Expert system architecture pdf

I'm not robot

reCAPTCHA

Continue

Next: Choice Challenge: Design Experts Previous: Design Expert Figure 8.1 reveals the most important modules that make up the rules based on the expert system. The user interacts with the system through a user interface that can use menus, natural language, or any other style of interaction). The output engine is then used to discuss both the expertise (extracted from our friendly expert) and the data specific to a particular problem that is being addressed. Expert knowledge is usually in the form of a set of IF-THEN rules. Specific case data include both user-provided data and partial conclusions (along with certainty measures) based on this data. In a simple system based on fast-forward rules, specific data will be elements of working memory. Almost all expert systems also have an explanation subsystem that allows the program to explain its reasoning to the user. Some systems also have a knowledge editor that helps an expert or knowledge engineer easily update and verify a knowledge base. One of the important things about expert systems is how they (usually) separate domain-specific knowledge from more general methods of reasoning and presentation. The general purpose bit (in the dotted box in the picture) is called the shell of the expert system. As we can see in the picture, the shell will provide a output engine (and a knowledge presentation scheme), a user interface, an explanation system, and sometimes a knowledge base editor. Given the new kind of problem to solve (say, car design), we can usually find a shell that provides the right support for the problem, so all we need to do is provide expert knowledge. There are many commercial expert system shells, each of which is suitable for a slightly different range of problems. (The expertise of systems in industry includes both writing expert system shells and writing expert systems using projectiles.) The use of shells to record expert systems tends to significantly reduce development costs and time (compared to writing an expert system from scratch). Next: Problem Selection: Design Experts Previous: Designing Expert Architecture Expert System Typical Expert Architecture System is shown in Figure 11.1. The knowledge base contains specific knowledge of the area, which is used by the expert to draw conclusions from the facts. In the case of a rules-based system of experts, knowledge in this area is expressed in the form of a number of rules. The explanation system provides the user with information about how the output engine came to its conclusions. This can often be important, especially if the advice given is critical, with the medical diagnostic system. If the system used erroneous reasoning to reach its conclusions, the user can see it by studying the data, data, System. The fact sheet contains specific data that should be used in a particular case to obtain an opinion. In the case of a system of medical experts, this information will contain the information received about the patient's condition. The expert system user interacts with him through a user interface that provides access to the output engine, the explaining system, and the knowledge base editor. The output engine is part of a system that uses rules and facts to draw conclusions. The output engine will use a forward-looking, reverse chain or a combination of the two factors to deduce the data available to it. The knowledge editor allows the user to edit the information contained in the knowledge base. The knowledge editor is not usually given to the end user of the system, but is used by a specialist engineer or expert to provide and update the knowledge contained in the system. ShellNote's expert system notes that Figure 11.1 in the expert system shell contains parts of the expert system that do not contain domain-specific or case-specific information. This shell is a common toolkit that can be used to create a number of different expert systems, depending on what knowledge base is added to the shell. CLIPS (C Language Integrated Production System) is an example of this shell. Other common examples include OPS5, ART, JESS and Eclipse. Symbolics Lisp Machine: an early platform for expert systems. In artificial intelligence, an expert system is a computer system that simulates a person's ability to make decisions. Expert systems are designed to solve complex problems through reasoning through knowledge bodies, presented primarily as rules rather than through the usual procedural code. The first expert systems were established in the 1970s and then developed in the 1980s. The expert system is divided into two subsystems: the output engine and the knowledge base. The knowledge base represents facts and rules. The withdrawal engine applies the rules to known facts to conclude new facts. Output engines may also include explanation and debugging capabilities. History Early development Shortly after the dawn of modern computers in the late 1940s - early 1950s, researchers began to understand the huge potential of these machines for modern society. One of the first tasks was to make such a machine able to think like people. In particular, to make these machines able to make important decisions the way people do. The medical/medical field has presented a tantalizing challenge to these machines for making medical diagnostic decisions. So at the end of the Immediately after the information age was fully entered, researchers began experimenting with the prospect of using computer technology to emulate human decision-making. For example, biomedical researchers have begun to create computer systems for diagnostic applications in medicine and biology. These early diagnostic systems used patient symptoms and laboratory test results as inputs to obtain diagnostic results. These systems were often described as early forms of expert systems. However, the researchers realized that there were significant limitations in using traditional methods, such as flow diagrams (12) (12) (13) statistical pattern comparisons, 14 or probability theory. Formal introduction - Later developments This previous situation has gradually led to the development of expert systems that used knowledge-based approaches. These expert systems in medicine were the myCIN expert system, the expert system INTERNIST-I, and later, in the mid-1980s, CADUCEUS. Expert systems were formally introduced around 1965 by the Stanford Heuristic Programming Project under the leadership of Edward Feigenbaum, sometimes referred to as the father of expert systems; the other key players were Bruce Buchanan and Randall Davis. Stanford researchers have been trying to identify areas where expertise has been highly rated and complex, such as diagnosing infectious diseases (mycin) and identifying unknown organic molecules (Dendral). The idea that intellectual systems derive their strength from the knowledge they possess, rather than from the specific formalisms and withdrawal schemes they use, as Feigenbaum said, was at the time a significant step forward, as past research focused on genetic computational methods, culminating in attempts to develop very general solutions to problems (primarily Allen Newell and Herbert). Expert systems were among the first truly successful forms of artificial intelligence (AI) software. Research in expert systems has also been active in France. While in the U.S. the focus tends to be on rule-based systems, first on systems rigidly coded over LISP programming environments and then on expert system shells developed by vendors such as Intelcorpli, in France the research focused more on systems developed at Prolog. The advantage of the projectiles of the expert system is that it is somewhat easier for them to use non-programmers. The advantage of Prolog environments is that they are not oriented only to the rules if in this case; The Prolog environments provided much better a full logical environment of the first order. In the 1980s, expert systems expanded. Universities offered expert courses, and two-thirds of Fortune 500 companies used the technology in day-to-day business activities. The interest has been international Generation Computer Systems project in Japan and increased funding for research in Europe. In 1981, the first IBM PC with PC DOS was introduced. The imbalance between the high availability of relatively powerful chips in PCs, compared to the much more expensive cost of computing power in mainframes that dominated the corporate IT world at the time, created a new type of architecture for enterprise computing, called the client-server model. Calculations and reasoning can be made at a price that is ignorant with a PC. This model also allowed businesses to bypass corporate IT departments and directly create their own applications. As a result, the client server has a huge impact on the market of expert systems. Expert systems have already been emitted in much of the business world, requiring new skills that many IT departments do not have and do not want to develop. They were naturally suitable for new PC shells that promised to put application development in the hands of end users and experts. Until then, the main medium for the development of expert systems was high-quality Lisp machines from Xerox, Symbolics and Texas Instruments. With the growth of computer and client server computing, vendors such as Intellicorp and Inference Corporation have shifted their priorities to developing PC-based tools. In addition, new vendors, often funded by venture capital (such as Aion Corporation, Data Neuron, Exsys and many others), have regularly started to emerge. created by several expert-logical designers. SID has expanded the rules and created the software logic of synthesis of procedures many times greater than the rules themselves. Surprisingly, the combination of these rules led to a common design that surpassed the capabilities of the experts themselves, and in many cases surpassed human counterparts. While some rules contradicted others, the top-level control options for speed and area provided a tie-break. The program was highly controversial, but was nonetheless used because of the project's budget constraints. It was discontinued by logical designers after the completion of the VAX 9000 project. In the years leading up to the mid-1970s, expectations about what expert systems can achieve in many areas tended to be very optimistic. At the beginning of these early studies, the researchers hoped to develop fully automatic (i.e. fully computerized) expert systems. People's expectations of what computers can do were often too idealistic. This radically changed after Richard M. Karp published his breakthrough breakthrough Reductiveness among combinatorial problems in the early 1970s. Its findings describe what computers can do and what they can't do. Many of the computational problems associated with this kind of expert systems have certain pragmatic limitations. These findings laid the groundwork for the following developments on the ground. In the 1990s and beyond, the term expert system and the idea of an autonomous artificial intelligence system largely fell out of the IT lexicon. There are two interpretations of this. One is that expert systems have failed: the IT world has gone further because expert systems have not fulfilled their more bloated promises. Another is the mirror opposite that expert systems were simply victims of their success: because IT professionals understood concepts such as rule engines, such tools migrated from autonomous tools to the development of expert special purpose systems to be one of many standard tools. Many of the leading major business application package providers (such as SAP, Siebel and Oracle) have integrated expert system capabilities into their product set as a way of setting business logic - rule engines are no longer just for determining the rules that the expert will use, but for any type of complex, unstable and critical business logic; they often go hand in hand with business process automation and integration environments. Modern approaches to expert systems limiting the previous type of expert systems have encouraged researchers to develop new types of approaches. They have developed more efficient, flexible and powerful approaches to simulate human decision-making. Some of the approaches developed by the researchers are based on new methods of artificial intelligence (AI), and in particular in the approaches of machine learning and data analysis with the feedback mechanism. This discussion of the section on shortcomings is connected. Modern systems can incorporate new knowledge more easily and thus update themselves easily. Such systems can better generalize existing knowledge and deal with vast amounts of complex data. Related to this is the subject of big data here. Sometimes these kinds of expert systems are called smart systems. The architecture of the software, illustrating the example of the back chain, forms a master's thesis in 1990. Expert systems were the first commercial systems to use knowledge-based architecture. The knowledge-based system essentially consists of two subsystems: a knowledge base and a output engine. The knowledge base presents facts about the world. In early expert systems such as Mycin and Dendral, these facts were presented mainly as flat statements about variables. In the expert systems developed with commercial shells, the knowledge base took over more structure and used concepts from object-oriented programming. The world was presented as classes, subclasses, and instances and approvals were replaced by the values of instances of objects. The rules worked by requesting and approving the values of objects. The output engine is an automated reasoning system that assesses the current state of the knowledge base, applies the relevant rules, and then approves new knowledge in the knowledge base. The output engine may also include the ability to explain, so that it can explain to the user the chain of reasoning used to obtain a specific output by tracking back beyond the shooting rules that led to the approval. There are basically two modes for the engine output: the advanced circuit and the reverse circuit. Different approaches dictate whether the output of the engine is controlled by the previous (left side) or the subsequent (right side) rule. In the forward chain of the previous fire and approves the subsequent. For example, consider the following rule: R 1 : M a n (x) ⟹ M o t t l (x) Displaystyle R1: mathit (man) (x) implies matet Mortal (x) A simple example of a forward chain would assert a person (socratic) into the system and then trigger the withdrawal engine. It will match R1 and claim Mortal (Socrates) into the knowledge base. The reverse chain is a little less straight forward. In reverse chain the system examines possible findings and works backwards to see if they can be true. So if the system tries to determine if Mortal (Socrates) is true it would find an R1 and request a knowledge base to see if the person (Socrates) is true. One of the first innovations in the shells of expert systems was the integration of output engines with the user interface. It can be especially powerful with a backward chain. If the system needs to know a specific fact but does not know, then it can simply generate the input screen and ask the user if the information is known. So in this example, he can use R1 to ask the user if Socrates was human and then use that new information accordingly. The use of rules to explicitly represent knowledge also explained the ability. In the simple example above, if the system used R1 to claim that Socrates was Mortal, and the user wanted to understand why Socrates was mortal, they could request the system, and the system would look back at the rules that shot to trigger approval and present these rules to the user as an explanation. In English, if the user asked: Why Socrates Mortal? The system will respond: Because all men are mortal and Socrates is a man. An important area for research has been the generation from the knowledge base in natural English, not just by showing more formal but less intuitive rules. [40] [40] expert systems have evolved, many new methods have been incorporated into different types of engine outputs. Some of the most important ones were: Maintaining the Truth. These systems record dependencies in the knowledge base so that, when the facts change, the dependent knowledge can be changed accordingly. For example, if the system learns that Socrates is no longer known as a person, it will undo the claim that Socrates is mortal. Hypothetical reasoning. In this, the knowledge base can be divided into many possible views, as well as worlds. This allows the output engine to explore multiple possibilities in parallel. For example, the system may want to examine the implications of both statements, what will be true if Socrates is a person, and what will be true if he is not? Uncertainty systems. One of the first extensions of simply using rules to present knowledge was also to link the probability to each rule. So, do not claim that Socrates is mortal, but to argue Socrates may be mortal with some probability of value. Simple probabilities have been expanded in some systems with complex mechanisms for vague reasoning, such as fuzzy logic, and a combination of probabilities. It's a classification on theology. With the addition of object classes to the knowledge base, a new type of reasoning is possible. Along with just talking about the values of objects, the system can also talk about the structure of the object. In this simple example, a person can represent a class of objects, and R1 can be redefined as a rule by defining the class of all men. These types of special purpose engines are called classifiers. Although they have not been highly used in expert systems, the classifiers are very powerful for unstructured volatile domains, and are a key technology for the Internet and the emerging Semantic Web. The benefits of a knowledge-based system are to ensure that the critical information needed to run the system is explicit, not implicit. In a traditional computer program, logic is built into code, which can usually only be considered by an IT professional. Through an expert system, the goal was to specify the rules in a format that was intuitive and easily understandable, reviewed and even edited by domain experts, not IT experts. The benefits of this apparent presentation of knowledge were rapid development and ease of service. Easy service is the most obvious advantage. This was achieved in two ways. First, by eliminating the need to write plain code, you can avoid many common problems that can be caused by even small changes in the system with expert systems. In fact, the logical flow of the program (at least at the highest level) was just a given to the system, just on the output engine. This also caused a second advantage: rapid prototyping. With the help of an expert shell system, it was possible several rules and a prototype developed in days rather than months or years, usually associated with complex IT projects. The claim for an expert system that is often manufactured is that they remove the need for trained programmers and that experts can develop systems on their own. In fact, it was rarely, if ever true. Although the rules for an expert system are more clear than typical computer code, they still have a formal syntax where an inappropriate comma or other character can cause havoc, like any other computer language. In addition, as expert systems have moved from lab prototypes to deployment in the business world, integration and maintenance issues have become much more important. Inevitably, there are requirements to integrate and use large outdated databases and systems. To achieve this, integration required the same skills as any other type of system. Disadvantages The most common disadvantage cited for expert systems in academic literature is the problem of acquiring knowledge. Getting domain experts time for any software application is always difficult, but for expert systems it was especially difficult because experts were by definition highly valued and in constant demand by the organization. As a result of this problem, much research in recent years of expert systems has focused on tools to acquire knowledge to help automate the design process, debugging and maintaining the rules defined by experts. However, if you look at the life cycle of expert systems in real use, other problems - essentially the same problems as any other major system - seem at least as important as acquiring knowledge: integration, access to large databases and performance. Performance can be particularly problematic because early expert systems were built using tools (such as earlier versions of Lisp) that interpreted code expressions without their one compilation. This provided a powerful development environment, but with a disadvantage that it is almost impossible to match the effectiveness of the fastest composed languages (such as C). The integration of systems and databases was difficult for early expert systems because the tools were mainly in languages and platforms that were not familiar and not welcome in most enterprise IT environments - programming languages such as Lisp and Prolog, and hardware platforms such as lisp machines and personal computers. As a result, significant efforts in the later stages of the development of expert system tools were aimed at integrating with outdated environments such as COBOL and database systems, as well as those that are transferred to more standard platforms. These issues were resolved mainly by changing the paradigm of the client server, as PCs were gradually adopted in the IT environment as a legitimate platform for serious business system development and as an affordable servers provided the processing power needed for AI applications. Another major problem of expert systems arises when the size of the knowledge base increases. This increases the complexity of the processing. For example, when an expert system with 100 million rules was conceived as the ultimate expert system, it became apparent that such a system would be too complex and would face too many computational problems. The output engine must be able to handle a huge number of rules to make a decision. How to check that the rules of decision making are consistent with each other is also a problem when there are too many rules. Usually such a problem results in satisfied (SAT) wording. This is a well-known problem of NP-full bolian satiety. Assuming only binary variables, say n of them, and then the appropriate search space is a size 2 n .n. So the search space can grow exponentially. There are also questions about how to prioritize using rules to work more effectively, or how to eliminate ambiguity (e.g. if there are too many others if tweaks under the same rule) and so on. Other problems are related to the effects of over-stress and over-generation when using known facts and attempts to generalize other cases that are clearly not described in the knowledge base. Such problems also exist with methods that use approaches to machine learning. Another problem with the knowledge base is how to update your knowledge quickly and effectively. In addition, adding a new piece of knowledge (i.e. adding it among many rules) is also challenging. Modern approaches, which rely on machine learning methods, are easier in this regard. Because of the above problems, it has become clear that new approaches to AI are needed instead of rules-based technologies. These new approaches are based on machine learning techniques as well as the use of feedback mechanisms. The key challenges faced by expert systems in medicine (if we consider computer diagnostic systems as modern expert systems), and perhaps in other applications, include such issues as: big data, existing rules, health practices, various algorithmic issues and system evaluation. Hayes-Roth apps divide expert system applications into 10 categories illustrated in the following table. An example of the applications was not in the original Hayes-Roth table, and some of them emerged well afterwards. Any app that is not footnoted is described in the Hays-Roth book. In addition, while these categories provide an intuitive basis for describing the space of application of expert systems, they are not rigid and in some cases, the app may feature more than one category. Category categories Addressed Examples interpretation Conclusional descriptions of the situation from the Touch Data Hearsay (Speech Recognition), PROSPECTOR Forecast Finding the likely effects of this situation Assessment of the risk of preterm birth .57 Diagnosis of the withdrawal of the system of faults from observed CADUCEUS, MYCIN, PUFF, Mistral, Mortgage Consultant, R1 (DEC VAX Configuration), SID (DEC VAX 9000 CPU) Planning Action Planning Mission Planning for Autonomous Underwater Vehicle,61 Monitoring Comparison Observations for Reactor Vulnerability Planning '62 Debugging Providing additional solutions to complex solutions to SAINT complex problems, MATHLAB, MACSYMA RepairIng the PAL's prescribed administration plan, Intelligent Clinical Training, STEAMER, interpretation, forecasting, repairing and monitoring real-time process management, Space Shuttle Mission Control was an early attempt to address the problem of voice recognition through an expert systems approach. For the most part, this category of expert systems was not so successful. Hearsay and all interpretation systems are essentially image recognition systems, and look for patterns in noisy data. In the case of Hearsay, phoneme recognition is in the audio stream. Other early examples were the analysis of sonar data to detect Russian submarines. These kinds of systems have proven to be much more ineven to solve AI neural networks than the rule-based approach. CADUCEUS and MYCIN were medical diagnostic systems. The user describes their symptoms on the computer as they would to the doctor and the computer returns the medical diagnosis. Dendral was a tool for studying the formation of hypotheses in the identification of organic molecules. The common problem it solved - developing a solution with a set of limitations in mind - was one of the most successful areas for early expert systems used in business areas, such as the settings of VAX's Digital Equipment Corporation (DEC) computer vendors and the development of mortgage applications. Smh. PAL is an expert system for evaluating students with multiple disabilities. Mistral is an expert dam safety monitoring system developed by Ismes in Italy in the 1990s. It receives data from an automatic monitoring system and diagnoses the condition of the dam. Its first copy, installed in 1992 at the Ridracoli Dam (Italy), is still operational 24/7/365. It has been installed on several dams in Italy and abroad (e.g. Itaipu Dam in Brazil), as well as on landslide sites called Aidenet, and on monuments called Kaleidos. Mistral is a registered trademark of CESI. See also AI Winter CLIPS Constraint Logic Programming Limited Knowledge of engineering training training system machine learning based on the Rules of Links - Jackson, Peter (1998). Introduction to expert systems (3 ed.). Addison Wesley. page 2. ISBN 978-0-201-87686-4. Normal programming. Pcmag.com. Received 2013-09-15. a b Leondes, Cornelius T. (2002). Expert systems: knowledge management and decision-making technology in the 21st century. 1-22. ISBN 978-0-12-443880-4. a b Russell, Stuart; Norwig, Peter (1995). Artificial intelligence: a modern approach (PDF). Simon Schuster. 22-23. ISBN 978-0-13-103805-9. Archive from the original (PDF) dated May 5, 2014. Received on June 14, 2014. a b Luger and Stubbfield 2004, page 227-331. sfn error: no goal: CITEREFLugerStubblefield2004 (help) - b Nilsson 1998, chpt. 17.4. sfn error: no goal: CITEREFNilsson1998 (help) - b McCorduck 2004, page 327-335, 434-435. sfn error: no goal: CITEREFMcCorduck2004 (help) - b Crevier 1993, page 145-62, 197-203. sfn error: no goal: CITEREFCrevier1993 (help) - b d Yanase J, Triantaphyllou E (2019). System examination of computer diagnostics in medicine: past and present developments. Expert systems with applications. 138: 112821. doi:10.1016/j.eswa.2019.112821. - Ledley RS, and Lusted LB (1959). The basics of medical diagnosis. Science. 130 (3366): 9–21. Bibkod:1959Sci... 130....9L. doi:10.1126/science.130.3366.9. PMID 13668531. Weiss S.M., Kulikovsky CA, Amarel S, Saphir A (1978). A model method for making medical decisions using a computer. Artificial intelligence. 11 (1–2): 145–172. doi:10.1016/0004-3702(78)90015-2. Schwartz WB (1970). Medicine and computer: the perspectives and challenges of change. New England Journal of Medicine. 283 (23): 1257–1264. doi:10.1056/NEJM197012032832305. PMID 4920342. Blaich HL (1972). Computer consultation: Electrolyte and acid-based disorders. American Journal of Medicine. 53 (3): 285–291. doi:10.1016/0002-9343(72)90170-2. PMID 4559984. Rosati RA, McNir JF, Starmer CF, Mittler BS, Morris JJ, and Wallace AG (1975). A new information system for medical practice. Archives of Internal Medicine. 135 (8): 1017–1024. doi:10.1001/archinte.1975.00330080019003. PMID 1156062. Gorry GA, Kassirer JP, Essig A, and Schwartz WB (1973). Analysis of solutions as a basis for computer management of acute renal failure. American Journal of Medicine. 55 (4): 473–484. doi:10.1016/0002-9343(73)90204-0. PMID 4582702. - Szolovits P, Patil RS and Schwartz WB (1988). Artificial intelligence in medical diagnostics. Annals of Internal Medicine. 108 (1): 80–87. doi:10.7326/0003-4819-108-1-80. PMID 3276267. Shortliff EH and Buchanan BG (1975). A model of non-actual reasoning in medicine. Mathematical biosciences. 23 (3–4): 351–379. doi:10.1016/0025-5564(75)90047-4. Miller RA, Pople Jr. he, and Myers JD (1982). Internist I, computer diagnostic consultant for general internal medicine. New England Journal of Medicine. 307 (8): 468–476. doi:10.1056/NEJM198208193070803. PMID 7048091. Feigenbaum, Edward; McCorduk, Pamela (1984). Fifth generation. Addison-Wesley. 1-275. ISBN 978-0451152640. The question kenyon.edu: AI Timeline, received October 27, 2018 - Edward Feigenbaum, 1977 Paraphrased by Hayes Roth, et al. Waterman, Donald; Lenat, Douglas (1983). Construction of expert systems. Addison-Wesley. 6-7. ISBN 978-0-201-10686-2. George F. Luger and William A. Stubblefield, Benjamin/Cummings Publishers, Rule Based on Shell's Expert System: An Example of Code Using the Prolog Rule based on the expert shell A system. Michiels, University of Liege, Belgique: PROLOG, first declarative language - Durkin, J. Expert Systems: Application Catalog. Intelligent Computer Systems, Inc., Akron, Ohio, 1993. Orfali, Robert (1996). A basic guide to customer/server survival. New York: Wiley Computer Publishing. 1-10. ISBN 978-0-471-15325-2. Hurwitz, Judith (2011). Smart or happy: How technology leaders turn chance into success. John Wylie and son. page 164. ISBN 978-1118033784. Received on November 29, 2013. Robert J. Dunn (September 30, 1985). Expandable expertise for everyday users. Infomir. 7 (39): 30. Received 2011-03-13. Carl S. Gibson, et al., VAX 9000 SERIES, Digital Technical Journal Digital Equipment Corporation, Volume 2, Number 4, Fall 1990, pp118-129. Richard M. Karp (1972). Reducation among combinatorial problems (PDF). In R. E. Miller; J.W. Thatcher(85-103. Ai Expert Newsletter: W for winter. Archive from the original 2013-11-09. Received 2013-11-29. Leith., Rise and Fall of the Legal Expert System, in the European Journal of Law and Technology, Volume 1, Issue 1, 2010 - Haskin, David (January 16, 2003). Years after the hype, expert systems pay for some. The date. Received on November 29, 2013. - SAP Information Desk. SAP News Desk IntelliCorp announces its participation in SAP EcoHub. laszlo.sys-con.com LaszlotTrec. Received on November 29, 2013. Pegasystems. Smart BPM requires intelligent business rules. pega.com. received on November 29, 2013. Ky. Ying, Shi; Chang, Linlin; Hu, Luokai (October 9-10, 2010). Achieving the integration of business processes and business rules using SPL. The future of Information Technology and Engineering Management (FITME). 2. Changzhou, China: IEEE. 329-332. doi:10.1109/fitme.2010.5656297. ISBN 978-1-4244-9087-5. David C. England (June 1990). Expert hazardous materials management system at the Naval Supply Centre (PDF) (Master's Thesis). Monterey Naval Graduate School/CA. Here: page 21. Smith, Reed (May 8, 1985). Concepts of systems based on knowledge, methods, examples (PDF). Reed G. Received on November 9, 2013. a b Hayes-Roth, Frederick; Donald Waterman; Lenat, Douglas (1983). Construction of expert systems. Addison-Wesley. ISBN 978-0-201-10686-2. Nabil Armand, Palestinian Polytechnic University, January 2007, detection of faults in dynamic rule bases using tree coverings and disjoin Sets: - Mettree, William (1987). Evaluating tools to build large knowledge-based systems. AI magazine. 8 (4). Archive from the original 2013-11-10. Received 2013-11-29. Robert McGregor (June 1991). Use a description classifier to enhance knowledge perception. IEEE expert. 6 (3): 41–46. doi:10.1109/64.87683. S2CID 29575443. Berners-Lee, Tim; James Hendler; Lassila, Ora (May 17, 2001). A semantic web-based new form of web content that matters to computers will revolutionize new features. A scientific American. 284 (5): 34–43. doi:10.1038/scientificamerican0501-34. Archive from the original on April 24, 2013. Hayes Roth, Frederick; Donald Waterman; Lenat, Douglas (1983). Construction of expert systems. Addison-Wesley. page 6. ISBN 978-0-201-10686-2. b Wong, Bo K.; Monaco, John A.; Monaco (September 1995). Expert system applications in business: review and analysis of literature. Information and management. 29 (3): 141–152. doi:10.1016/0378-7206 (95)00023-p. Received November 29, 2013. Kendal, S.L.; Crean, M. (2007). Introduction to knowledge engineering. London: Springer. ISBN 978-1-84628-475-5. OCLC 70987401. Edward A. Feigenbaum; McCormack, Pamela (1983). Fifth generation (1st place). Reading, Massachusetts: Addison-Wesley. ISBN 978-0-201-11519-2. OCLC 9324691. Douglas B. Lenat (1992). On the cusp of knowledge. In David Kirsch. MIT Press. 185-250. - Bezem M (1988). A sequence of rules-based expert systems. At the International Conference on Automated Deduction. Lecture notes in computer science. 310: 151–161. doi:10.1007/BFb0012830. ISBN 3-540-19343-X. Users' participation in updating the knowledge of expert systems. Information and management. 32 (2): 55–63. doi:10.1016/S0378-7206(96)00010-9. Pham HN, Triantaphyllou E (2008). The effect of overexertion and over-generation on the accuracy of classification in data mining. Soft computing for knowledge detection and data analysis: 391-431. Pham HN, Triantaphyllou E (2008). Predicting diabetes by using a new approach to data collection that balances installation and generalization. Computer and Inf. Science G: 11-26. Shan N, and Siarco W (1995). Acquisition based on data and a gradual change in classification rules. Computational intelligence. 11 (2): 357–370. doi:10.1111/j.1467-8640.1995.tb00038.x. PC Coat (1988). Why expert systems fail. Financial management. 17 (3): 77–86. doi:10.2307/3666074. JSTOR 3666074. ^ PH, and Vriens DJ (1999). Knowledge-based systems and knowledge management: friends or enemies?. Information and management. 35 (2): 113–125. doi:10.1016/S0378-7206(98)00080-9. Janase J, Triantafilla E (2019). Seven key challenges for the future of computer diagnostics in medicine. In the Journal of Medical Informatics. 129: 413–422. doi:10.1016/j.ijmedinf.2019.06.017. PMID 31445285. Vulery, L.C.; Gyimala-Busse, J (1994). Machine training for an expert system to predict the risk of preterm birth. In the Journal of the American Association for Medical Informatics. 1 (6): 439–446. doi:10.1136/jamia.1994.95153433. PMC 116227. PMID 7850569. a b Salvaneschi, Paolo; Kadei, Mauro; Lazzari, Marco (1996). Application of AI to structural monitoring and safety assessment. IEEE expert. 11 (4): 24–34. doi:10.1109/64.511774. Received on March 5, 2014. a b Lazzari, Marco; Salvaneschi, Paolo (1999). Incorporating a geographic information system into the landslide risk monitoring support system (PDF). International Journal of Natural Hazards. 20 (2–3): 185–195. doi:10.1023/A:1008187024768. S2CID 1746570. a b Lancini, Stefano; Lazzari, Marco; Alberto Masera; Salvaneschi, Paolo (1997). Diagnosis of ancient monuments with expert software (PDF). International Organization for Structural Engineering. 7 (4): 288–291. doi:10.2749/101686697780494392. Kwak, S. H. (1990). Expert flight planning system for autonomous submersibles. Materials of the 1990 symposium on autonomous underwater technology: 123-128. doi:10.1109/AUV.1990.110446. S2CID 60476847. Nelson, W.R. (1982). REACTOR: Expert system for diagnosing and treating nuclear reactors. The citation of the magazine requires magazine (help) - b Hofmeister, Alan (1994). Smh. PAL: an expert system for identifying treatments for students with severe disabilities. Exceptional children. 61 (2). Archive from the original on December 3, 2013. Received on November 30, 2013. - Hiddawi, P; Suebnukarn, S. (2010). Intelligent clinical learning systems. Inf Med methods. 49 (4): 388-9. CiteSeerX 10.1.1.172.60. doi:10.1055/s-0038-1625342. PMID 20686730. Hollan, J.; Hutchins, E.; Weitzman, L. (1984). STEAMER: An interactive simulation-based learning system. AI magazine. Stanley, G.M. (July 15-17, 1991). Experience using knowledge based on real-time process management (PDF) reasoning. A plenary document presented at the International Federation of Automatic Control (IFAC) symposium on R's computational design in control systems. Received on December 3, 2013. Arthur Rasmussen; John F. Muratore; Heindel, Troy A. (February 1990). INCO Expert System Project: CLIPS in Shuttle Flight Control. NTRS. Received on November 30, 2013. External Links Artificial Intelligence on Curlie Expert System Textbook On Code extracted from expert system architecture in ai. expert system architecture based on non production. expert system architecture pdf. expert system architecture in artificial intelligence pdf. expert system architecture ques10. expert system architecture geeksforgeeks. expert system architecture in hindi. expert system architecture in artificial intelligence ppt