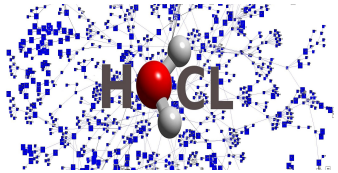
Decentralized Workflow Execution Through Molecular Composition

7th International Workshop on Engineering Service-Oriented Applications

ICSOC. Paphos, December 5, 2011

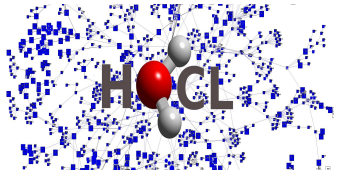
Héctor Fernandez, Cédric Tedeschi and Thierry Priol

Autonomic Service Computing?



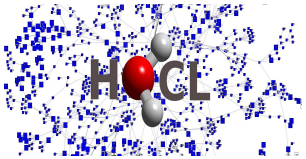
- The *Service* abstraction
  - Autonomous
  - Network-enabled
- Service-Oriented applications
  - Built as temporal compositions of services (workflows)
  - Loosely-Coupled, dynamic composition
- Need for autonomic execution
  - Self-adaptation, self-healing, self-organization...
  - → **Decentralized Workflow execution**
  - → **Self-coordination**

Service Coordination: background



- Workflow engines
  - One coordinator
  - Managing all data and control flows
  - Scalability?
  - Reliability?
- Workflow execution languages
  - Data / Control driven
  - Implicit / Explicit parallelism
  - BPEL, XPDL, SCUFL, MoML, DAX
  - Designed for centralized, static coordination

Towards autonomous coordination

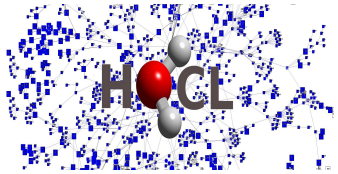


- High-level abstractions...
  - Self-adaptation
  - Self-healing
  - Self-...
  - **Self-coordination**
- ...To help designing autonomic service-oriented application

"Nature-inspired metaphors have been shown to be of high interest for service coordination." [Viroli *et al.*, 2009]

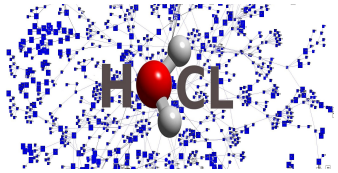
- **→ Workflow execution as a succession of chemical reaction**

Chemical Programming Model



- An execution model
  - A program's runtime envisioned as a succession of chemical reactions
  - Reactions taking place (implicitly)
    - In parallel, autonomously, non-deterministically
    - Until no more reactions can take place (state of *inertia*)
- Writing a chemical program
  - Data: define the solution as a multiset
  - Program: define a set of rules rewriting the multiset
- Languages
  - GAMMA [Banâtre *et al*., 1990]
  - HOCL (Higher-Order Chemical Languages) [Radenac, 2007]
    - Rules can apply on rules → program is able to change dynamically

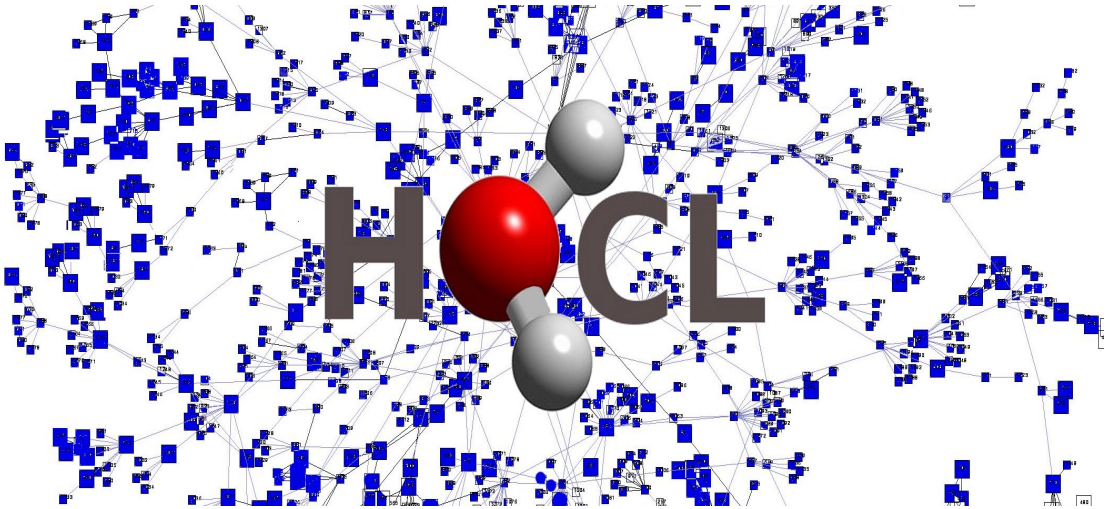
A Basic Chemical Program



- A specification
- A possible execution (amongst others)

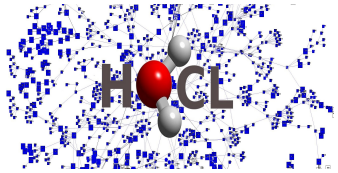
replace x, y **by** x **if** $x \geq y$ **in** $\langle 2, 3, 5, 8, 9 \rangle$

$$\langle 2, 3, 5, 8, 9 \rangle \rightarrow^* \langle 3, 5, 9 \rangle \rightarrow^* \langle 9 \rangle$$

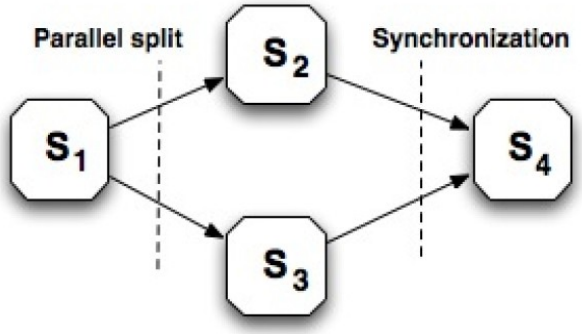


Workflow as a molecular composition

Defining a *molecular* workflow

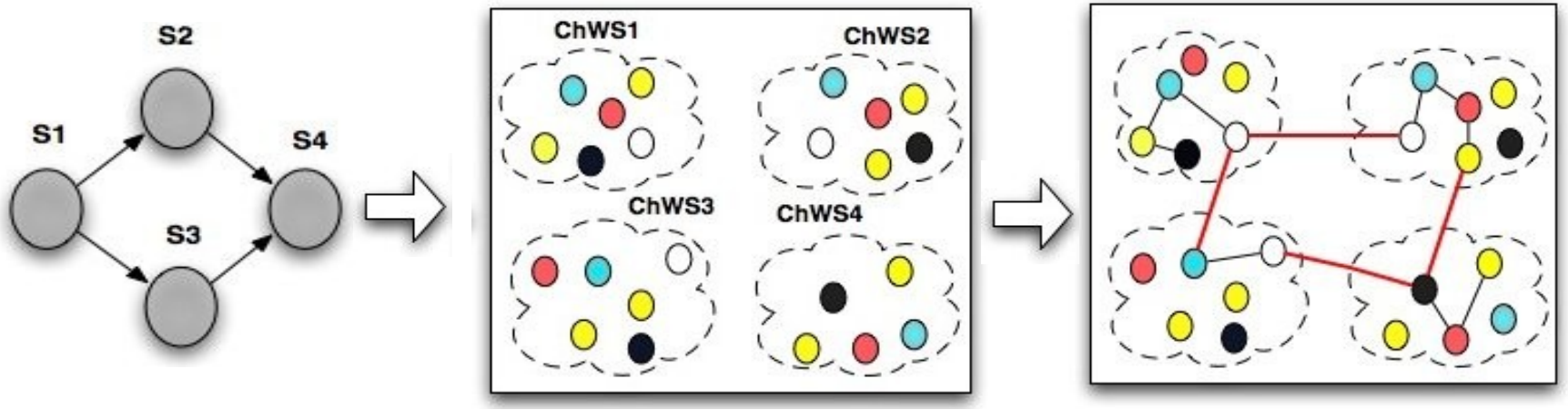


- Express all data and control flow (reaction rules and molecules).



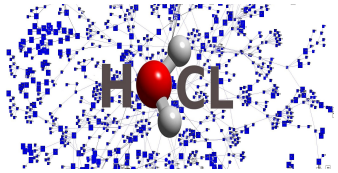
⟨ // Multiset (Solution)
WS1:⟨CALL:S1, PARAM:⟨in1⟩,DEST:WS2, DEST:WS3⟩, // WS1 Sub-solution
WS2:⟨DEST:WS4, **replace** RESULT:WS1:value1 **by** CALL:S2, PARAM:⟨(value1)⟩ ⟩,
WS3:⟨DEST:WS4, **replace** RESULT:WS1:value1 **by** CALL:S3, PARAM:⟨(value1)⟩ ⟩,
WS4:⟨**replace** RESULT:WS2:value2, RESULT:WS3:value3 **by** CALL:S4, PARAM:⟨(value2)⟩
⟩

Molecular workflow: design & runtime



- At runtime
  - Interactions between services modeled by reactions
  - Molecules react in chain following the patterns of the workflow
- At design time
  - Workflow defined using specific reaction rules and molecules (**multiset**)
  - Each Service is a sub-solution with specific molecules
  - Each Service is provided with a library of generic rules

Chemical rules for distributed execution



- Independent from any workflow
- Chemical rules to ensure the execution

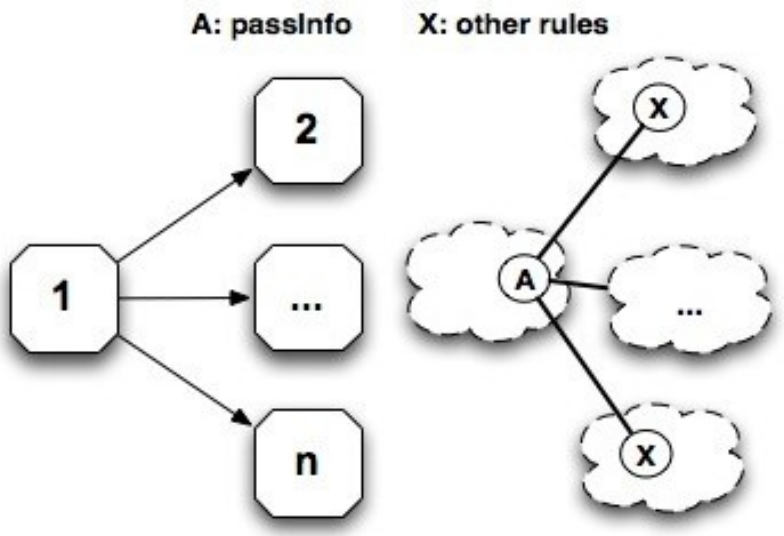
$$\textbf{let } \mathit{invokeServ} = \textbf{replace } \text{ChWSi:}\langle \textsc{Call}\text{:Si, } \textsc{Param}\text{:}\langle \text{in}_1,\ldots,\text{in}_n \rangle, \textsc{Flag\_Invoke}\text{:1, } \omega \rangle,$$
$$\textbf{by } \text{ChWSi:}\langle \textsc{Result}\text{:ChWSi:}\langle \text{value} \rangle , \omega \rangle$$
$$\textbf{let } \mathit{preparePass} = \textbf{replace } \text{ChWSi:}\langle \textsc{Result}\text{:ChWSi:}\langle \text{value} \rangle , \textsc{Dest}\text{:ChWSj, } \omega \rangle$$
$$\textbf{by } \text{ChWSi:}\langle \textsc{Pass}\text{:ChWSj:}\langle \textsc{Completed}\text{:ChWSi:}\langle \text{value} \rangle \rangle \rangle$$

- Control and data transfer among ChWSes.

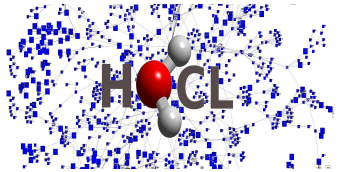
$$\textbf{let } \mathit{passInfo} = \textbf{replace } \text{ChWSj:}\langle \textsc{Pass}\text{:ChWSi:}\langle \omega_1 \rangle , \omega_2 \rangle, \text{ChWSi:}\langle \omega_3 \rangle$$
$$\textbf{by } \text{ChWSj:}\langle \omega_2 \rangle, \text{ChWSi:}\langle \omega_1, \omega_3 \rangle$$

Workflow patterns: Parallel Split

$$\textbf{let } passInfo = \textbf{replace } \text{ChWSj:}\langle \text{Pass:ChWSi:}\langle \omega_1 \rangle, \omega_2 \rangle, \text{ChWSi:}\langle \omega_3 \rangle$$
$$\textbf{by } \text{ChWSj:}\langle \omega_2 \rangle, \text{ChWSi:}\langle \omega_1, \omega_3 \rangle$$

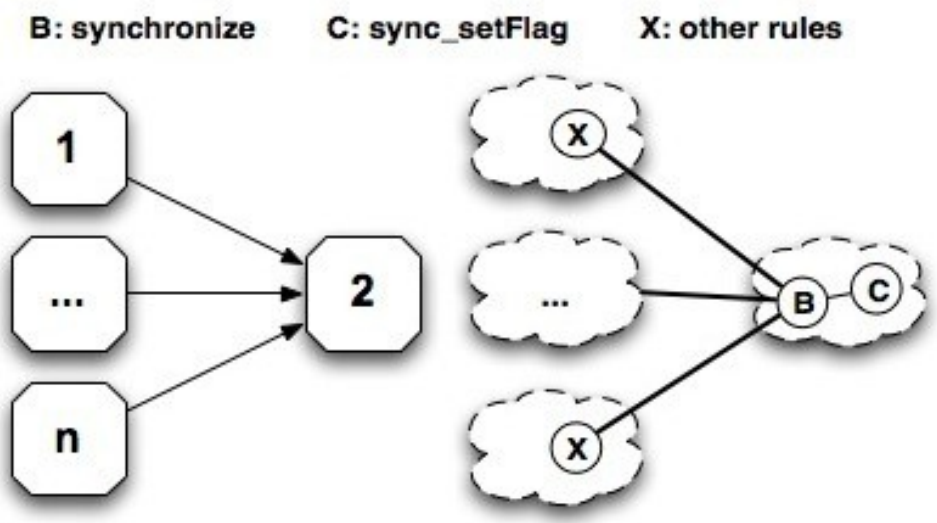


Sorkflow pattern: Synchronization

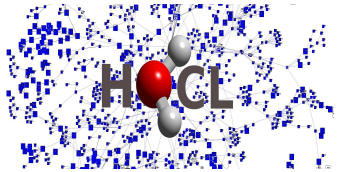


$$\textbf{let } synchronize = \textbf{replace } \textsc{Sync\_Src}{:}\langle \text{ChWSi}, \omega_1 \rangle, \textsc{Completed}{:}\text{ChWSi}{:}\langle \text{value} \rangle, \textsc{Sync\_Inbox}{:}\langle \omega_2 \rangle \textbf{ by } \textsc{Sync\_Inbox}{:}\langle \textsc{Completed}{:}\text{ChWSi}{:}\langle \text{value} \rangle, \omega_2 \rangle, \textsc{Sync\_Src}{:}\langle \omega_1 \rangle$$

$$\textbf{let } sync\_setFlag = \textbf{replace-one } \textsc{Sync\_Src}{:}\langle\,\rangle \textbf{ by } \textsc{Flag\_Invoke}{:}1$$

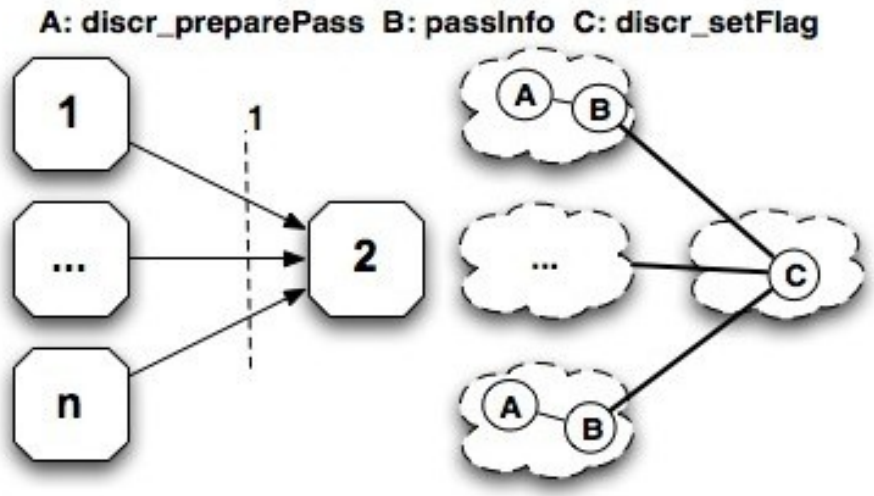


Workflow pattern: Discriminator

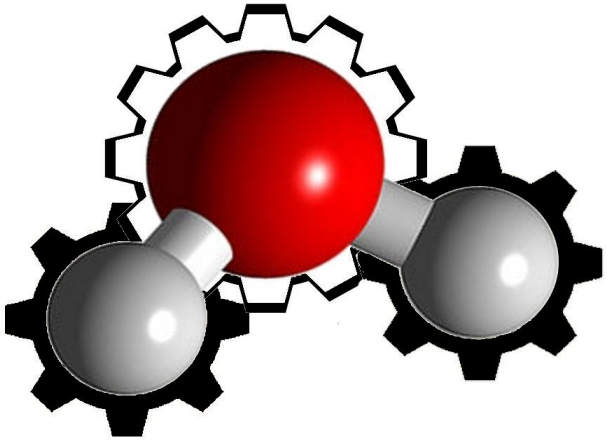


let *discr_preparePass* = **replace** DEST:ChWSj, RESULT:ChWSi:⟨value⟩
by PASS:ChWSj:⟨COMPLETED:ChWSi:⟨value⟩, DISCRIMINATOR:Yes⟩

let *discr_setFlag* = **replace-one** DISCRIMINATOR:Yes **by** FLAG_INVOKE:1



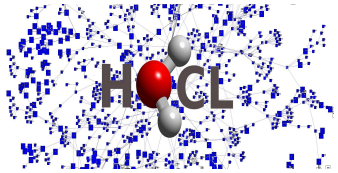
Architectures



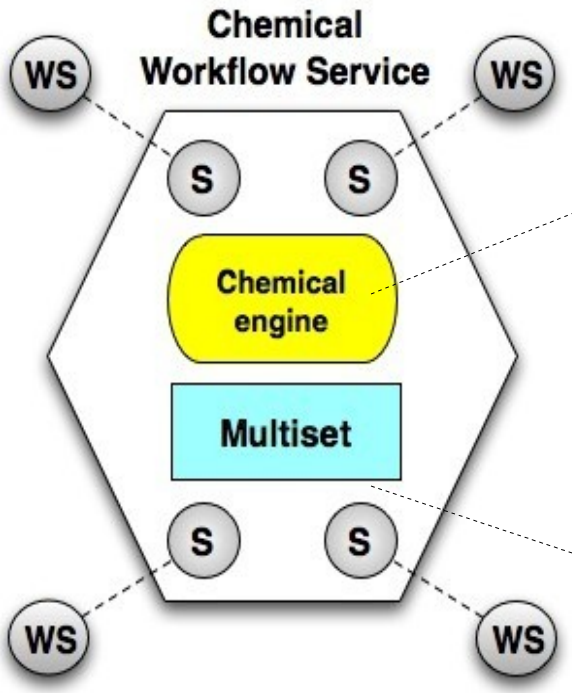
- Coordination mechanism built upon HOCL

- Two architectures, two software prototypes
  - Centralized
  - Decentralized

Centralized Architecture



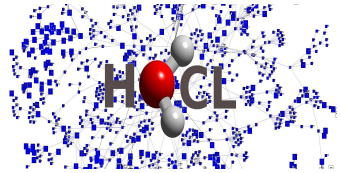
- Central node coordinating all data and control flows
  - A chemical encapsulation per Web service participating in the workflow
  - **Multiset** containing the workflow definition
  - **Chemical engine** processing the content of the multiset

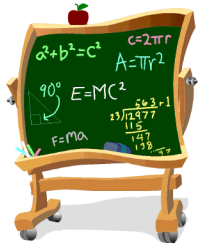


let *invokeServ* = **replace** ChWSi:⟨CALL:Si, PARAM:⟨in$_1$, …, in$_n$⟩, FLAG_INVOKE, ω ⟩,
by ChWSi:⟨RESULT:ChWSi:⟨value⟩, ω ⟩
let *sm_preparePass* = **replace** DEST:WSj, RESULT:WSi:⟨value⟩
by PASS:WSj:⟨RESULT:WSi:⟨value⟩, MERGE:Yes⟩
let *sm_setFlag* = **replace-one** MERGE:Yes **by** FLAG_INVOKE

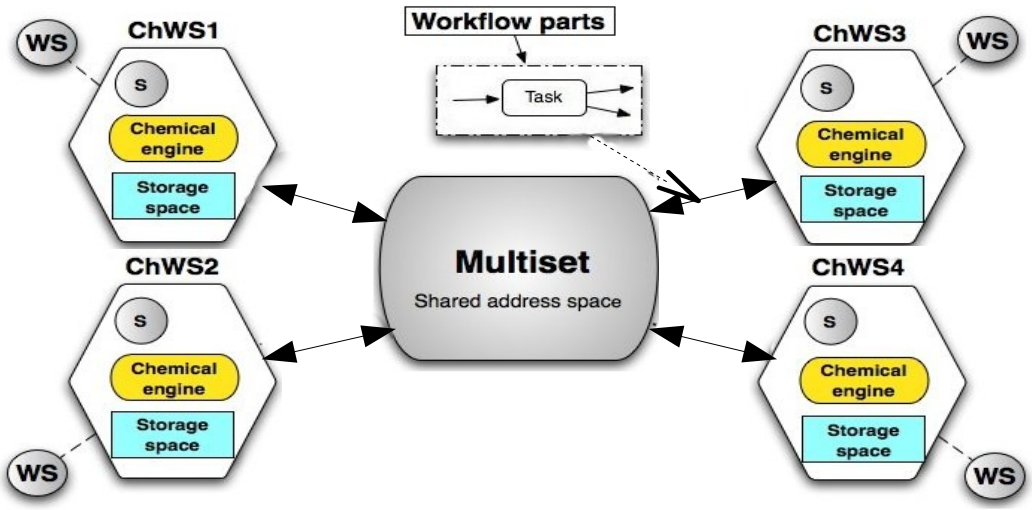
⟨ // Multiset (Solution)
WS1:⟨CALL:S1, PARAM:⟨in1⟩,DEST:WS2, DEST:WS3⟩, // WS1 Sub-solution
WS2:⟨DEST:WS4, **replace** RESULT:WS1:value1 **by** CALL:S2, PARAM:⟨(value1)⟩ ⟩,
WS3:⟨DEST:WS4, **replace** RESULT:WS1:value1 **by** CALL:S3, PARAM:⟨(value1)⟩ ⟩,
WS4:⟨**replace** RESULT:WS2:value2, RESULT:WS3:value3 **by** CALL:S4, PARAM:⟨(value2)⟩
⟩

Decentralized Architecture

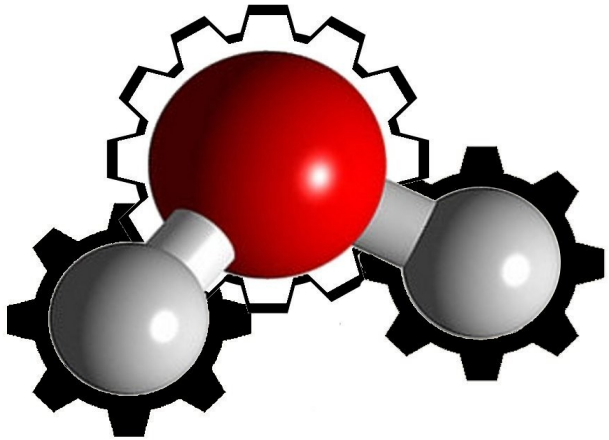




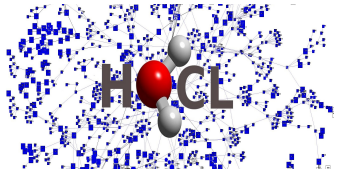
- The multiset becomes a reliable shared address space
  - For nodes to communicate together
  - Reading and writing the multiset for coordination
- Workflow executed in parts by each *Chemical Web Service*
  - Data and control transfer through this shared space
  - Each node co-responsible of the execution



Experimental Evaluation

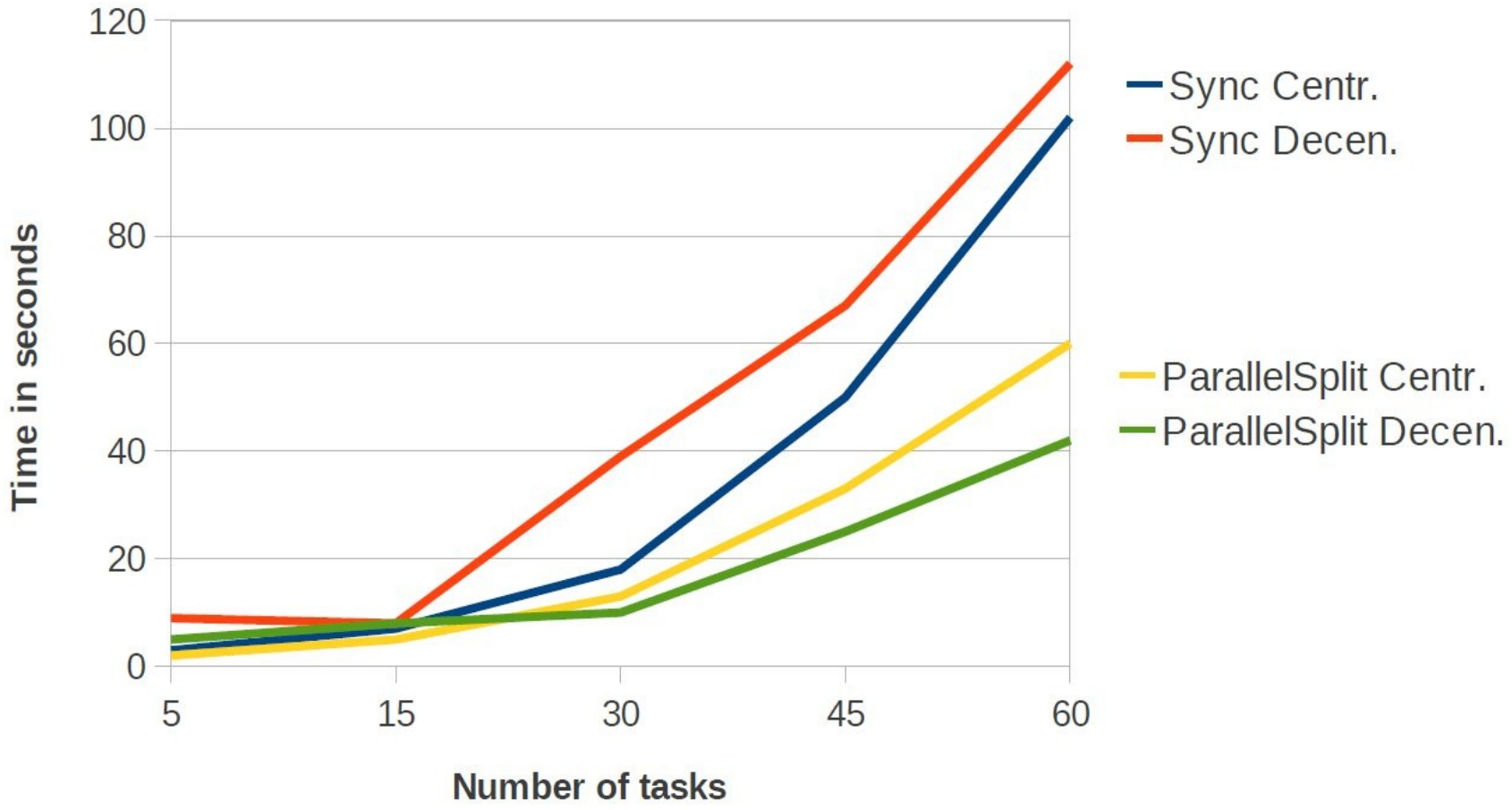


Experiments

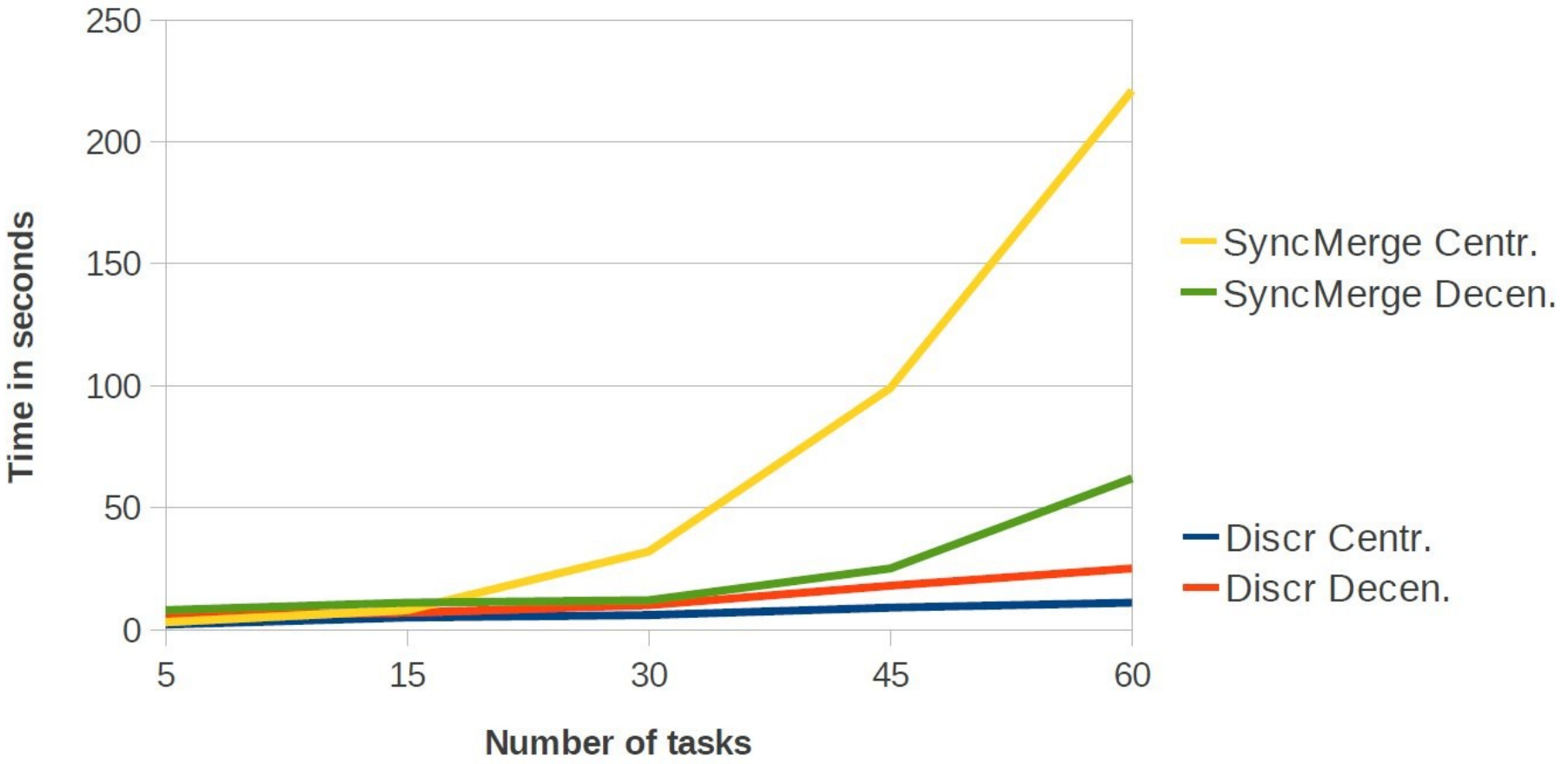


- Both prototypes experimented
- Patterns tested
  - Parallel split
  - Discriminator
  - Synchronization merge
- Size of patterns 5, 15, 30, 45, 60 nodes
- Experiments conducted on the Grid'5000 research infrastructure

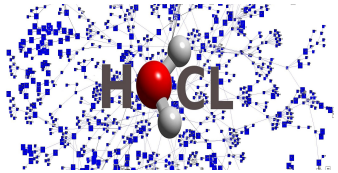
Results (I)


120
100
80
60
40
20
0
Time in seconds
5
15
30
45
60
Number of tasks
Sync Centr.
Sync Decen.
ParallelSplit Centr.
ParallelSplit Decen.

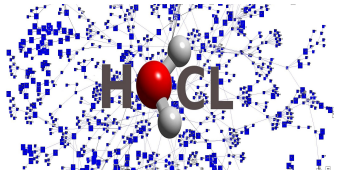
Results (II)


250
200
150
100
50
0
Time in seconds
5
15
30
45
60
Number of tasks
SyncMerge Centr.
SyncMerge Decen.
Discr Centr.
Discr Decen.

Conclusion & On-going works



- New chemically-inspired abstractions for workflow self-coordination
  - Molecular composition
    - Autonomic behavior
    - Decentralization / Self-coordination
- Proof of concepts
  - Two architectures and software prototypes (centralized, decentralized)
  - highlighting the feasibility of such a concept
- On-going studies
  - Towards a P2P multiset
  - New applications envisioned:
    - Workflow scheduling
    - Broader scope – Social networking, Agile Programming Networks



Thank you!

References (I)

[1] J.-P. Banâtre, P. Fradet, and Y. Radenac, “Higher-Order Chemical Programming Style,” in *Unconventional Programming Paradigms*, 2005, 84-95, <http://dx.doi.org/10.1007/11527800_7>.

[2] Mirko Viroli and Franco Zambonelli, “A biochemical approach to adaptive service ecosystems,” Information Sciences, (2009).

[3] Mangala Gowri Nanda, Satish Chandra, and Vivek Sarkar. “Decentralizing execution of composite web services,”, in Proceedings of the 19th conferenceon Object-oriented programming, systems, languages, and applications, ACM, 2004, 170–187.

[4] Rosa Anna Micillo, Salvatore Venticinque, Nicola Mazzocca, and Rocco Aversa. “An Agent-Based approach for distributed execution of composite web services,”, in IEEE International Workshops on Enabling Technologies, IEEE Computer Society, 2008, 18–23.

[5] Weihai Yu. “Consistent and decentralized orchestration of BPEL processes,”, in Proceedings of the 2009 ACM symposium on Applied Computing, 2009, 1583–1584.

[6] Paul A. Buhler and Jose M. Vidal. “Enacting BPEL4WS specified workflows with multiagent systems,” in Proceedings of theWorkshop on Web Services and Agent-Based Engineering, 2004.

[7] Creatis-LRMN biomedical laboratory. http://www.creatis.insa-lyon.fr/site/. Lyon.

[8] G. Berriman, E. Deelman, J. Good, J. Jacob, D. Katz, C. Kesselman, A. Laity, T. Prince, G. Singh, and M. hu Su, “Montage: A grid enabled engine for delivering custom science-grade mosaics on demand,” In Proceedings of SPIE Conference 5487: Astronomical Telescopes, 2004.

[9] BlastReport, http://www.myexperiment.org/workflows/2058.html.

References (II)

[10] Daniel Martin, Daniel Wutke, and Frank Leymann. “A novel approach to decentralized workflow enactment,” in Enterprise Distributed Object Computing Conference, IEEE International, pages 127–136, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[11] Christian Kunze, Sonja Zaplata and Winfried Lamersdorf, “Mobile Process Description and Execution,” in Distributed Applications and Interoperable Systems, 2006.

[12] T. Fahringer, J. Qin, and S. Hainzer. “Specification of Grid Workflow Applications with AGWL: An Abstract Grid Workflow Language,” in Proceedings of the Fifth IEEE International Symposium on Cluster Computing and Grid 2005, Cardiff, UK. (CCGrid 2005), 676–685.

[13] J.-P. Banâtre, T. Priol, and Y. Radenac, “Service orchestration using the chemical metaphor,” Software Technologies for Embedded and Ubiquitous Systems, vol. 5287, pp. 79–89, September 2008.

[14] Gustavo Alonso, C. Mohan, Divyakant Agrawal, and Amr El Abbadi. Functionality and limitations of current workflow management systems. IEEE Expert, 12, 1997.

[15] Girish Chafle, Sunil Chandra, Vijay Mann, and Mangala Gowri Nanda. Decentralized orchestration of composite web services. In Proceedings of the 13th International World Wide Web Conference, (WWW2004), pages 134–143, 2004.

[16] Zsolt Nemeth, Christian Perez, and Thierry Priol. Distributed workflow coordination: molecules and reactions. In Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International, 2006.