Object Knowledge for Retrieval and Diagnosis

Yamasaki, S.; Sasakura, M.; and Iwata, K.

Abstract—*We analyze some ideas of objective knowledge as (1) an agent to implement content retrieval, and (2) conjectured diagnosis. We then specify the ideas by a formality so that we may interpret objective knowledge as method. Since the method contains implementable views, we can see that objective knowledge may be not only data, but also procedures. By means of a diagnostic system practice as well as content retrieval, we may acknowledge such an aspect of objective knowledge. As in application of DSM (Diagnostic and Statistical Manual of mental disorder) to decision of disorders caused by syndromes, the expert (who is not always a doctor) may take a role of interaction between the patient and the automated system based on DSM. However, the automated system does not necessarily make the expert capability useful, as long as the expert is just an auxiliary agent between them. We incorporate a significant function (role) into the whole diagnostic process for the expert (agent) to make some conjecture diagnosis against suspicious syndrome, before the query-answer mental stage for each cause syndrome with reference to conjecture diagnosis. In general, we interpret conjecture diagnosis as objective knowledge, while conjecturing is based on reasoning. This manner can be generalized to design scheme for automated diagnosis of two stages: (1) the function of an expert agent, considered as conjecturing diagnosis, and (2) the verification of conjectured diagnosis, whether it is automatic or re-examined from professional views.*

Index Terms—*Intelligent systems (ISY), Knowledge structure, Reasoning*

1. Introduction

From the ontology views with type theory (as in [4]), the object may be identified from the method of procedures. In this paper, we have an outlook on the role of objects as methods, where the task domains are taken from Web usability, object-oriented programming and diagnosis in psychology.

In the field of knowledge representation, procedural aspects in knowledge have been studied as reasonings. The keynote of [21] focused on:

- reasoning in distributed environments and its correctness as procedure, and
- diagnosis as basis of causal theory applied to a consultant system.

In the keynote, we next turned to objective knowledge, apart from procedural aspect of it in reasoning. As in object-oriented programming ([4]), the object recursively involves methods (procedures) as well as objects. Motivated by construction of e-learning system designs, we rather take sequential structure formed by objective knowledge concatenations than recursive structure of objects. With primitive rules of replacements of objects by sequences of objects, how to form structure to denote some sequential process of primitive knowledge processing is a problem.

In this paper, we deal with the problem of how to interpret the objects as methods. As typical objects, we have:

- an agent with keywords (an object) to retrieve the contents in the distributed system whose behaviours are described with a pseudo-language, and
- conjectured diagnosis with DSM ([1]) for detection of mental disorders.

We firstly show that an agent with keywords is both a communication medium and a tool (method) to acquire keywords (knowledge).

We then turn to an illustrative example of a program, where methods contained in an object are evoked. As the second problem, we focus on an object in a diagnostic automation problem:

It is not so easy how to automatically diagnose mental syndromes obtained by means of interaction between the licensed expert (who is not always a doctor) and the patient. In case that the expert may not be always the psychiatrist to possibly prescribe, he or she must contribute to rather redundant interactions with the patient to get some answers to the presented questionnaire, but diagnoses reasoned with answers may not be verified. In such a case, time complexity of interactions and insufficiency in the verified diagnosis caused by the expert frustrates the effect of interactions which it may take more time for.

We consider two stages:

1. a stage to automatically support the expert's knowledge or to implement a part of diagnosis for the expert must be of much help, such that conjectured disorders may

Manuscript received November 26, 2009.

Susumu Yamasaki is with Computer Science Department of Okayama University, Japan (e-mail: yamasaki@momo.cs.okayama-u.ac.jp). Mariko Sasakura is with Computer Science Department of Okayama University, Japan (e-mail: sasakura@momo.cs.okayama-u.ac.jp). Kenichi Iwata is with Computer Science Department of Okayama University, Japan (e-mail: iwa@momo.cs.okayama-u.ac.jp).

be reasoned as diagnoses, and

2. another stage to verify conjectured diagnoses by means of exact question and answer interactions managed by the expert,

such that the expert can make an effective role without any loss through automated conjecturing.

The problem is how to conjecture disorders from claimed syndromes. As a solution, we elaborate a consistent reasoning with a rule for the diagnosis from syndromes and with constraints, which may be applied to a mental disorder diagnosis with formulation of DSM ([1]). If the conjectured diagnosis is reasoned, the diagnosis is interpreted as an object containing methods or methodologies. In this sense, we list up the problem of conjecturing disorders from syndromes and its solution by two stages of diagnosis.

This paper is organized as follows. In Section 2, we present an object as method, concerning the content retrieval with keywords. In Section 3, the way of conjecturing diagnosis is presented. In Section 4, concluding remarks of this paper are made to observe the views on objective knowledge as method.

2. CONTENT RETRIEVAL

We consider the content retrieval by means of key words as an Web usability, where the keywords contain both positive and negative aspects to denote a content. As an agent, we assume a case that a request containing key words searches the Web site pages which involve expected, coherent keywords. For the reliable response of a Web page to the request which can cause an enumeration (apperception) of the page and make the page included in a list (of the request data structure):

- We need an interaction between a managing process with a request as a medium, and multi-site (of a distributed system), where each page has got a recursive link structure in a site.
- Any page supposedly responds to the request as a program with knowledge content (keywords) such that if the keywords of the page are consistent with those of the request, then they are to be merged with those of the request, otherwise (that is, the keywords of the page are inconsistent with those of the request), the page cannot make a reliable response and it is rejected/neglected in the request search.

The request itself searches a consistent page in the sense that their keywords are mutually consistent, and also acquire consistent keywords from the page.

We now make a sketch on the whole system, which consists of a managing program, a request (data structure), and sites with their own pages:

(i) A managing program communicates with a site through a request (data structure) of keywords as a medium. Among communication requirements of sites, only one from a site is selected, and other requirements are excluded until the adopted communication would be over.

(ii) The request is not only a data structure, but also a function to acquire consistent keywords from reliable (acceptable) pages in a site and not to get any keyword at all from rejected pages (note: see (iv) for the ideas of reliability/acceptability and rejection). The request is unique, whether it is regarded as a medium or function, since the keywords contained by it may be amended through visits to site pages.

(iii) Each site in the system contains pages under the site environment.

(iv) Each page of a site involves both a program (to make the request data consistently revised) and keywords. If the page contains consistent keywords with those of the request, it is regarded as reliable or acceptable. Otherwise it is thought of as rejected so that the request is not to be revised.

The request data structure is defined as follows, where it contains a memory to store reliable (acceptable) pages through searching.

Format of Request (BNF)

```
<Request>  ::= Name <p-keywordList>
                   <n-keywordList> <PageList>
<p-keywordList> ::= Keyword+
<n-keywordList> ::= Keyword*
<PageList> ::= Page*
```

The above variables Name, Keyword+, Keyword* and Page* are of data type “string". The function to acquire consistent keywords but not to get inconsistent ones is performed by means of a program equipped with in each page, such that Request contains no program. It may be represented as an XML-file.

An Example of Request

```
<Request>
 <Request-name> Name </Request-name>
 <p-keywordList> <p-keyword> A </p-keyword>
          <p-keyword> B </p-keyword>
          .....
 </p-keywordList>
 <n-keywordList> <n-keyword> a </n-keyword>
          <n-keyword> b </n-keyword>
          .....
 </n-keywordList>
 <PageList>
 </PageList>
</Request>
```

The "Manager" (managing program) has an interaction with a site in a distributed multi-site system, with the data structure "Request".

Manager

```
Manager::Managing
 begin
  read Request;
  u = 0;
```

```
  while u = 0 do
   begin
    if requirement_i exists then
     begin
      select requirement_i;
      send Request (or none unless Request
                    exists)
          to Site_i
     end;
    if Request reaches then
     begin
      if "to be continued" then
       take Request
      else
       u = 1
     end
   end
 end
```

The Manager communicates with more than two sites in distributed environments, where each $Site_i$ keeps all the pages to consistently respond to Request. If Manager is interpreted as virtual, a mutual communication between sites are considered as available, where such a mutual one is later formulated.

$Site_i$

```
Site_i::Assigning
 begin
  send requirement_i to Manager;
  v = 0;
  while v = 0 do
    if Request reaches then
     begin
      take Request;
      for all pages do
       begin
        assign Request to a page;
        revise Request
       end;
       send Request to Manager;
       if "to be continued" then
        send requirement_i to Manager
       else
        v = 1
     end
 end
```

Each page takes the form consisting of a program and (a data structure of) keywords. The following Name, Keyword+, Keyword* and ProgramName are of data structure "string". The above Page can be expressed in an XML form.

Format of Page (BNF)

```
  <Page> ::= Name <p-keywords> <n-keywords>
             ProgramName
  <p-keywords> ::= Keyword+
  <n-keywords> ::= Keyword*
```

An Example of Page (XML)

```
<Page>
 <Name> PageName </Name>
 <p-keywords> <keyword> A </keyword>
  <keyword> B </keyword>
  .....
 </p-keywords>
 <n-keywords> <keyword> a </keyword>
  <keyword> b </keyword>
  .....
 </n-keywords>
 <ProgramName> Checking </ProgramName>
</Page>
```

The program of "Page" can be represented as follows. It acquires consistent keywords.

Program of Page

```
Page::Checking
 begin
  if p-keywords and Request::n-keywordList
       have some common keyword
       then return Request
  else if n-keywords and Request::p-keywordList
          have some common keyword
       then return Request
  else
   begin
    Request::p-keywordList =
          the merge of  Request::p-keywordList
                and p-keywords are merged;
    Request::n-keywordList =
          the merge of Request::n-keywordList
                and n-keywords are merged;
    add this page to Request::PageList;
    return  Request
   end
 end
```

If we regard the program "Manager" as implicit and functionally virtual in a distributed system of sites, then we can have a communication between two sites (through the program Manager). Then we interpret Request as a situation (or a state) transitive to another by means of the page: If the page is rejected, there is no transition. On the other hand, the empty transition may occur when Request is not amended through keywords of a page, that is, keywords of Request covers all the keywords of the page.

Note that a reliable (acceptable) page sequence can be kept in the data structure Request, even if the program Manager is supposedly implicit (virtual) for site communications with Request. This page sequence is required to be recognized as an object sequence in the calculus of this manuscript.

3. *Conjecturing Diagnosis*

Based on DSM ([1]), the structured clinical interview for DSM (SCID) contains the rules in questionnaire form like a flowchart [7], where the expert can be wise to interact with the patient and to draw the answers to the questionnaire for diagnosis. There may be an automated way from the answers to some diagnosis, with the aspects:

- answers to the questionnaire as objective knowledge
- diagnosis as method

In this setting, there is a problem that the expert may not be always the licensed psychiatrist to possibly prescribe. In addition, the way may be often of high computational complexity in reasoning when the interaction between even the good expert and the handicapped patient might be complex. These are to be re-considered with reference to information system designs. To cope with such problems, we must:

(i) make the expert's skill more adopted, and
(ii) make the whole diagnosis effective.

We have an suggestion to take two stages for (partially) automated diagnosis:

(a) to conjecture diagnosis by consistent reasoning from assumed syndromes and constraints, and
(b) to verify conjectured diagnosis by means of the questionnaire with the expert.

We then regard conjectured diagnosis as objective knowledge, while conjecturing is a method without inconsistency.

We construct a prototype system for assisting to make a diagnosis of a mental disorder based on DSM and SCID. The system consists of the two stages mentioned above. We call the former (a) as "the first stage" and the latter (b) as "the second stage". We describe the details of the system.

The First Stage

We pick up symptoms from the description of DSM and construct a knowledge base for the first stage. Some symptoms are effective in distinguishing a disorder from others. Other symptoms are common among disorders. Some symptoms must be seen for diagnosing some disorders, and other symptoms may be seen in some cases and may not be seen in other cases of a disorder. Therefore, we choose some characteristic symptoms and we use consistent reasoning to find suspicious disorders from them.

The form of consistent reasoning is

$$A \triangleright B_1,\ldots,B_m,C_1,\ldots,C_n$$

where:

- A is a disorder.
- $B_1,\ldots,B_m,C_1,\ldots,C_n$ are symptoms.

The manual may include descriptions among relations of disorders and symptoms. The consistent reasoning can be constructed in the manner.

On condition that

(i) there is a rule $A \triangleright B_1,\ldots,B_m,C_1,\ldots C_n$,

(ii) $not\ C_i$ fails $(1 \le i \le n)$,

(iii) there are symptoms $A_1,\ldots A_l,B_1,\ldots B_m$, and

(iv) A_1 and A_2 and $\ldots$ and A are consistent,

reason the disorder A as diagnosis.

Because $not\ C_i$ fails, we can consistently assume a symptom $C_i\,(1 \le i \le n)$ such that symptoms

$$B_1,\ldots,B_m,C_1,\ldots C_n$$

are to be listed up. On the other hand, assumed symptoms $A_1,\ldots,A_l$ would never deny the conjectured disorder A, because the conjunction of A_1 and A_2 and $\ldots$ and A_l and A is consistent.

For a list of claimed symptoms, consistent reasonings may make the expert learn more than 2 disorders such that conjectured disorders are gathered into a set as conjectured diagnosis.

In the prototype system we developed, the first stage is implemented simply:

1. The system lists up all symptoms related disorders.
2. An expert decides a threshold rate.
3. An expert checks all symptoms that a patient has.
4. If the number of checked symptoms related to a disorder goes over the threshold rate, the system picks up the disorder.
5. All disorders the system picked up are displayed on the screen for the second stage.

The Second Stage

The second stage will start for a disorder that is listed up by the first stage of the system. The expert firstly looks at the list of the suspicious disorders, such that the expert can pick up a disorder that seems to be the possibly true disorder of the patient. Then the verification stage (the second stage) starts on here.

The system asks a question that follows SCID, while the expert answers. This answer includes a thought of the expert. The expert interacts with the patient, and judges how to answer to the question from the system. This means the expert interacts with the system also, not only with the patient. The system stored the answer to the database, and shows a new question to the expert. The system also shows the process of the flow. This helps the expert to judge the disorder of the patient. The expert can think about the answer of the question from the system too.

The cycle is continued until the flow reaches the end. At the end of the flow, the system displays if the suspicious disorder of the patient would truly match to the symptoms or not.

The questions in the second stage of SCID have 3 values. The answer for the question is one of ?, -, or +. The answer is interpreted by : ? means inadequate information, - means absent or sub-threshold (negative), and + means present (positive). If the answer is +, the patient has the symptom asked by the question.

The questions of SCID can be categorized into two types following:

Type 1 The next question is decided by the answer of this question.

Type 2 The next question is decided by the answers of the previous questions including this question. In this type of questions, sometimes more than 10

previous questions are related to decision of the next question.

The system supports these two types of questions and controls flows in the second stage.

System Design

The proposed system aims to assist the psychiatrist in making a diagnosis. It is designed as a Web-based system. The architecture of the system is shown in Figure 1.

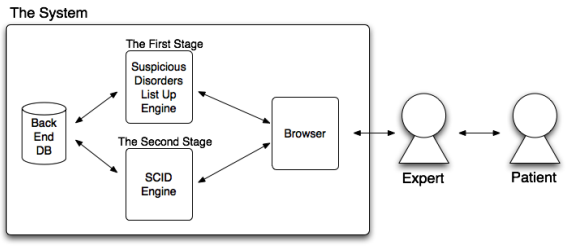

Figure 1 The system architecture of a diagnosis system.

The system consists of four parts: Back End DB, Suspicious disorders listing up engine, SCID engine and Browser. Back End DB keeps knowledge for diagnoses and records of diagnoses. Suspicious disorders listing up engine is an implementation of the first stage. SCID engine is an implementation of the second stage. Browser is an interface between the system and a psychiatrist. Any popular Web browser can be used as the Browser.

Figure 2, 3 and 4 show snapshots of the system. Figure 2 is a snapshot of the first stage. A psychiatrist asks and observes symptoms of a patient and checks in boxes of present symptoms. Figure 3 presents the result of Figure 2: a list of suspicious disorders. When the psychiatrist clicks the "go" button in Figure 3, the second stage starts. Figure 4 is a snapshot of the second stage. In the second stage, questions are displayed on the screen in sequence. The psychiatrist makes inputs of answers for the questions by checking an appropriate item, such that the result flows: the patient has the suspicious disorder or not.

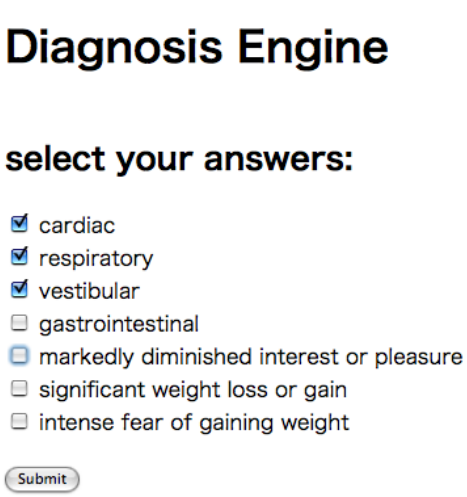

Figure 2 The first step.

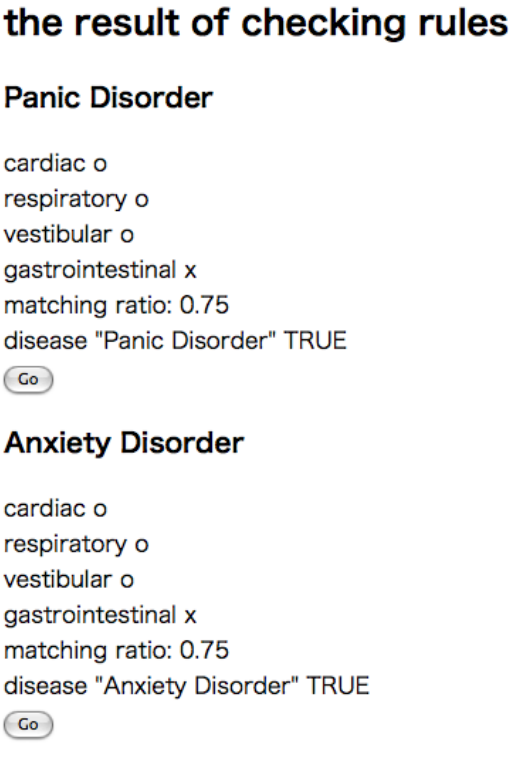

Figure 3 The suspicious disorders.

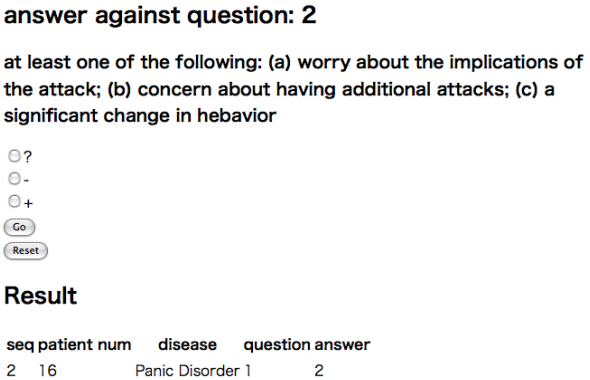

Figure 4 The second step.

4. CONCLUDING REMARKS

With reference to [21], we can mention the same ideas:

As regards the procedural aspects, we have our established results on (a) procedural correctness in distributed logic programming and (b) acquisition function in a consultant system with causal theory. These aspects are included in automation of consultant systems ([18], [19]).

For structural analysis and synthesis of objective knowledge and through the content retrieval problem, we present an abstract framework of (c) the distributed system.

Through the subjects in (a)-(c), semantic issues are the common item for formality and abstraction with applications. For semantical conditions, procedural knowledge is based on model theory in logic, and sequential knowledge is owing to the assumed rule of follower relation. As some interaction is involved in the grammatical constraint system ([20]), human machine interaction and its abstraction should be made clearer for the contributions to the social system developments and robotics, different from agent technology refinements and effects.

In a strategy scheme like the structure of Section 2, the objective knowledge may be concerned with:

- space
- time

The more expressive, the more complicated. But as in action semantics ([13]), the treatments of space and/or time are required.

As a problem left for study, we have how to organize a formal system including situations which are described by logical formulae (as in [14]).

We also deal with a practical problem of how to organize conjectured diagnosis before its verification for the mental disorder detection. The conjecturing is a method, while the conjectured is an object such that an object as method is seen when we treat conjectured disorders from syndromes.

Because objective knowledge and reasoning contain diagnoses as methodologies in artificial intelligence, the views of this paper on objective knowledge as method are thrown into significant points as well as ontology notions, compared with the frameworks:

1. The logical analysis standpoint has contained a wide range of formal systems since its organization ([8], [10], [12] [16]), including hybrid logic ([2], [3]) and event calculi ([5], [6]).
2. Action and knowledge have been in details studied ([13], [15]).
3. The agent technologies in [9], [11], [17] are formulated from functional and algebraic aspects.

REFERENCES

[1] American Psychiatric Association, Diagnostic and Statistical Manual of Mental Disorders, 4th Edition Text Revision DSM-IV-TR, 2000.

[2] Areces,C. and Blackburn,P., Repairing the interpolation in quantified logic, Annals of Pure and Applied Logic, 123, 287--299, 2003.

[3] Brauner,T., Natural deduction for hybrid logics, J. of Logic and Computation, 14, 329--353, 2004.

[4] Bruce,K.B., Foundations of Object-Oriented Languages, MIT Press, 2002.

[5] Cervesato,I., Chittaro,L. and Montanari,A., A general modal framework for the event calculus and its skeptical and credulous variants, Proc. of 12th European Conference on Artificial Intelligence, pp.12--16, 1996.

[6] Dean,T. and Boddy,M., Reasoning about partially ordered events, Artificial Intelligence, 36, pp.375--399, 1988.

[7] First,M.B., Spitzer,R.L., Gibbon,M. and Williams,J.B., User's Guide for the Structured Clinical Interview for DSM-IV Axis I Disorder, American Psychiatric Press Inc., 2002.

[8] Genesereth,M.R. and Nilsson,N.J., Logical Foundations of Artificial Intelligence, Morgan Kaufmann, 1988.

[9] Hoare,C.R.A., Communicating Sequential Processes, Prentice-Hall, 1986.

[10] Lloyd,J.W., Foundations of Logic Programming, 2nd, Extended Edition, Springer-Verlag, 1993.

[11] Milner,R., Communication and Concurrency, Prentice-Hall, 1989.

[12] Minker,J. (ed.), Foundations of deductive Databases and Logic Programming, Morgan Kaufmann Publishers, Inc., 1987.

[13] Mosses,P.M., Action Semantics, Cambridge University, 1992.

[14] Reiter,R., A theory of diagnosis from first-principles, Artificial Intelligence, 32, 1, pp.57--95, 1987.

[15] Reiter,R., Knowledge in Action, The MIT Press, 2001.

[16] Russell,B., History of Western Philosophy, Routledge, 2000.

[17] Russell,S. and Norvig,P., Artificial Intelligence--A Modern Approach--, Prentice-Hall, 1995.

[18] Sasakura,M. and Yamasaki,S., An explanation reasoning procedure applicable to loop transformation in compiler, Proc. of ACM ESEC/FSE International Workshop on Intelligent Technologies for Software Engineering, WITSE 03, pp.34--39, Helsinki, 2003.

[19] Yamasaki,S., Automated consultant to acquire knowledge for human interface, Proc. of International Conferences on Advances in the Internet, Processing, Systems and Interdisciplinary Research, IPSI-2005 Amsterdam (CD-ROM), 2005.

[20] Yamasaki,S., An interactive constraint system and its relation to logic in AI, Proc. of 19th Belgium/Netherlands Artificial Intelligence Conference, pp.299-306, Utrecht, 2007.

[21] Yamasaki,S. and Sasakura,M., Knowledge: Procedural and objective aspects, Proc. of International Conferences on Advances in the Internet, Processing, Systems and Interdisciplinary Research, VIPSI-2008 (CD-ROM), Santa Margherita Ligure, 2008.

Susumu Yamasaki received Eng.D in information science from Kyoto University in 1980. In 1985/86, he was a visiting fellow at Dept. of Computer Science, University of Warwick, U.K. In 1987, he joined Okayama University as a professor in computing and artificial intelligence. His major includes theoretical computer science and automated reasoning.

Mariko Sasakura received Eng.D in information science from Kyushu University in 2000. She is an assistant professor of Okayama University in computing and artificial intelligence.

Kenichi Iwata is a PhD candidate at Graduate School of Natural Science and Technology, Okayama Univertisy.