS to simulate the Cytoplasm reaction during the Glycolysis

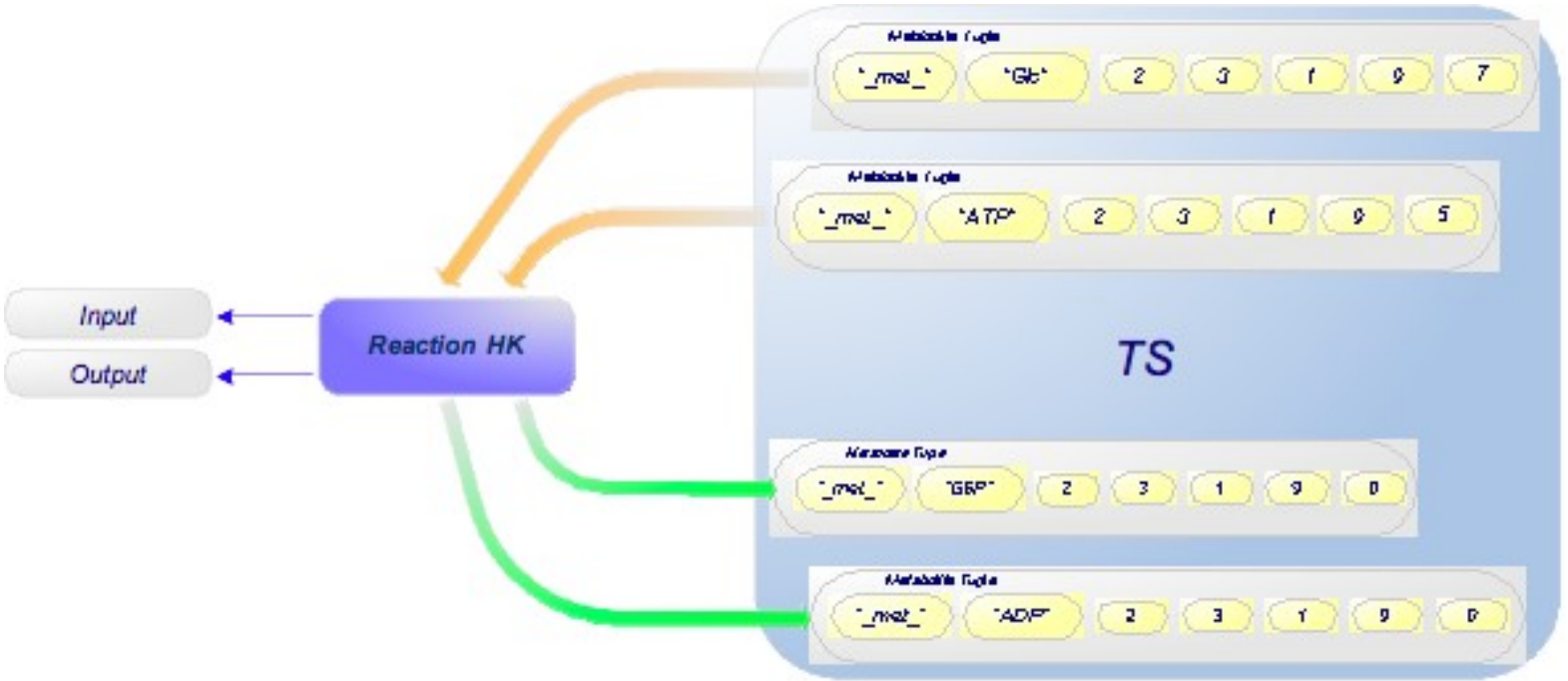
Kalliope-Klaim tuples



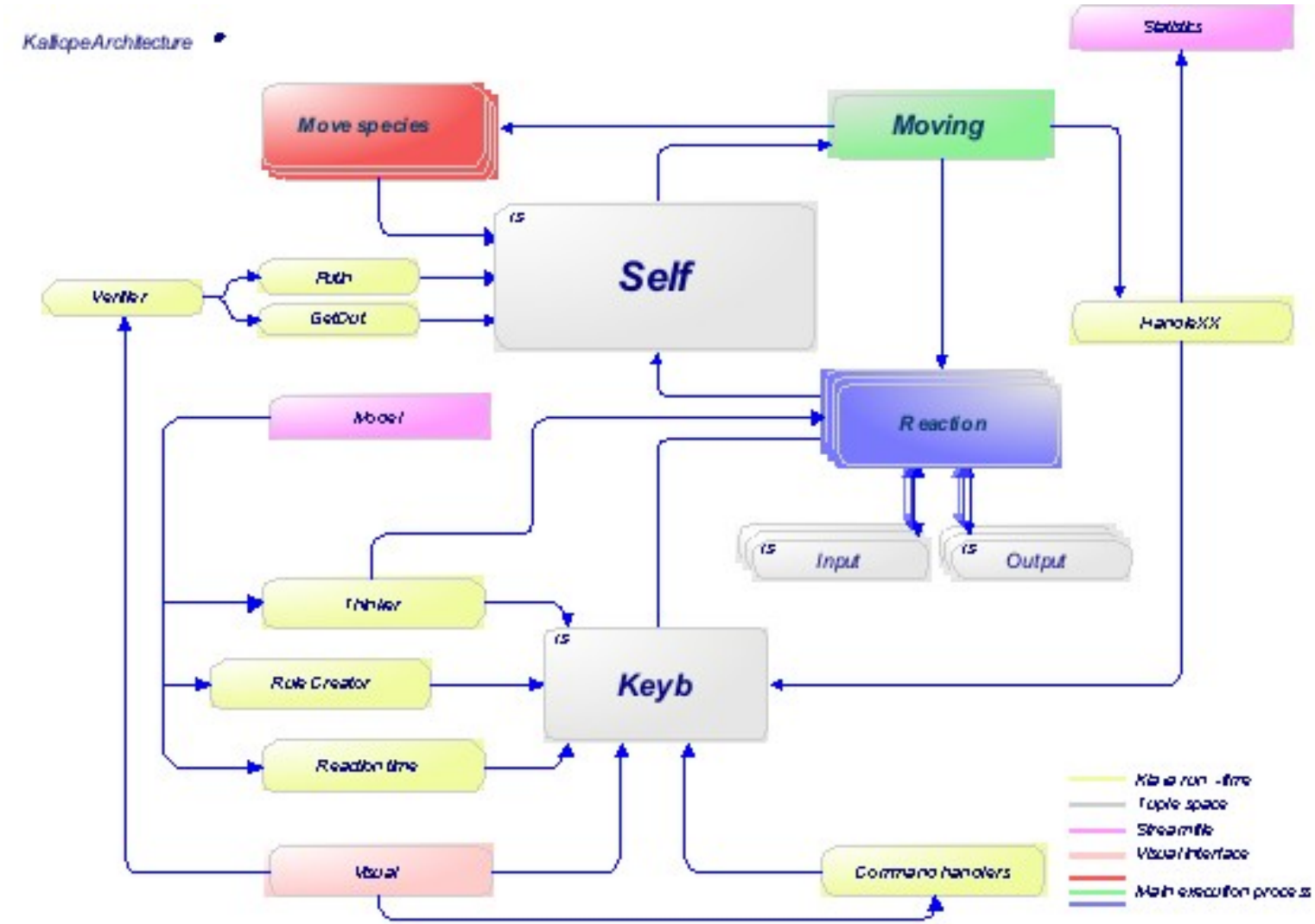
Kalliope movement



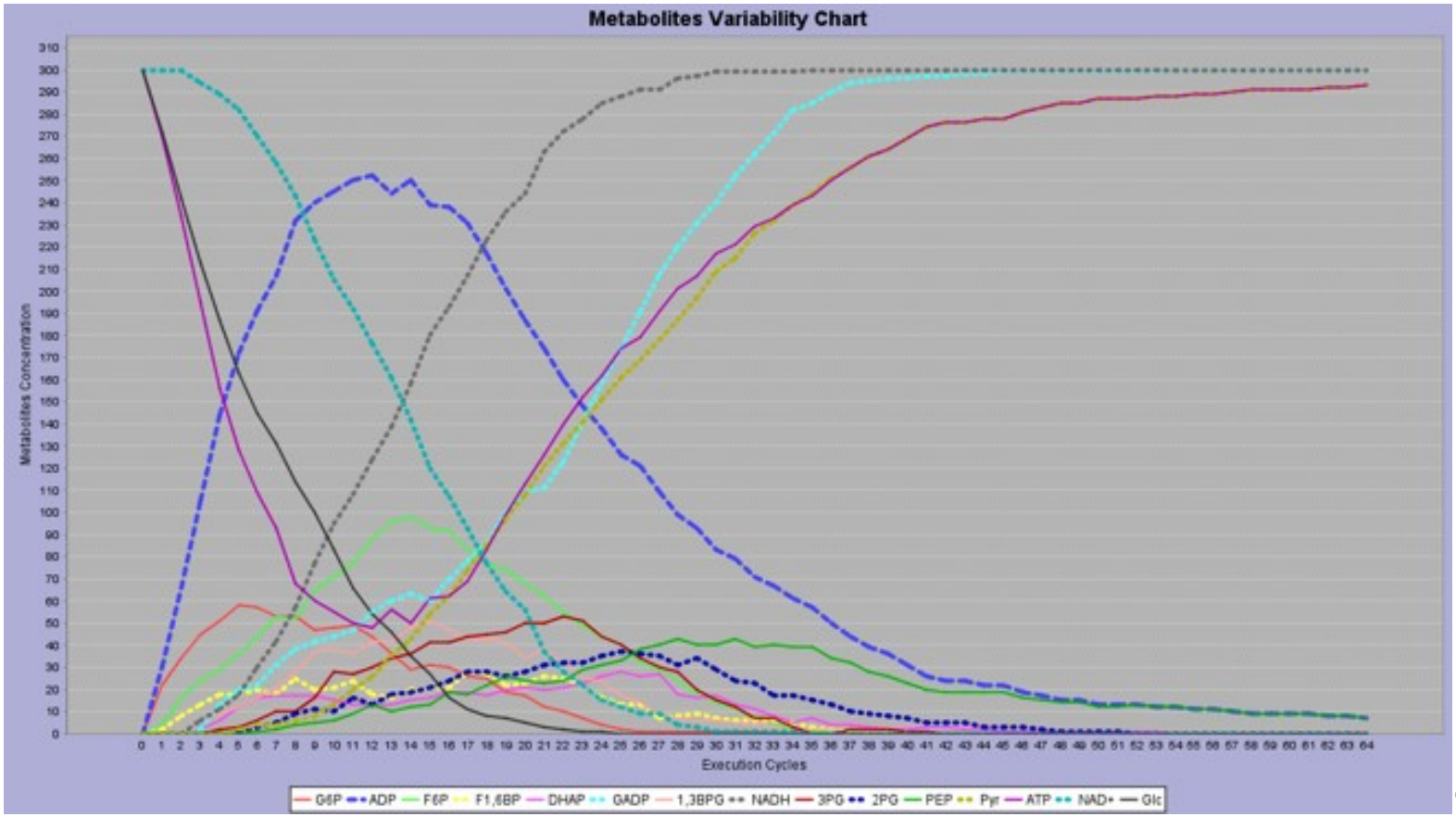
Kalliope reactions



Kalliope simualtor architecture



Kalliope simulation results



Contact us for the Kalliope sw: emanuela.merelli, nicola.cannata, luca.tesei@unicam.it

Coordinator
Flavio Corradini

Researchers
Diletta Cacciagrano
Nicola Cannata
Rosario Culmone
Maria Rita Di Berardini
Emanuela Merelli
Andrea Polini
Alberto Polzonetti
Luca Tesei

Phd Students
Ezio Bartocci
Francesco De Angelis
Francesca Piersigilli
Barbara Re
Oliviero Riganelli
Leonardo Vito
Roberta Alfieri
Federica Chiappori

Cosy Reserach Group