

WP1 @ M17

Progresses and Plan

Mirko Viroli
`mirko.viroli@unibo.it`

ALMA MATER STUDIORUM—Università di Bologna

Frankfurt, 27/2/2012



- 1 General status of WP1
- 2 Discussing reviewers' concerns
- 3 On the new set of eco-laws
- 4 Alchemist simulator
- 5 Plan for Y2 and deliverables



Outline

- 1 General status of WP1
- 2 Discussing reviewers' concerns
- 3 On the new set of eco-laws
- 4 Alchemist simulator
- 5 Plan for Y2 and deliverables



WP1 at a glance – from the DoW

Task T1.1 (M1-M24) – Component and interaction model

- Computational model for eco-laws and interaction of components
- A blueprint for design of infrastructure

Task T1.2 (M1-M24) – Semantic Representation

- LSA model, and connection with work on semantic reasoning
- Implication to simulation and verification of properties

Task T1.3 (M13-M36) – Methodology

- M: Complete methodology for SAPERE (guidelines, manuals, tools)
- A: Simulation and analysis tools for “what if” analysis



WP1 status (work from M13 to M16)

Task T1.1 (M1-M24, 8.5 PM left) – Component and interaction model

- Development of Alchemist (cross-cutting T1.1, T1.2 and T1.3)
- Figuring out a possible minimal set of eco-laws, and implications

Task T1.2 (M1-M24, 14 PM left) – Semantic Representation

- Developing a prototype LSA-space with built-in semantic representation

Task T1.3 (M13-M36, 24 PM left) – Methodology

- Analysis: Alchemist + Early search of related formal models
- Methodology: Andrea will present initial ideas



WP1 papers in Y2

- Paper on Alchemist submitted at JOS
- Paper on WP1 model accepted at SAC 2012
- Paper on Web-tech accepted at an IGI-Global book on WS



Outline

- 1 General status of WP1
- 2 Discussing reviewers' concerns**
- 3 On the new set of eco-laws
- 4 Alchemist simulator
- 5 Plan for Y2 and deliverables



On reviewers' comments

Issue 1: Prediction/control of complexity

- What happens when a system is left evolving?
- How to predict/control complex behaviours to avoid undesired effects
- Investing effort in modelling ecological behaviours (extinction..)
- Investing effort in prediction (analytical and by simulation)

Issue 2: Usability of programming model

- How high is programming / testing effort?
- Is testing in a closed system possible?
- Impact of usability to wide-spread deployment of SAPERE concepts

Some answers/1

Issue 1: Prediction/control of complexity

- What happens when a system is left evolving?
- How to predict/control complex behaviours to avoid undesired effects
- Investing effort in modelling ecological behaviours (extinction..)
- Investing effort in prediction (analytical and by simulation)

Answers, i.e., works to do

- Strengthening our work on simulation frameworks
- Simulating patterns of ecosystems
- Try to link the model with others that are “tractable”
 - Do not expect much: do we have the expertise?
- Should SAPERE change what predict/control/test means?
- Rely on known eco/chemical patterns to avoid some bad behaviour
- Reason about what the new “eco-laws” set makes us gain

Some answers/2

Issue 2: Usability of programming model

- How high is programming / testing effort?
- Is testing in a closed system possible?
- Impact of usability to wide-spread deployment of SAPERE concepts

Answers, works to do, and questions..

- Should be answered in collaboration with WP5 and WP4?
 - Partially captured by fixing the “eco-laws set”
 - Let's provide template agents & LSAs for useful ecosystems
- What is the deployment model of SAPERE?
- Usability very connected to T1.3 (fully unveiled only at M36)
 - Should prepare to show something at M24, defer the rest at M36



On formal predictability/controllability

Can we devise formal techniques?

- There is a whole FP7 Call – open now – just on that (FOCAS)
- SAPERE is a large-scale, knowledge-intensive, open, stochastic system—a nightmare for people doing formal verification!
- We need some rough abstraction, but which?
- The research community is not ready for this, so we can't be..

Any plan? (Other than using simulation..)

- Stick to a small set of SOS patterns, or chemical systems
- Provide semi-formal proofs of general properties like non-divergence
 - Rely on continuous abstractions [Viroli et.al., TAAS2011]
 - From chemical rules to ODEs, then solving the ODE.
- Invest effort on T2.3 – Self-management

Outline

- 1 General status of WP1
- 2 Discussing reviewers' concerns
- 3 On the new set of eco-laws
- 4 Alchemist simulator
- 5 Plan for Y2 and deliverables



Is WP1 still valid?

Main facts that remain valid

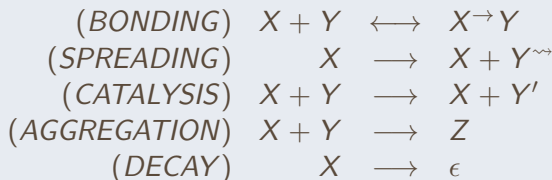
- Agents inject & modify own LSAs, observe only via bonding
- Bonds as LSA references used as values, working locally
- LSAs admit an easy mapping to RDF
- LSA situation reified as properties (location, time,...)
- Eco-laws are chemical-like and admit a mapping to SPARQL

What can/should be changed

- Eco-law language changes role, but it is there
- LSAs as trees, not as tuples (facilitates LSA relocation)
- Bonds used also as communication streams
- Should eventually pick one matching function

The eco-laws set

Chemical patterns



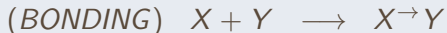
Even if we stick to those, there is still a rather large design space

- How do we match X and Y on the left?
- How do we impose a rate?
- How do we create Z or Y' on the right?
- How do we make an LSA apply to one/more laws?



Bonding eco-law (symmetric version needed)

Chemical pattern



How do we match X and Y on the left?

- Whole X and Y match
- X declares how Y should be
- X declares which subpart of it should match any part of Y
- X declares which subpart of it should match which some part of Y

How do we create the bond?

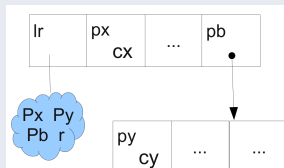
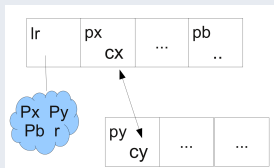
- X declares a property that will hold the bond
- Each LSA has one given property that holds bonds
- Some bond can be unique, multiple, uni/bidirectional

One example of (BONDING)

Expressed in Y1 eco-law language

```
?X:[?PX = (?CX);
    :lawrequest has (:type = (:bonding);
                    :rate = (?R);
                    :bond = (?PB);
                    :localprop = (?PX);
                    :targetprop = (?PY) ]);] +
?Y: [?PY = (?{CY: ?CY matches ?CX})]
--?R-->
?X:[ ?PB has (?Y)] + ?Y
```

Pictorially



One example of (BONDING)

Expressed in Y1 eco-law language

```
?X:[?PX = (?CX);
    :lawrequest has (:type = (:bonding);
                    :rate = (?R);
                    :bond = (?PB);
                    :localprop = (?PX);
                    :targetprop = (?PY) ]);] +
?Y: [?PY = (?{CY: ?CY matches ?CX})]
--?R-->
?X:[ ?PB has (?Y)] + ?Y
```

Features

- LSAs have a :lawrequest property
- X declares the subpart of it (PX) and the subpart of Y (PY)
- The bond is an additional reference into ?PB

Another example of (BONDING)

Expressed in Y1 eco-law language

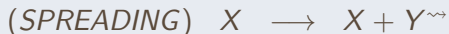
```
?X:[?PX = (?CX);
      :lawrequest has (:type = (:bonding);
                      :rate = (?R);
                      :bond = (?PB);
                      :localprop = (?PX);
                      :bondnameloc = (?PNL) ]);] +
?Y: [?PY = (?{CY: ?CY matches ?CX})]
--?R-->
?X:[ ?PB has (?Y); ?PNL = (?PY)] + ?Y
```

Features

- X declares PX but not PY , PY is automatically found
- the name of PY property goes into $?PNL$, so that X 's agent knows it
- This is to be used if you do not know Y 's LSA
(but if you do, this eco-law is not very useful)

Spreading eco-law

Chemical pattern



How do we select X on the left?

- X declares he wants to be spread

How do we create Y ?

- A clone of X /an empty LSA, plus some change
- The “change” as a $Y.p = f(X.q)$ where p, q, f are declared by X
- f is the name of an external function or an expression

Where do we send Y ?

- In all neighbours (but which topology?)
- X declares where, and it could be a set

One example of (SPREADING)

Expressed in Y1 eco-law language

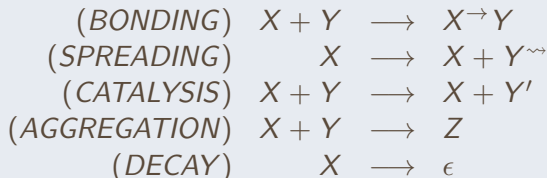
```
?X:[?P = (?C); ?PL = ?CL;
    :lawrequest has (:type = (:spreading);
                    :rate = (?R);
                    :tfunction = (?F);
                    :locationprop = (?PL);
                    :prop = (?P))]]
--?R-->
?X + ?Y:[clones ?X; ?P = (?{C2: C2 is ?F(?C)});
    :location = ?CL; :previous = (?X)]
```

Features

- X and Y have only different P (is that enough?)
- This way, Y spreads itself, indefinitely
(We can handle this by assuming functions can be partial)
- Still cannot easily make Y depend on distance, or topology

The eco-laws set again

Chemical patterns



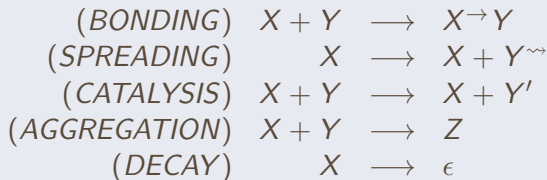
We considered BONDING and SPREADING, others are similar

- CATALYSIS: X and Y should match, Y' is a change of Y
- AGGREGATION: X and Y should match, Z is a new one obtained e.g. like $Y^{\rightsquigarrow}$ in SPREADING
- DECAY: like SPREADING, though we remove X



A more refined design space

Chemical patterns



Ultimately we need

- ① A matching technique for X and Y on left
- ② A change technique for updating/creating LSAs
- ③ A selection technique for X alone on left



1 - Matching

A naive implementation of matches

One that all WPs can assume to use by M20

- The two values/descriptions should be identical
- An element `sapere:any` on one side matches anything on the other side

WP1 stance

- We cannot throw away “semantics”, can we?
- WP1 shall study other matches by M20 or M24 as key goal of T1.2
- An agenda:
 - WP3, WP5 will tell us what is needed by M20 (strict!)
 - WP1 will specify them by M24
 - WP4 will possibly implement later

2 - Changes

What do those update functions do?

- They take one value or aggregate and produce another
- In some places we might constraint them (ODI,..)
- They can be partial

Some (mostly alternative) approaches

- Just name some that WP4 implement by M20
- Invent a mini-language to express them
- Rely on SPARQL

WP1 stance

- Shortly decide for some functions in WP4
- In WP1 we shall deepen the issue of expressiveness

3 - Selection

This is easier

- An LSA declares if it should be subject to (SPREADING,DECAY)
- The use of partial transformation functions can be a further selection mechanism based on X 's content
- Can select by using predicates operators working on the left



A candidate set

Design choices

- An LSA carries a structured “eco-law request”, with all the necessary information
 - Eco-law name
 - Application rate (possibly ASAP)
 - Which part of the LSA is used to match the other reactant
 - Effects on the LSA
- Match is orthogonal: ok to momentarily fix a syntactic one
- Effects are functional operators on an LSA part

A candidate set so far could be provided

- No clue this is as expressive as needed
- The right way to go, to me, is to work on it for 1-2 weeks

Eco-laws 1/3

(BONDING)

```
?X:[?PX = (?CX); :lawrequest has (:type = (:bonding);
                                :rate = (?R);
                                :bond = (?PB);
                                :localprop = (?PX);
                                :targetprop = (?PY) ]);] +
?Y: [?PY = (?{CY: ?CY matches ?CX})]
--?R-->
?X:[ ?PB has (?Y)] + ?Y
```

(UNBONDING)

```
?X:[?PX = (?CX); :lawrequest has (:type = (:bonding);
                                :rate = (?R);
                                :bond = (?PB);
                                :localprop = (?PX);
                                :targetprop = (?PY) ]);] +
?Y: [?PY = (?{CY: not(?CY matches ?CX)})]
--?R-->
?X:[ ?PB has-not (?Y)] + ?Y
```

Eco-laws 2/3

(SPREADING)

```
?X:[?P = (?C); ?PL = ?CL; :lawrequest has ([:type = (:spreading);
:rate = (?R);
:tfuction = (?F);
:locationprop = (?PL);
:prop = (?P)])]
--?R-->
?X + ?Y:[clones ?X; ?P = (?{C2: C2 is ?F(?C)});
:location = ?CL; :previous = (?X)]
```

(DECAY)

```
?X:[?P = (?{C:?Pred(?C)}); ?PL = ?CL;
:lawrequest has ([:type = (:decay);
:rate = (?R);
:tpredicate = (?Pred);
:prop = (?P)])]
--?R--> 0
```

Eco-laws 3/3

(CATALYSIS)

```
?X:[?PX = (?CX); ?PT = (?CT); :lawrequest has ([:type = (:catalysis);
                                                    :rate = (?R);
                                                    :tfunction = (?F);
                                                    :tprop = (?PT);
                                                    :localprop = (?PX);
                                                    :targetprop = (?PY) ]);]+

?Y: [?PY = (?{CY: ?CY matches ?CX}) ]
--?R-->
?X: [?PT = (?{CT2: CT2 = ?F(?CT)})] + ?Y
```

(AGGREGATION)

```
?X:[?P = (?CX); :lawrequest has ([:type = (:catalysis);
                                    :rate = (?R);
                                    :spredicate = (?Pred);
                                    :tfunction = (?F);
                                    :aggrprop = (?P)]);] +

?Y: [?P = (?{CY: ?Pred(?CX,?CY)}) ]
--?R-->
?Z: [clones ?X; ?P = (?{CZ: CZ = ?F(?CX,?CY)})]
```

Outline

- 1 General status of WP1
- 2 Discussing reviewers' concerns
- 3 On the new set of eco-laws
- 4 Alchemist simulator**
- 5 Plan for Y2 and deliverables



On the Alchemist simulation tool

Few facts.. Danilo will explain more

- A sub-task started in Y1
- Pros: Performance, small abstraction gap
- Cons: Reporting/graphics worst than more famous tools
- We will keep developing it, and will produce manuals



Outline

- 1 General status of WP1
- 2 Discussing reviewers' concerns
- 3 On the new set of eco-laws
- 4 Alchemist simulator
- 5 Plan for Y2 and deliverables



Activities during Y2

WP1 agenda

- T1.1 Fully formalise the eco-laws set, and then clean the model
- T1.1 Bootstrap formal verification of properties
- T1.2 Study matching with good expressiveness/performance
- T1.2 Find a proper place for ontologies
- T1.2 Complete semantic LSA-space (also to WP4)
- T1.3 Complete the core of Alchemist
- T1.3 Bootstrap methodology



Deliverables

D1.2@M20 – Complete operational model (14PM:7B+3G+3M+1S)

- Completion of operational model
- Extended live semantic annotation framework

D1.3@M24 – final oper. and semantic model (8PM:3B+1G+1M+3S)

- Final revision of computational model
- Final revision of live semantic annotation framework

D1.4@M24 – Early report on methodology (8PM:4B+2G+2M)

- Early analysis of methodological guidelines
- Early analysis of solutions for analysis tools



Rough content of deliverables

D1.2@M20 – Complete operational model (14PM:7B+3G+3M+1S)

- First version of the cleaned model
- First results on verification
- First experimentation with semantic matches and ontologies

D1.3@M24 – Final oper. and semantic model (8PM:3B+1G+1M+3S)

- Final versions of the above after extensive feedbacks from WP*
- (Verification will move to T1.3 in Y3)

D1.4@M24 – Early report on methodology (8PM:4B+2G+2M)

- Early analysis of methodological guidelines (see Andrea's talk)
- Alchemist core release

Some critical points

Warnings for WP1 in Y2

- Effectiveness of the eco-law set, and its impact on the model
- Effectiveness/efficiency of semantic matching
- T1.2 still requires much effort
- Formal analysis and simulation seem to crosscut T1.2/T1.3
- Satisfactorily tackling science issues already at Y2?
- The model is still not challenged as the proposal envisioned..

TODO related to WP1 for other units

- STA: Keeping the semantic side of SAPERE live..
- UNIMORE: Building libraries that make the model useful..
- UNIGE: Predicting emergence..
- JKU: Challenging the ecological side of SAPERE..