MICROSOFT Research Cambridge
R2D2 Workshop - May 12-13 2008

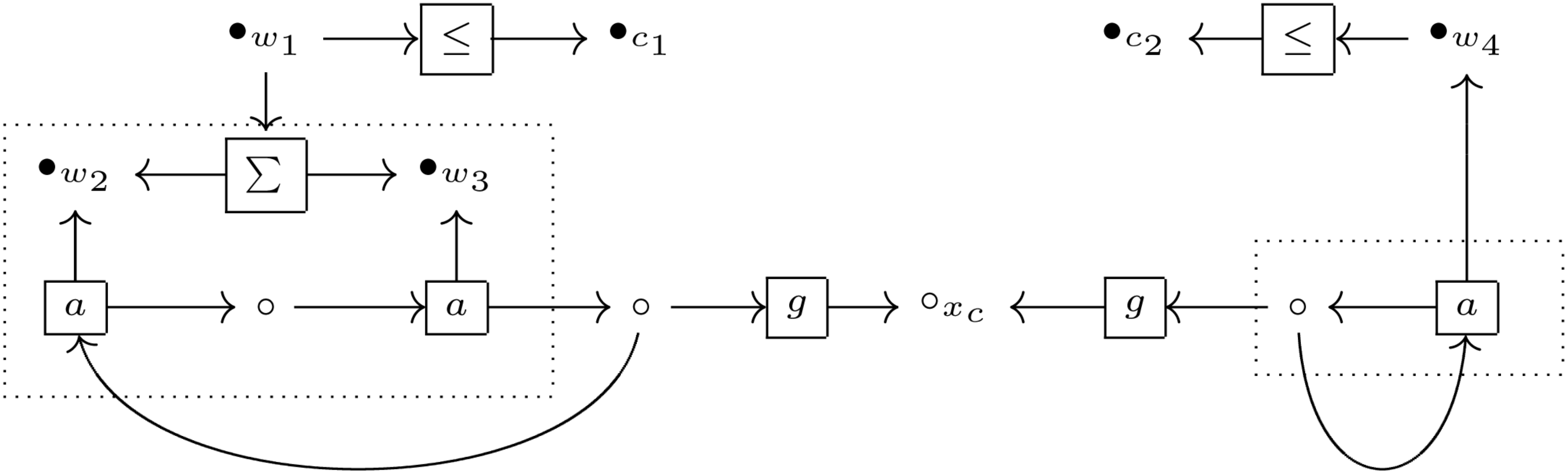
Architectural Design Rewriting as an Architecture Description Language

Ugo Montanari
Università di Pisa

joint work with
Roberto Bruni, Alberto Lluch Lafuente, Univ. Pisa
and Emilio Tuosto, Univ. of Leicester

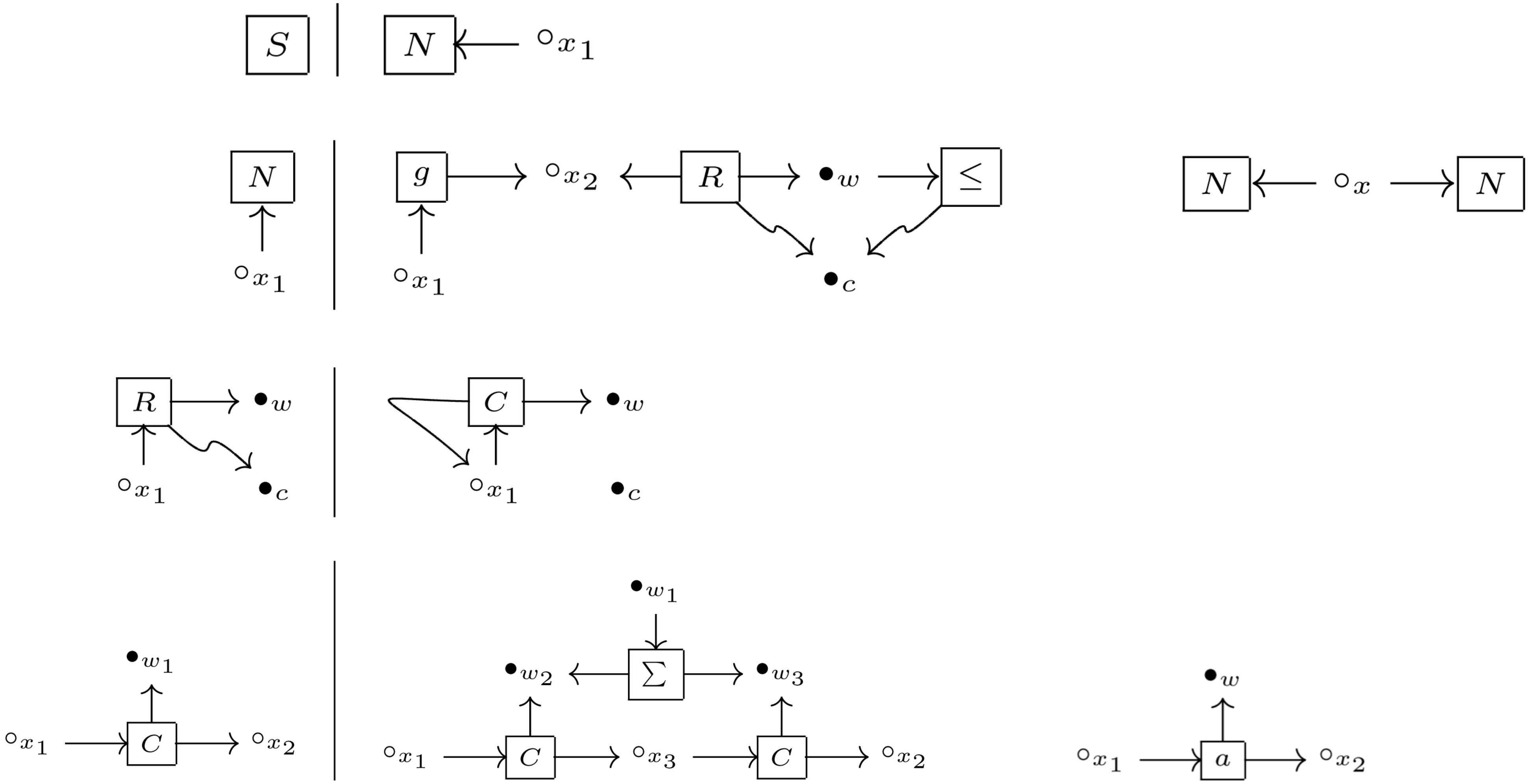
ADR, Architectural Design Rewriting: An Example

Rings of Agents with Gateways

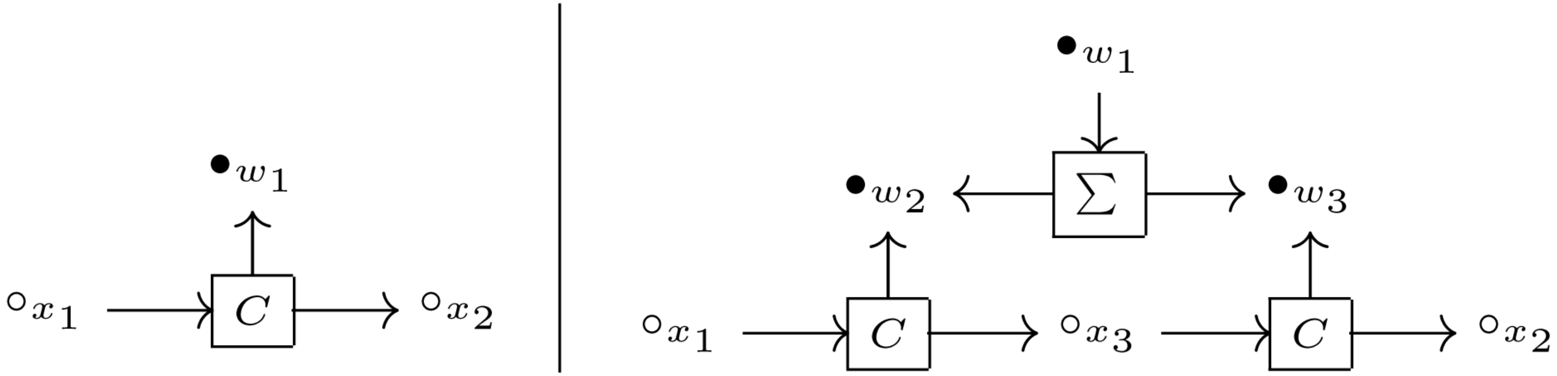


The combined load of all the agents of a ring
should not exceed a given threshold

Productions



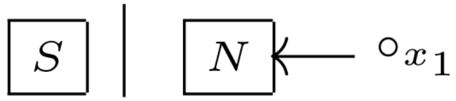
Productions Are Operations



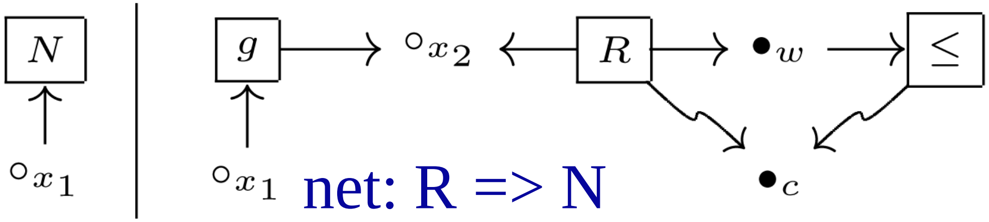
chains: C x C => C

- Design = graph with interface
- Operation chains maps two C-designs to a C-design
  chains(X:C,Y:C):C
- Values of the algebra are software architectures, sorts are nonterminals, carriers are architectural styles
- Ordinary process algebra-like operations on graphs (parallel composition, restriction) are easily represented

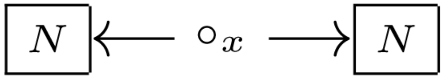
An Algebra of Design Operations



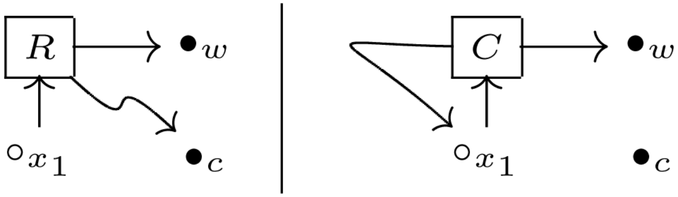
system: N => S



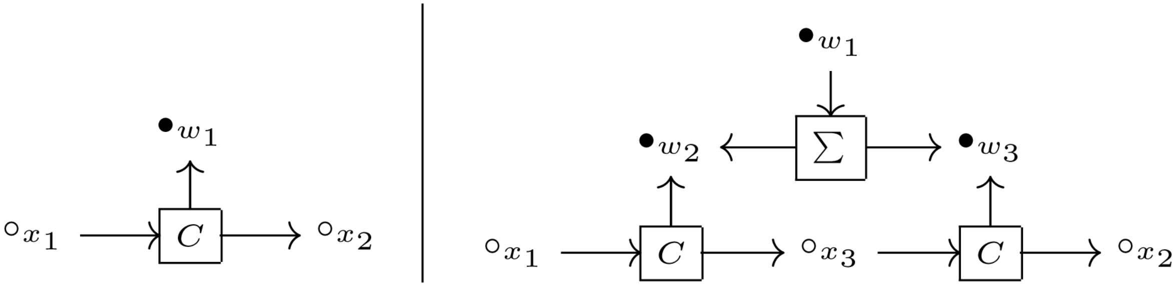
net: R => N



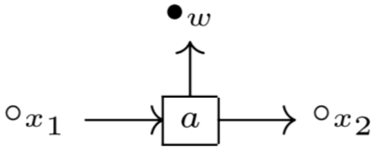
nets: N x N => N



ring: C => R

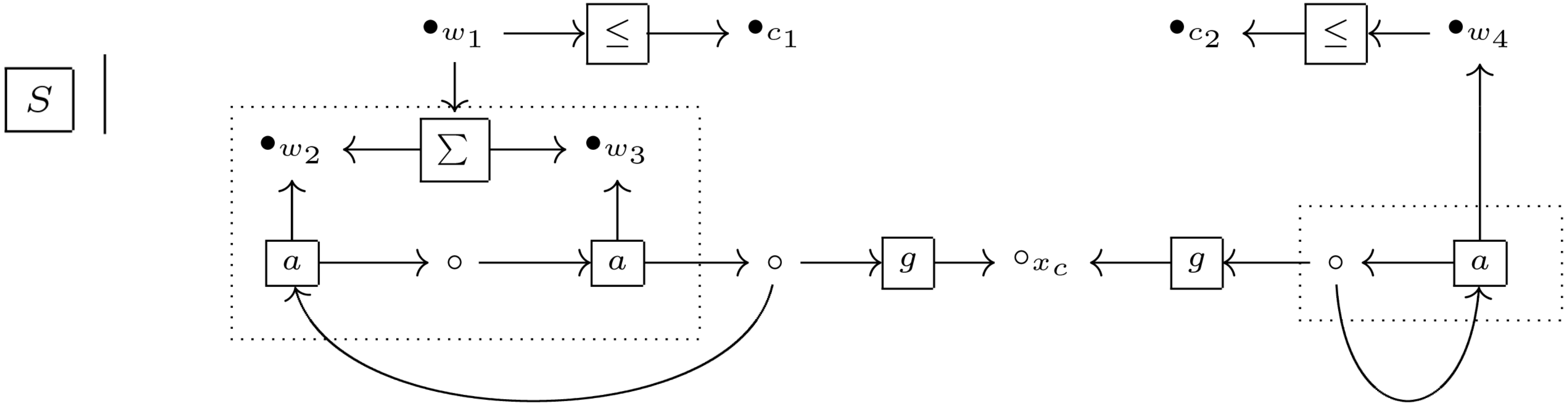


chains: C x C => C



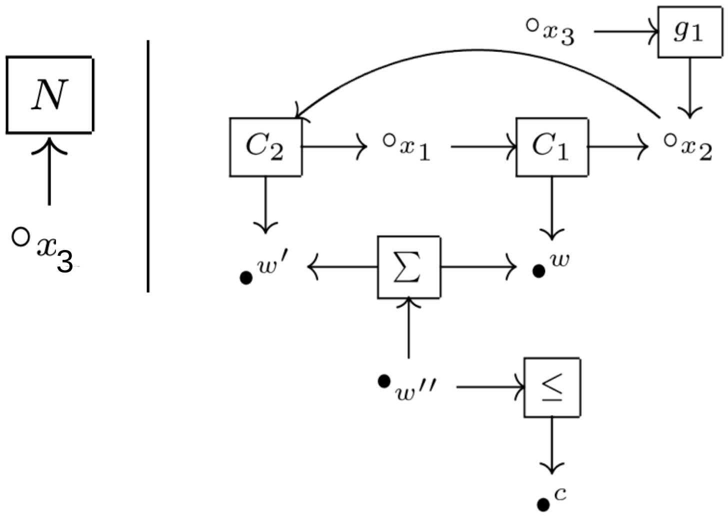
chain: => C

Rings of Agents with Gateways



system(nets(net(ring(chains(chain,chain)),net(ring(chain)))): S

Partial Designs



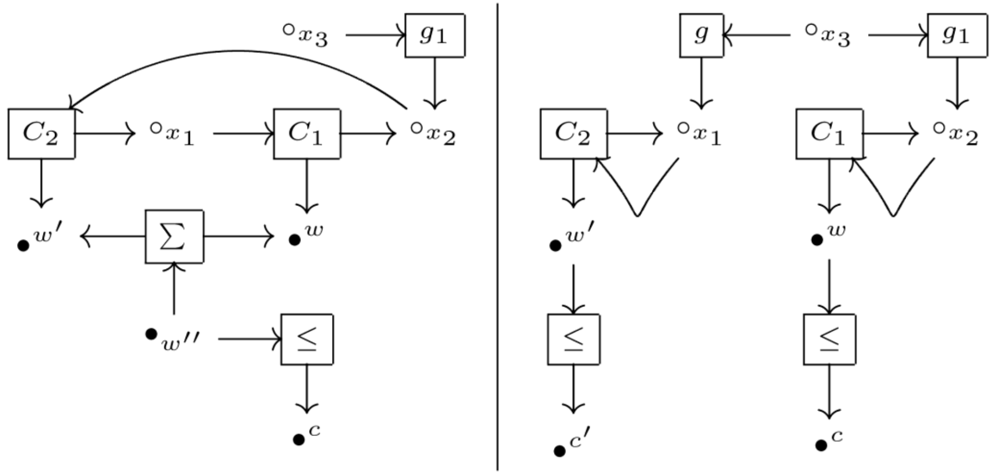
net(ring(chains(X:C,Y:C))): N

=

net(ring(Z:C))[chains(X:C,Y:C):C/Z]

- Terms with variables represent partial designs
- Substitution means refinement; the inverse abstraction
- Terms have more information than their evaluations: they are proofs of type

Term Rewriting



net(ring(chains(X:C,Y:C))):N

=>

nets(ring(X:C),ring(Y:C)):N

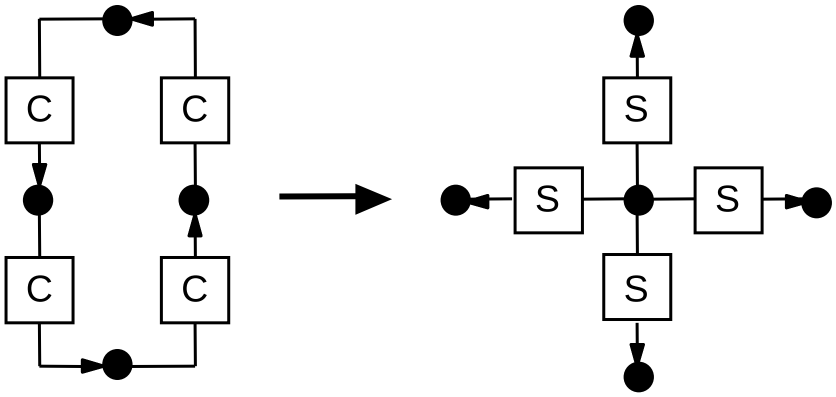
- Rewriting rules L => R con be instantiated and contextualized
- Rewritings are concurrent and guaranteed to preserve typing
- Rewritings can be triggered by the constraint structure:

e.g. here w''≥c/2 and w≥c/6 and w'≥c/6, when implied by the present set of constraints, could trigger a duplication of the ring

Conditional Rewriting SOS style

- Conditional rules can be of the form:
- X: A =a=> Y:B implies L(X):C =b=> R(Y):D
- Complex transitions can be constructed which guarantee synchronous updatings, e.g. nested wrappings for QoS
- Types need not be preserved, but consistent type modifications can be proved
- Process algebra-like semantics with synchronization, extrusion, etc. can be modeled

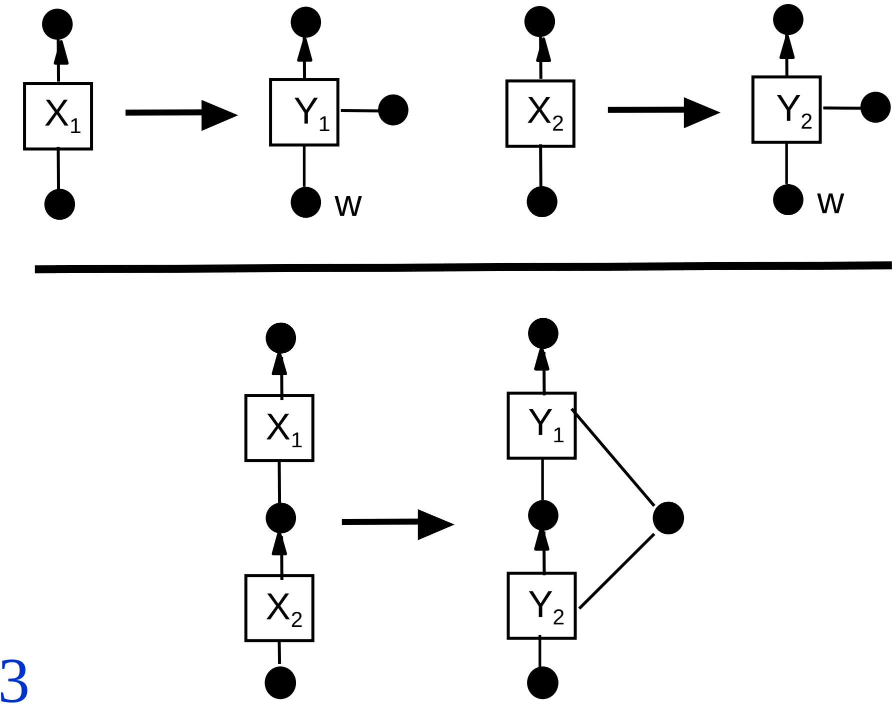
A Rewriting

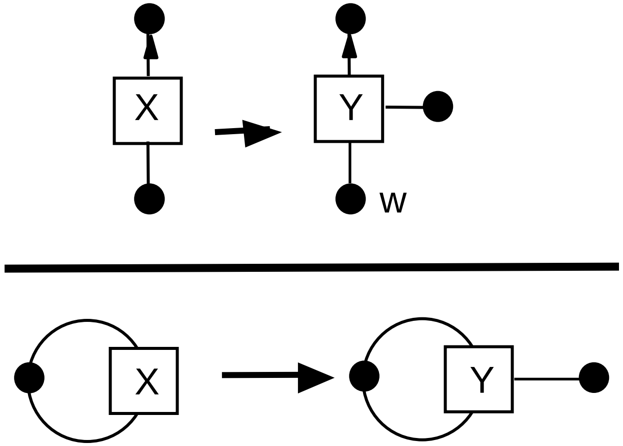


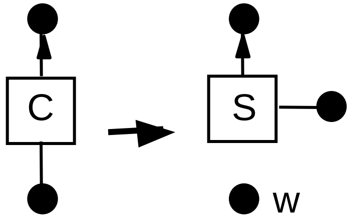
ring(chain(chain(C,C),chain(C,C))):1 =>

star(join(join(ray(S),ray(S)),join(ray(S),ray(S)))):1

Conditional Rewriting Rules







C:2 => ray(S:2):3

$$\frac{X:2 => Y:3 \qquad .}{ring(X):1 => star(Y):1}$$

$$\frac{X_1:2 => Y_1:3 \qquad X_2:2 => Y_2:3\ .}{chain(X_1,X_2):2 => join(Y_1,Y_2):3}$$

Conclusion

- ADR models design, execution and reconfiguration phases
- Process calculi tailored to software architecture
- Presentation in ADR style of:
  - SRML, Sensoria Reference Modeling Language
  - SHR, Synchronized Hyperedge Replacement
  - UML
  - REO, by Farhad Arbab et al., CWI
- SENSORIA case studies about web services
- Implementation in MAUDE
  - Graphical structure
  - Basic operations
  - Reconfiguation as rewriting
- ADR site: http://www.albertolluch.com/index.html?x=adr.html