Hozo: An Ontology Building Environment

Riichiro Mizoguchi and Kouji Kozaki I.S.I.R., Osaka University

Introduction

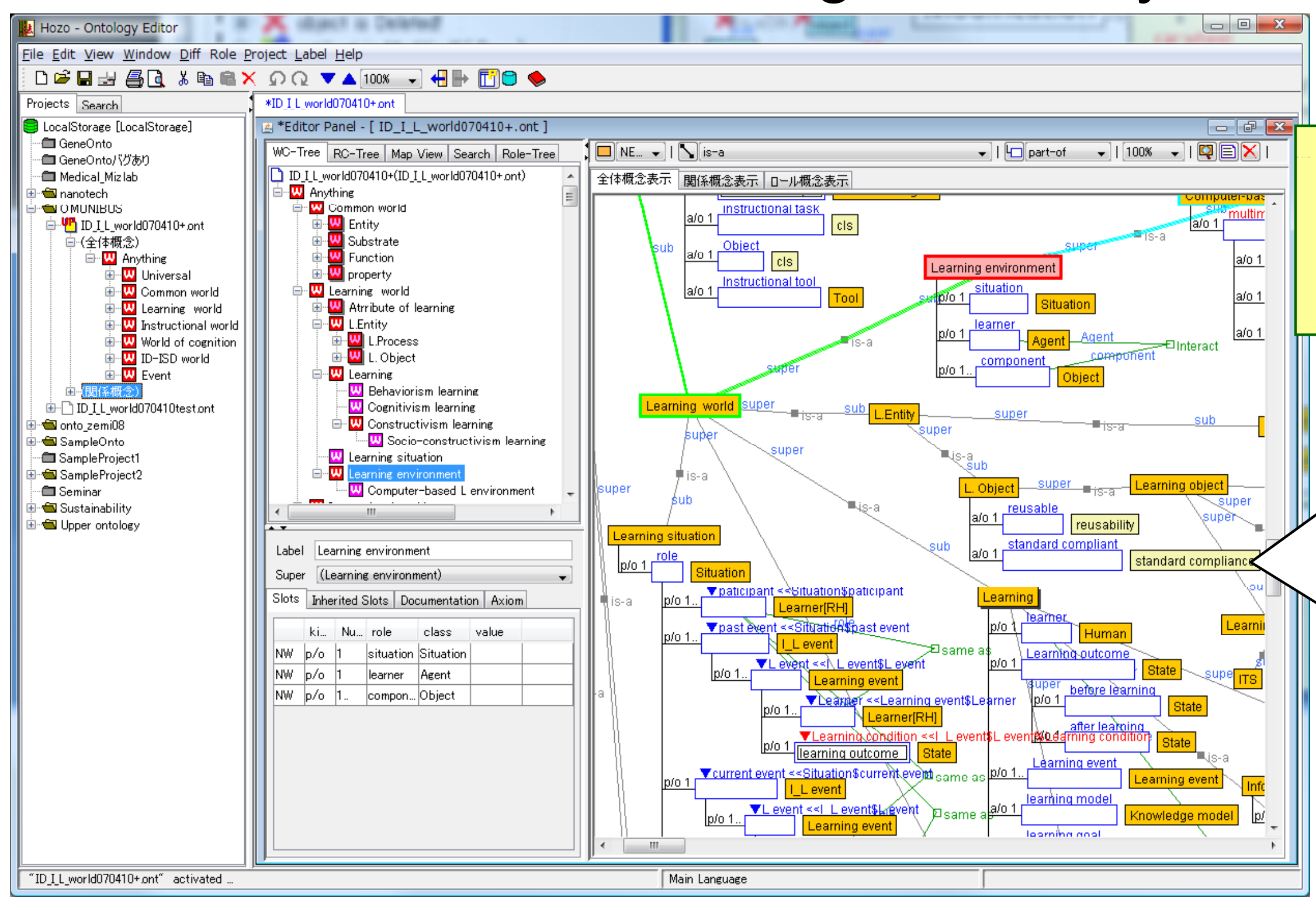
Hozo is a tool for building ontologies in a distributed environment. It has more than 1,500 users in the world and has been used to implement **OMNIBUS** ontology[http://edont.qee.jp/omnibus/doku.php], the world-first heavy-weight ontology of learning and instructional theories.

Required resources:

OS: Windows2000, XP, Vista, Mac OS X or higher
Java: Java 2 Platform, Standard Edition v 1.5.0_10 or higher
Resources necessary for distributed development through internet: WebDAV server (IIS6.0 or higher, not mandatory)

Hozo-Ontology Editor

Ontology Editor is an ontology development tool, based on a fundamental consideration of an ontological theory.

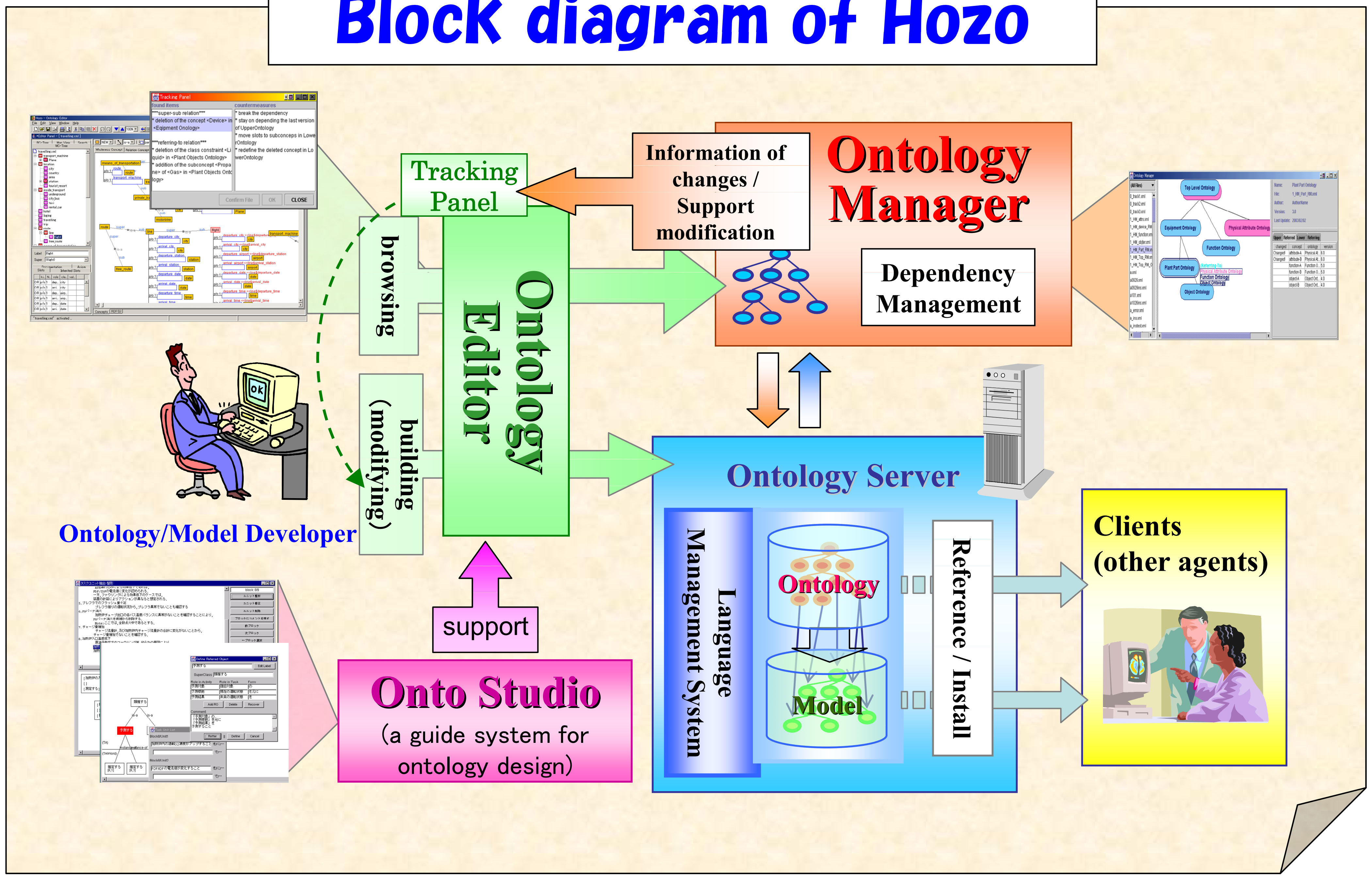


You can GET at www.hozo.jp

Main functionalities

- Browsing and modifying ontologies by simple mouse operations.
- Supporting Multi-windows
- Various ways to brows ontlogies
  - Map view, Zoom, Search
- Printing ontologies
- Exports in RDF(S), OWL
- … etc

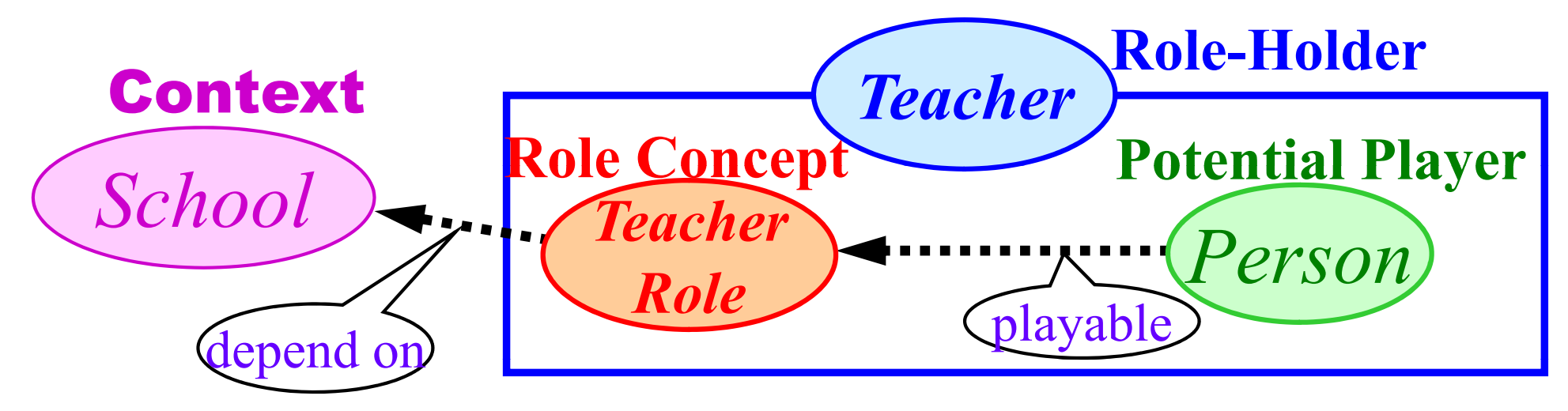
Block diagram of Hozo



The theory of "Role"

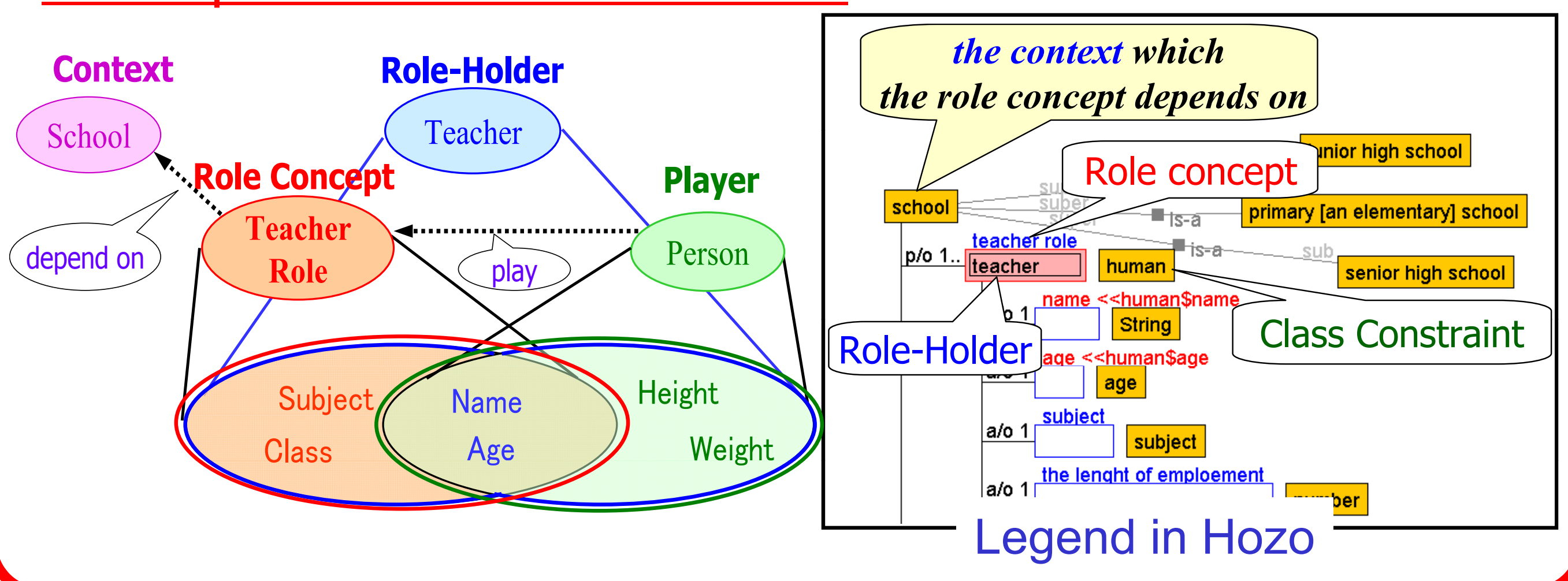
Fundamental scheme of a role and a role holder

*"In a **context**, there are **potential players** who can play **role concepts** and thereby become **role holders**"*
School — Person — Teacher Role — Teacher



Role concept: Concepts ***played by something within a context***.
Potential Player (Class Constraint): A class of things which *are **able to play** an instance of a role concept*.
Role Holder: An entity which ***is actually playing*** a role.

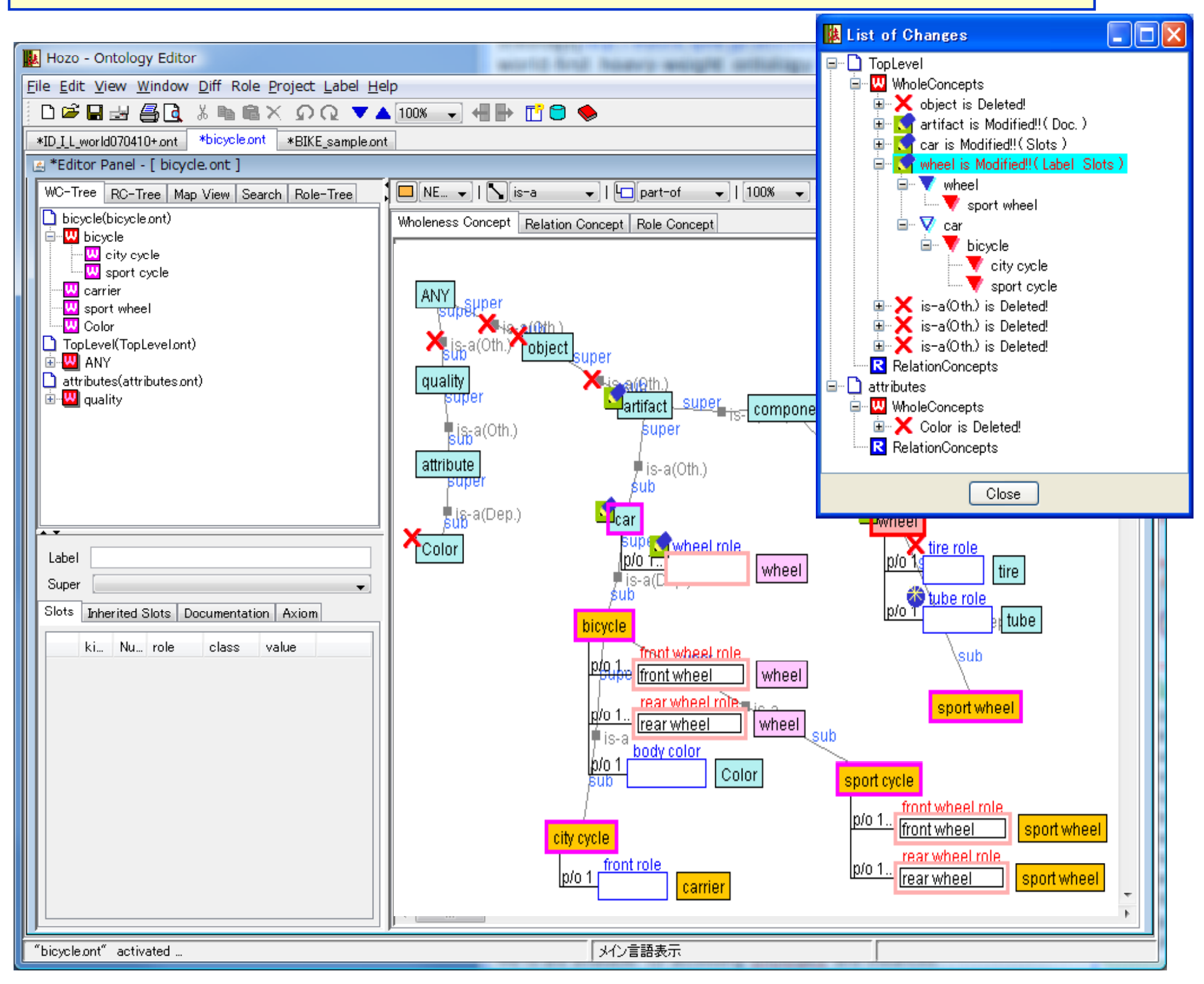
Conceptual Framework of Role



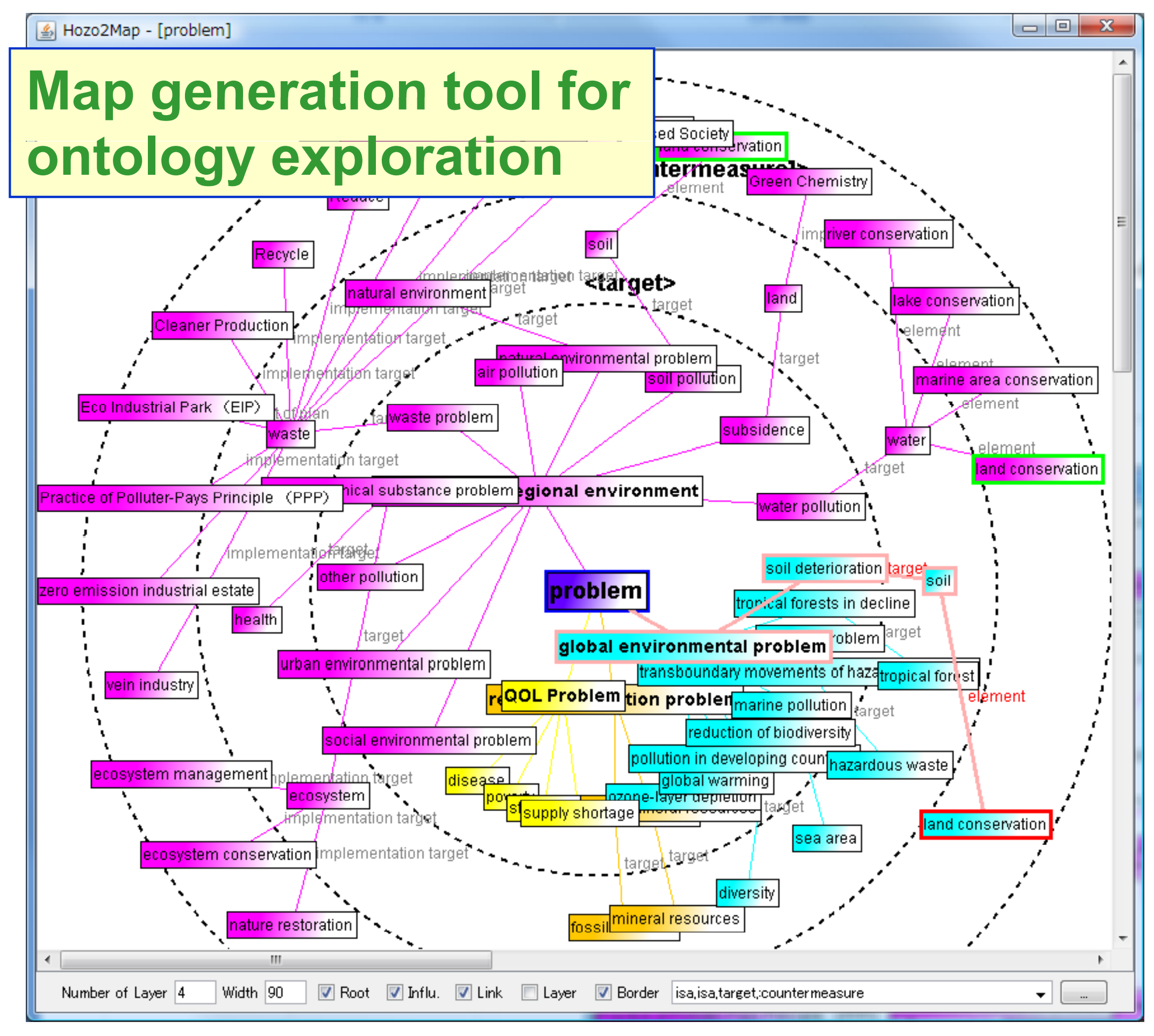
The main characteristics of Hozo

- Unlike OWL, **its conceptual level is closer to that of humans**.
- Single inheritance
- Its representation scheme is based on the frame structure
- It helps users build ontologies with Roles in a natural way supported by the **advanced theory of Roles** [Mizoguchi 2007].
- It is easy to represent a nested structure of slots. (Any slot can have its own slots)
- Inheritance information is explicit and is always accessible.
  - Two ways of inheritance: one from super classes through *is-a* link and the other from *class constraint*
- A friendly GUI is available
- Version management is available with a useful function for displaying changes.
- Ontology building in a distributed environment over internet is supported.
- APIs are available for accessing ontologies and instances.
- An instance model builder is available with the same GUI of the ontology editor. It is useful when you build a model of a particular plant using plant ontology.

Version management with a function for displaying changes



Map generation tool for ontology exploration



Rough schedule of the demo

1. Basic operations using common examples
2. Making nodes, adding/removing slots by inheritance, etc.
3. Inheritance from a super concept
4. Introducing the concept of **Roles** and how to define and use Roles.
5. Inheritance from the class constraint(role-player)
6. Version management with the display function of changes
7. OWL code generation
8. **Map generation tool**
9. Exploration of **OMNIBUS** ontology[OMNIBUS]

Concluding remarks

・Hozo is appropriate for building a heavy-weight ontology, especially, a philosophically-sound ontology rather than a light-weight ontology.

・One important heuristic for building a good ontology is to forget about OWL which is a yet another DL language and to learn what an ontology is language-independently, and then use Hozo.

References

[OMNIBUS] Mizoguchi, R., Hayashi, Y., and Bourdeau, J.: **Inside Theory-Aware and Standards-Compliant Authoring System**, Proc. of SWEL'07, pp. 1-18, 2007.

[Mizoguchi 2007] Mizoguchi R., Sunagawa E., Kozaki K. and Kitamura Y., **A Model of Roles within an Ontology Development Tool: Hozo**, J. of Applied Ontology, Vol.2, No.2, pp.159-179. Sep. 2007.