REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
30-10-2018

2. REPORT TYPE
Technical Report

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE

Market Analysis Report on Potential Resources for Knowledge-Based Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Nehemiah Liu, Clinical Decision Support & Automation Research Branch, U.S. Army Institute of Surgical Research

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

U.S. Army Institute of
Surgical Research
3698 Chambers Pass
JBSA Fort Sam Houston, TX
78234-6315

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Medical Research
and Materiel Command
Fort Detrick, Maryland 21702-
5012

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
An expert system is a knowledge-based system, with the only difference between the two being in nomenclature. While an expert system describes an artificially intelligent system capable of making decisions commensurate with those made by a human expert, a knowledge-based system describes an architecture consisting of a knowledge base, user interface, and inference engine. This analysis showed that there were many potential candidate resources for developing a knowledge-based expert system. However, due to government restrictions on certain Internet sites, this study was not able to examine all characteristics of potential resources. An ideal candidate resource should have the following strengths: it should be easily accessible, have extensive documentation, possess speed, employ known algorithms, allow for forward/backward chaining of rules, be easy to understand, and be easy to implement on a personal computer and on a mobile platform. Many of the Java-based and Python-based resources met these criteria. Recommended resources include Drools, Jess, and Hammurapi Rules for Java and Pyke for Python.

15. SUBJECT TERMS
Medical Knowledge Bases, Expert Systems

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			USAMRMC
U	U	U	UU	40	**19b. TELEPHONE NUMBER** *(include area code)*

Market Analysis Report on Potential Resources for Knowledge-Based Systems

Prepared For:

U.S. Army Medical Research and Materiel Command
Telemedicine and Advanced Technology Research Center
Medical Intelligent Systems

U.S. Army Institute of Surgical Research
Clinical Decision Support & Automation Research Branch

Prepared By

Nehemiah Liu
Clinical Decision Support & Automation Research Branch
U.S. Army Institute of Surgical Research
3698 Chambers Pass
JBSA Fort Sam Houston, TX 78234-6315

This report has been created to provide a market analysis and comparison of potential resources for developing a medical knowledge base, as well as recommendations.

Table of Contents

Summary

The following market analysis was conducted to identify and compare potential resources for developing a knowledge-based expert system. Such resources will help facilitate the successful completion and final delivery of a project titled *AI-Assist for Combat Medic*, a joint effort between the Medical Intelligent Systems group of the US Army Medical Research and Materiel Command, Telemedicine and Advanced Technology Research Center, and the Clinical Decision Support & Automation Research Branch of the US Army Institute of Surgical Research.

The purpose of the *AI-Assist for Combat Medic* project is to 1) update/expand current Joint Tactical Combat Casualty Care (JTCCC) guidelines for application to Prolonged Field Care (PFC) environments (e.g., incorporate continuous monitoring and treatment during PFC) and 2) represent these guidelines using a knowledge-based framework that can be integrated with existing algorithms, decision support, and machine learning (ML) / artificial intelligence (AI) in order to form an expert system. The eventual goals are to develop and validate this knowledge-based system and accompanying technologies and transition them to a mobile environment for use on an Android platform, akin to the Army Nett Warrior or Special Operations Command (SOCOM) Android Tactical Assault Kit (ATAK) platform. In short, the *AI-Assist for Combat Medic* project will expand current JTCCC guidelines for PFC scenarios and research the design of a knowledge base encapsulating these guidelines to facilitate development of a mobile decision support system that will enhance decision-making and combat casualty care at the point of injury during prolonged settings.

A key component of the *AI-Assist for Combat Medic* project is the development of a medical knowledge base (and compatible system). A knowledge-based system is an expert system, with the only difference between the two being in nomenclature. While an expert system describes an artificially intelligent system capable of making decisions commensurate with those made by a human expert, a knowledge-based system describes an architecture consisting of a knowledge base, user interface, and inference engine. Thus, a major challenge of this project is to select a feasible resource in order to represent the JTCCC guidelines (applied to PFC environments) as knowledge explicitly. In other words, the challenge is to develop the knowledge base for an expert system using an object-oriented paradigm, ontologies, or rules. Other challenges to consider in light of this project area are as follows: how to implement the knowledge-based framework using a specific knowledge-based technology or rule-based language and which algorithms, data elements, novel indicators of patient assessment, and/or ML/AI techniques to integrate with the knowledge base.

Objectives

Based on current online search engines and tools, this study provided information on potential resources for developing a knowledge-based expert system. Additionally, information was collected on specific examples. As such, this report provided helpful information for future research, development, and implementation purposes.

The following objectives were met through the research:

1. Identify potential resources for developing a knowledge-based expert system

2. Compare these potential resources according to details, advantages and disadvantages (limitations)

3. Provide recommendations for the *AI-Assist for Combat Medic* project

Potential resources were evaluated based on the following characteristics and details: available documentation, type of algorithms used, language, syntax, environment, speed, forward/backward chaining of rules, and ease of implementation on a personal computer and on a mobile platform. Furthermore, potential resources were examined based on advantages and limitations.

Potential Resources for Developing Expert Systems (C, MLM, PHP, Python, XML)

• ARDEN
- HL7 International standard
- ANSI-approved markup language
- Can encode/represent medical knowledge

• CLIPS
- Widely used shell environment
- Based on the C language
- Rule-based or object-based

• d3web
- Open-source Java-based XML framework
- Runs knowledge bases, including rules, decision trees, flow charts, cost-benefit dialog strategies, time-based reasoning, etc.

• DEX
- Qualitative multi-attribute decision modeling methodology
- Integrates multi-criteria decision modeling with rule-based expert systems

• OpenCyc
- Open-source version of the Cyc technology (the world's largest and most complete general knowledge base and common sense reasoning engine)

• OpenExpert
- PHP expert system tool
- Mainly focus on applications for legal expert systems

• Protégé
- Free, open-source ontology editor and framework
- Leading ontological engineering tool

• Pyke
- Python knowledge engine
- Employs logic programming to develop expert systems in Python

Potential Resources for Developing Expert Systems (Java)

• Algernon
- Efficient and concise knowledge base traversal and retrieval
- Straightforward access to ontology classes and instances
- Supports both forward and backward chaining

• Blaze Advisor
- Used by Sun Microsystems
- Can verify the rules and procedural logic

• Drools
- Open source Java-based framework
- Employs the Rete, ReteOO, and PHREAK algorithms
- Allows for fast and reliable evaluation of rules and complex event processing
- Object-Oriented
- Declarative logic programming
- Flexible
- High performance execution

• DTRules
- Rules engine in Java using decision tables

• Esper
- Enables rapid development that process large volumes of incoming messages or events
- Supports event stream processing: time-based, interval-based, length-based and sorted windows; grouping, aggregation, sorting, filtering and merging of event streams; SQL-like query language using insert into, select, from, where, group-by, having and order-by clauses; Inner-joins and outer joins (left, right, full)
- Filters and analyzes events in various ways, and responds to conditions of interest in real-time

• Euler
- A backward-chaining reasoner enhanced with Euler path detection

• Hammurapi Rules
- Simplifies complex problems into smaller steps
- No need to learn specialized rules language
- Rules and facts are written in Java

- Same rules and rulesets can be applied or forward or backward chaining

• InfoSapient
- Semantics of business rules expressed

• Jamocha
- Rule engine and expert system shell environment
- Comes with an FIPA-compliant agent
- The agent is based on the Multiagent System JADE that supports speech-acts and the FIPA agent interaction protocol.

• JCHR
- An embedding of Constraint Handling Rules (CHR) in Java
- The multi-paradigmatic integration of declarative, forward chaining CHR rules and constraint (logic) programming within the imperative, OO host language
- High performance is achieved through an optimized compilation to Java code
- Suited for the high-level development of expert systems, incremental constraint solvers and constraint-based algorithms

• jColibri
- Implements case-based reasoning

• Jena
- Includes a rule-based inference engine, an ontology application programming interface, and a query engine

• Jena 2
- Java framework for writing semantic web applications
- It has a reasoner subsystem which includes a generic rule based inference engine together with configured rule sets
- The subsystem is designed to be extensible so that it should be possible to plug a range of external reasoners into Jena

• JEOPS
- A forward chaining rule engine
- Can define rules in Java application servers, client apps and servlets
- Declarative programming
- Employs forward-chaining, first-order production rules

• JESS

- Fastest
- Scripting environment written in Java
- Can manipulate and reason Java objects
- Alternative rules engine and scripting environment to CLIPS

• JLisa
- CLIPS-like rule engine with a common LISP interface in Java
- Powerful framework for building business rules accessible to Java
- JLisa is more powerful than CLIPS because it has the expanded benefit of having all the features from common LISP available.

• JLog
- An implementation of a Prolog interpreter, written in Java
- BSF-compatible language
- It includes built-in source editor, query panels, online help, animation primitives, and a GUI debugger

• JRuleEngine
- Open source
- Based on Java specification
- Rules can be loaded by an XML file or JRuleEngine APIs, so rules can be stored externally into a database
- Its distribution consists of a JAR library, source, examples and javadoc

• JShop2
- Simple Hierarchical Ordered Planner (SHOP) written in Java
- Domain-independent automated-planning systems
- Based on ordered task decomposition, which is a type of Hierarchical Task Network (HTN) planning
- Uses a new "planner compilation" technique to achieve faster execution speed

• JSL (Java Search Library)
- Library written in Java
- Provides a framework for general searching on graphs
- Standard search algorithms depth-first, breadth-first and A* are provided

• JTP (Java Theorem Prover)
- Based on a very simple and general reasoning architecture
- Easy to extend the system by adding new reasoning modules (reasoners), or by customizing or rearranging existing ones

• JxBRE
- Light-weight BRE (Business Rules Engine) for controlling the process flow for an application
- Uses XML to control the application

• Mandarax
- Based on backward reasoning
- Easy integration of all kinds of data sources
E.g., database records can be easily integrated as sets of facts

• MINS Reasoner
- Mins is not Silri
- A reasoner for Datalog programs with negation and function symbols
- supports the Well-Founded Semantics

• mProlog
- Sub-product of the 3APL-M project
- It delivers a reduced Prolog engine, optimized for J2ME applications
- The 3APL-M project is a platform for building applications using Artificial Autonomous Agents Programming Language (3APL) as the enabling logic for the deliberation cycles and internal knowledge representation

• OFBiz Rule Engine
- Business rules engine
- Backward chaining is supported

• OpenLexicon
- Business rules and business process management tool
- Rapidly develops applications for transaction and process-based applications
- 2 main components of Lexicon: the metadata repository and the business rules engine
- Can evaluate small, in-line Java expressions

• OpenL Tablets
- Create decision tables in Excel and use them in Java application in a convenient type-safe manner
- Use data tables in Excel for data setup and testing
- Eclipse plugin controls validity of Excel tables

• Open Rules
- Less expensive and Easier to develop

- Full-scale, open source business rules management framework
- Create, deploy, execute, and maintain decision services
- Can meet the increasing demand for complex rule bases and high-transaction volumes
- User-friendly Rules Administrator that utilizes the power of MS Excel and Eclipse

• Pellet OWL Reasoner
- Open-source Java based OWL Description Logics reasoner
- It can be used in conjunction with either Jena or OWL API libraries
- provides functionalities to see the species validation, check consistency of ontologies, classify the taxonomy, check entailments and answer a subset of RDQL queries
- Based on the tableaux algorithms developed for expressive Description Logics

• Prova
- It extends Mandarax by providing a proper language syntax, native syntax integration with Java, and agent messaging and reaction rules
- The language is used as a rules-based backbone for distributed web applications in biomedical data integration

• RDFExpert
- Uses the Jena API and parser
- A simple expert system shell that uses RDF for all of its input: knowledge base, inference rules and elements of the resolution strategy employed
- Supports forward and backward chaining

• SweetRules
- Integrated toolkit for semantic web rules
- Based on RuleML (Rule Markup Language)
- Supports logic programs extension of RuleML
- Can prioritize conflict handling and procedural attachments for actions and tests

• Termware
- Term Processing System
- Better cost effective
- Comes with Java debug interface
- Includes computational algebra systems, formal models analysis and transformation

• TyRuBa
- Supports higher order logic programming: variables and compound terms are allowed everywhere in queries and rules
- Speeds up execution by making specialized copies of the rule-base

- Builds an index for fast access to rules and facts in the rule base
- The indexing techniques works also for higher-order logic
- TyRuBa does 'tabling' of query results

• Zilonis
- An extremely efficient, multithreaded rules engine based on a variation of the forward chaining Rete algorithm
- Can define a scope for a user or group of users with inheritance of rules between them
- The rules language is similar to CLIPS

Arbitrarily Selected Resources with Details and Examples

Five potential resources were selected arbitrarily for in-depth details and examples.

• ARDEN: an HL7 International standard, as well as an ANSI-approved markup language for encoding/representing medical knowledge

• CLIPS: a widely used shell environment based on the C language for developing knowledge-based (rule-based or object-based) expert systems

• d3web: an open-source Java-based XML framework to run knowledge bases, including rules, decision trees, flow charts, cost-benefit dialog strategies, time-based reasoning, etc.

• Drools: an open source Java-based framework for building and managing business rules in order to emulate the decision-making ability of human experts

• JESS: an alternative rules engine and scripting environment to CLIPS based on the Java language and has the same syntax as Java.

ARDEN

ARDEN or the HL7 International Arden Syntax for Medical Logic Modules (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=2).

ARDEN (also known as Arden syntax) is an HL7 International standard, as well as an ANSI-approved markup language for encoding/representing medical knowledge. It can be used to develop clinical rules and logic for clinical decision support systems in order to generate alerts and messages to providers. Arden syntax comprises of rule sets, called Medical Logic Modules (MLMs), which are processed by a program, called a monitor.

The open source compiler Arden2ByteCode translates MLMs to Java bytecode that is executable on the Java Virtual Machine. The accompanying plugin for Eclipse called Arden4Eclipse supports MLM developers with the power of a full-fledged integrated development environment. (https://en.wikipedia.org/wiki/Arden_syntax)

Advantages
1) HL7 International standard
2) Practical for encoding specific medical concepts
3) Practical for implementing clinical decision support systems
4) Supports time stamping of data elements and events in MLMs
5) Syntax similar to natural language and easily readable
6) Easily handles patient data created at different times by two components
7) MLMs easily documentable, annotatable, and manageable

Limitations
1) Syntax involves curly braces
2) Compilation issues
3) Inconsistencies when using various operators and statements simultaneously
4) Not easily portable to a mobile platform

The following MLM calculates the body mass index (BMI):

```
maintenance:
 title:       body mass index;;
 mlmname:     BMI_complex;;
 arden:       Version 2.5;;
 version:     1.00;;
 institution: Medexter Healthcare;;
 author:      Karsten Fehre;;
 specialist:  Harald Mandl;;
```

```
 date:         2013-11-11;;
 validation:  testing;;
library:
 purpose:     body mass index;;
 explanation: calculation of body mass index
              input: compound list with:
                (number) size in m,
                (number) weight in kg,
                (time)   birth date.
              output: compound list with:
                If the age is not less then 19 and the classification
                wrt.WHO is not normal, a message containing
                the BMI  and the classification will be returned.
              The classification follows the definition by the WHO, 2008.
              The interpretation follows the BMI definition. For non adults
              (age < 19) the definition by Kromeyer-Hauschild is used with
the
              3. and the 97. percentile.
 ;;
 keywords:    BMI, body mass index;;
 citations:   ;;
 links:       http://de.wikipedia.org/wiki/Body-Mass-Index;;
knowledge:
 type:               data_driven;;
 data:
   (size, weight, birth) := argument;           // input of MLM
   bmiEvent := EVENT {bmiEvent};
 ;;
 priority:    ;;
 evoke:
   bmiEvent;;
 logic:

   // calculation of BMI

   let bmi be weight / (size ** 2);             // BMI
   age := currenttime - birth;                  // age

   // classification wrt. WHO (only for adults)
   if     age < 19 years then classification := null;
   elseif bmi < 16       then classification := localized 'strongunder';
   elseif bmi < 17       then classification := localized 'modunder';
   elseif bmi < 18,5     then classification := localized 'slightunder';
   elseif bmi < 25       then classification := null;
   elseif bmi < 30       then classification := localized 'obese';
   elseif bmi < 35       then classification := localized 'obeseI';
   elseif bmi < 40       then classification := localized 'obeseII';
   else                       classification := localized 'obeseIII';
   endif;
```

```
   bmi := bmi formatted with localized 'msg';  // construct the localized
message

   conclude classification is present ;        // if there is a
classification, execute the action slot
 ;;
 action:
   return bmi || classification || ".";         // return result
 ;;
 urgency:     ;;

resources:
 default: de;;
 language: en
   'msg'        : "The patient's BMI %.1f is not in the normal range and is
classified as ";
   'strongunder': "severe thinness";
   'modunder'   : "moderate thinness";
   'slightunder': "mild thinness";
   'obese'      : "pre-obese";
   'obeseI'     : "obese class I";
   'obeseII'    : "obese class II";
   'obeseIII'   : "obese class III"
 ;;
 language: de
   'msg'        : "Der BMI %.1f des Patienten ist nicht im normalen Bereich
und wird klassifiziert als ";
   'strongunder': "starkes Untergewicht";
   'modunder'   : "mäßiges Untergewicht";
   'slightunder': "leichtes Untergewicht";
   'obese'      : "Präadipositas (Übergewicht)";
   'obeseI'     : "Adipositas Grad I";
   'obeseII'    : "Adipositas Grad II";
   'obeseIII'   : "Adipositas Grad III"
 ;;
end:
```

The following MLM checks the diastolic blood pressure of a patient:

```
maintenance:
 title: To check the diastolic blood pressure of the patient;;
 mlmname: Hypotension;;
 arden: version 2.7;;
 version: 1.00;;
 institution: Latrobe University Bundoora;;
 author: Lakshmi Devineni;;
 specialist: ;;
 date: 2013-06-02;;
 validation: testing;;
```

```
library:
 purpose: check if the diastolic blood pressure of the patient is within
limits;;
 explanation: This MLM is an example for reading data and writing a message;;
 keywords: hypotension; categorization;;
 citations: ;;
 links: http://en.wikipedia.org/wiki/Hypotension;;
knowledge:
 type: data_driven;;
 data:
 /* read the diastolic blood pressure */
 diastolic_blood_pressure := read last
 {diastolic blood pressure}; /* the value in braces is specific to your
 runtime environment */
 /* If the height is lower than height_threshold, output a message */
 diastolic_pressure_threshold := 60;
 stdout_dest := destination
 {stdout};
 ;;
 evoke: null_event;;
 logic:
 if (diastolic_blood_pressure is not number) then
 conclude false;
 endif;
 if (diastolic_blood_pressure >= diastolic_pressure_threshold) then
 conclude true;
 else
 conclude false;
 endif;
 ;;
 action:
 write "Your Diastolic Blood Pressure is too low (hypotension)"
 at stdout_dest;
 ;;
resources:
 default: de
 ;;
 language: en
 'msg' : "The normal range from 60 to 90";
 ;;
 language: de
 'msg' : "Der Normalbereich von 60 bis 90";
 ;;
end:
```

CLIPS

CLIPS or C Language Integrated Production System
(http://www.clipsrules.net/, http://clipsrules.sourceforge.net/)

CLIPS is a widely used shell environment based on the C language for developing knowledge-based (rule-based or object-based) expert systems. It incorporates a complete object-oriented language for writing expert systems, as well as a user interface similar to that that of the LISP programming language.

CLIPS employs the Rete algorithm, and its rules engine has already been developed in C. Therefore, CLIPS allows for forward chaining of rules.

Advantages
1) High-level production rule interpreter, with shell environment
2) Syntax similar to LISP
3) Facts and rule-base similar to Prolog
4) Runs on UNIX, Linux, DOS, Windows, and MacOS
5) Extensive documentation
6) Includes object-oriented constructs

Limitations
1) Based on the C language
2) No backward chaining of rules

The following implements a simple advising system:

```
CLIPS>
(deffacts prerequisites
   (after COP1000 take COP2000)
   (after COP1001 take COP2001)
   (after MAC1000 take MAC2000)
   (after MAC1001 take MAC2001)
   (after ENG1000 and ENG1001 take ENG2000)
   (student Cindy COP1000 MAC1001)
   (phase welcome))
CLIPS>
(defrule welcome
   ?f <- (phase welcome)
   =>
   (retract ?f)
   (assert (phase process))
   (printout t "Welcome to the advising system." crlf))
CLIPS>
```

```
(defrule rule2
   (phase process)
   (after ?course1 take ?course2)
   (student ?name $? ?course1 $?)
   =>
   (printout t "Since " ?name " has taken " ?course1 " , I suggest taking " ?course2
"." crlf))
CLIPS> (reset)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (watch rules)
CLIPS> (run)
FIRE    1 welcome: f-7
<== f-7     (phase welcome)
==> f-8     (phase process)
==> Activation 0      rule2: f-8,f-1,f-6
==> Activation 0      rule2: f-8,f-4,f-6
Welcome to the advising system.
FIRE    2 rule2: f-8,f-4,f-6
Since Cindy has taken MAC1001 , I suggest taking MAC2001.
FIRE    3 rule2: f-8,f-1,f-6
Since Cindy has taken COP1000 , I suggest taking COP2000.
CLIPS>
```

d3web

d3web and KnowWE
(http://www.d3web.de/)

d3web is an open-source Java-based XML framework to run knowledge bases, including rules, decision trees, flow charts, cost-benefit dialog strategies, time-based reasoning, etc. It does not provide a way to properly create/author knowledge bases, except through Java code on an application programming interface level. On the other hand, KnowWE (Knowledge Wiki Environment) is a semantic web interface built on JSPWiki that allows for knowledge bases and ontologies to be created, tested, debugged, and executed through multiple users and supports control versioning.

https://en.wikipedia.org/wiki/D3web
Some applications (both, commercial and free) created using the d3web diagnostic engine are

- SmartCare™ (a medical closed-loop system for weaning ventilated patients, created by Dräger)
- SonoConsult (a medical support system for evaluating sonographic examinations
- eDOC (a web-based system for self-diagnosing various medical issues)

Advantages
1) Based on Java and XML
2) Supports ontology development
3) Extensive documentation
4) Commercial applications exist

Limitations
1) Difficult to access documentation and tools
2) Not easily portable to a mobile platform

Drools

JBoss Drools
(http://www.drools.org/learn/documentation.html)

Drools is an open source Java-based framework for building and managing business rules in order to emulate the decision-making ability of human experts. It involves a forward-chaining and backward-chaining inference-based rules engine that employs the Rete, ReteOO, and PHREAK algorithms and allows for fast and reliable evaluation of business rules and complex event processing.

Drools belongs to the JBoss KIE (Knowledge is Everything) portfolio that allows for agile development and includes a web application and web-based workbench. It supports an XML paradigm.

Advantages
1) High-level production rule interpreter, with Eclipse environment
2) Runs on UNIX, Linux, DOS, Windows, and MacOS
3) Extensive documentation
4) Includes Java classes
5) Forward/backward chaining
6) Extends Rete algorithm

Limitations
1) Follows a JBoss programming paradigm

The following example demonstrates state and salience in Drools:

Salience State: Rule "A to B"
```
rule "A to B"
    when
        State(name == "A", state == State.FINISHED )
        b : State(name == "B", state == State.NOTRUN )
    then
        System.out.println(b.getName() + " finished" );
        b.setState( State.FINISHED );
end
```

Salience State: Rule "B to C"
```
rule "B to C"
        salience 10
    when
        State(name == "B", state == State.FINISHED )
        c : State(name == "C", state == State.NOTRUN )
```

```
    then
        System.out.println(c.getName() + " finished" );
        c.setState( State.FINISHED );
end
```

Salience State: Rule "B to D"

```
rule "B to D"
    when
        State(name == "B", state == State.FINISHED )
        d : State(name == "D", state == State.NOTRUN )
    then
        System.out.println(d.getName() + " finished" );
        d.setState( State.FINISHED );
end
```

Agenda Group State Example: Rule "B to C"

```
rule "B to C"
      agenda-group "B to C"
      auto-focus true
  when
      State(name == "B", state == State.FINISHED )
      c : State(name == "C", state == State.NOTRUN )
  then
      System.out.println(c.getName() + " finished" );
      c.setState( State.FINISHED );
      kcontext.getKnowledgeRuntime().getAgenda().getAgendaGroup( "B to
D" ).setFocus();
end
```

Agenda Group State Example: Rule "B to D"

```
rule "B to D"
      agenda-group "B to D"
  when
      State(name == "B", state == State.FINISHED )
      d : State(name == "D", state == State.NOTRUN )
  then
      System.out.println(d.getName() + " finished" );
      d.setState( State.FINISHED );
end
```

JESS

JESS or Java Expert System Shell
(https://www.jessrules.com/)

JESS is an alternative rules engine and scripting environment to CLIPS based on the Java language and has the same syntax as Java. As such, it utilizes Java classes for writing expert systems, as well as the Eclipse environment.

JESS also employs an advanced version of the Rete algorithm, and its rules engine has already been developed in Java. Therefore, JESS allows for forward/backward chaining of rules, as well as working memory queries.

<u>Advantages</u>
1) High-level production rule interpreter, with Eclipse environment
2) Syntax similar to CLIPS
3) Facts and rule-base similar to CLIPS
4) Runs on all platforms
5) Extensive documentation
6) Includes Java classes

<u>Limitations</u>
1) Request free license from Sandia National Laboratories

The following simple example shows how to use peering. The API consists solely of a special constructor in the Rete class:

```
import jess.*;
public class ExPeering {
    public static void main(String[] argv) throws JessException {
        // Create the "original" engine
        Rete engine = new Rete();

        // Load a rule into it
        engine.eval("(defrule rule-1 (A ?x) => (printout t ?x crlf))");

        // Create a peer of the first engine
        Rete peer = engine.createPeer();

        // Assert different facts into the two engines
        engine.assertString("(A original)");
        peer.assertString("(A peer)");

        // Run the original engine; prints "original"
```

```
        engine.run();

        // Run the peer; prints "peer"
        peer.run();
    }
}
```

Findings at a Glance

This study showed that there were many potential candidate resources for developing a knowledge-based expert system for the *AI-Assist for Combat Medic* project. However, due to government restrictions on certain Internet sites, this study was not able to examine all the characteristics of potential resources.

An ideal candidate resource should have the following strengths: it should be easily accessible, have extensive documentation, possess speed, employ known algorithms, allow for forward/backward chaining of rules, be easy to understand, and be easy to implement on a personal computer and on a mobile platform.

Many of the Java-based and Python-based resources met these criteria.

Recommended resources include Drools, Jess, and Hammurapi Rules for Java and Pyke for Python.

Analysis of Alternatives

EXECUTIVE SUMMARY

In addition to a market analysis above, the following analysis of alternatives was conducted to identify and compare potential resources for developing knowledge-based expert systems.

To reiterate, an expert system is a knowledge-based system, with the only difference between the two being in nomenclature. While an expert system describes an artificially intelligent system capable of making decisions commensurate with those made by a human expert, a knowledge-based system describes an architecture consisting of a knowledge base, user interface, and inference engine.

Again, this analysis showed that there were many potential candidate resources for developing a knowledge-based expert system. However, due to government restrictions on certain Internet sites, this study was not able to examine all characteristics of potential resources.

An ideal candidate resource should have the following strengths: it should be easily accessible, have extensive documentation, possess speed, employ known algorithms, allow for forward/backward chaining of rules, be easy to understand, and be easy to implement on a personal computer and on a mobile platform.

Many of the Java-based and Python-based resources met these criteria.

Recommended resources include Drools, Jess, and Hammurapi Rules for Java and Pyke for Python.

REVISION SUMMARY

Author	Revision	Date	Comments
Nehemiah Liu	1.0	31 Oct 2018	Original draft

SECTION 1: INTRODUCTION

1.1 Background

Currently, there are known capability gaps in civilian and especially military prehospital care that need to be addressed in order to deal with unprecedented challenges in the future. One of these gaps is the limited ability to turn prehospital data, be it vital signs, waveforms, or interventions made, into actionable information to reduce mortality and complications. What is often available for advanced care in U.S. trauma systems is not always available to combat casualties on the battlefield. Therefore, new approaches for automation, documentation, processing, and sensing modalities must be incorporated into the next generation of Tactical Combat Casualty Care (TCCC) solutions, especially for Prolonged Field Care (PFC) scenarios. Contrariwise, the most recent Joint TCCC (JTCCC) guidelines do not address PFC challenges. They mainly provide guidance on how to initiate interventions on the scene prior to immediate evacuation, thereby neglecting changes in vital signs, trends, and/or other changes in patient conditions.

The purpose of the AI-Assist for Combat Medic project is to 1) update/expand current JTCCC guidelines for application to PFC environments (e.g., incorporate continuous monitoring and treatment during PFC) and 2) represent these guidelines using a knowledge-based framework that can be integrated with existing algorithms, decision support, and machine learning (ML) / artificial intelligence (AI) in order to form an expert system. The eventual goals are to develop and validate this knowledge-based system and accompanying technologies and transition them to a mobile environment for use on an Android platform, akin to the Army Nett Warrior or Special Operations Command (SOCOM) Android Tactical Assault Kit (ATAK) platform. In short, the AI-Assist for Combat Medic project will expand current JTCCC guidelines for PFC scenarios and research the design of a knowledge base encapsulating these guidelines to facilitate development of a mobile decision support system that will enhance decision-making and combat casualty care at the point of injury during prolonged settings.

A key component of the AI-Assist for Combat Medic project is the development of a medical knowledge base (and compatible system). A knowledge-based system is an expert system, with the only difference between the two being in nomenclature. While an expert system describes an artificially intelligent system capable of making decisions commensurate with those made by a human expert, a knowledge-based system describes an architecture consisting of a knowledge base, user interface, and inference engine. Thus, a major challenge of this project is to select a feasible resource in order to represent the JTCCC guidelines (applied to PFC environments) as knowledge explicitly. In other words, the challenge is to develop the knowledge base for an expert system using an object-oriented paradigm, ontologies, or rules. Other challenges to consider in light of this project area are as follows: how to implement the knowledge-based framework using a specific knowledge-based technology or rule-based language and which algorithms, data elements, novel indicators of patient assessment, and/or ML/AI techniques to integrate with the knowledge base.

1.2 Purpose

The purpose of this document is to present the results of the Analysis of Alternatives and identify potential resources for developing knowledge-based expert systems.

1.3 Scope

Based on current online search engines and tools, this analysis provided information on potential resources for developing knowledge-based expert systems. Additionally, information was collected on specific examples. This analysis provided helpful information for future research, development, and implementation purposes.

1.4 Study Team/Organization

U.S. Army Medical Research and Materiel Command
Telemedicine and Advanced Technology Research Center
Medical Intelligent Systems

U.S. Army Institute of Surgical Research
Clinical Decision Support & Automation Research Branch

SECTION 2: GROUND RULES AND ASSUMPTIONS

2.1 Scenarios

This project and its results are pertinent to a wide group of patients suffering from prehospital trauma in PFC settings, including patients from mass civilian casualty situations and battlefield scenarios. In these settings, resources and expertise may be limited, and thus, the use of JTCCC guidelines deployed as a knowledge-based system (in conjunction with predictive algorithms and artificial intelligence) on a mobile platform could greatly enhance the medic's capabilities to assess/diagnose combat casualties. In proposing this project, we also envision a scalable, adaptable, and interoperable system. Because guidelines need to be changed to adapt to challenges in the future, representation of these guidelines using a knowledge-based expert system will facilitate rapid translation of guidelines into our warfighters' hands. This has also direct military relevance as these systems can sustain or inform providers to be logistically prepared for which and how many resources to use during PFC scenarios. A mobile platform deployed as a knowledge-based system may not only encourage medics to use the system for PFC scenarios but also enhance their capabilities anywhere and anytime during theater.

2.2 Risks

Risk #1: There is a possibility that some of the guidelines expanded for PFC scenarios will not fit the object-oriented paradigm. Risks will be mitigated by representing facts as assertions about data and representing other guidelines using hierarchies of classes and subclasses, as well as relations between classes and behavior. Another resort is to employ independent rules and interactions within the knowledge base itself through the firing of daemons on objects and chaining rules. Lastly, to mitigate risks, classifiers will be employed to deduce the relationships between objects. As a note, continuous information can be discretized or averaged using software. Thus, guidelines incorporating real-time monitoring may be structured for specific time points, rather than for continuous measurements.

Risk #2: There is the possibility that implementation of a knowledge base using one chosen knowledge-based technology will not be successful. However, this is a minimal risk, as most guidelines providing actionable information can be implemented as an if-then rule. Risks will be mitigated by identifying those guidelines that do not fit the knowledge-based paradigm and by representing facts as assertions about data. Also, risks will be mitigated by identifying a list of potential resources for developing knowledge-based expert systems and implementing guidelines in parallel.

2.3 Environment

Because guidelines need to be changed to adapt to challenges in the future, representation of these guidelines using a knowledge-based expert system will facilitate rapid translation of guidelines into our warfighters' hands. This has also direct military relevance as these systems can sustain or inform providers to be logistically prepared for which and how many resources to use during PFC scenarios. A mobile platform deployed as a knowledge-based system may not only encourage medics to use the system for PFC scenarios but also enhance their capabilities anywhere and anytime during theater.

2.4 Assumptions

Major assumption: Potential resources for developing knowledge-based expert systems may be adapted for and integrated into a mobile platform for rapidly changing, austere environments.

2.5 Constraints

Time and budget constraints.

SECTION 3: ALTERNATIVES

3.1 Potential Resources (C, MLM, PHP, Python, XML)

• ARDEN
- HL7 International standard
- ANSI-approved markup language
- Can encode/represent medical knowledge

• CLIPS
- Widely used shell environment
- Based on the C language
- Rule-based or object-based

• d3web
- Open-source Java-based XML framework
- Runs knowledge bases, including rules, decision trees, flow charts, cost-benefit dialog strategies, time-based reasoning, etc.

• DEX
- Qualitative multi-attribute decision modeling methodology
- Integrates multi-criteria decision modeling with rule-based expert systems

• OpenCyc
- Open-source version of the Cyc technology (the world's largest and most complete general knowledge base and common sense reasoning engine)

• OpenExpert
- PHP expert system tool
- Mainly focus on applications for legal expert systems

• Protégé
- Free, open-source ontology editor and framework
- Leading ontological engineering tool

• Pyke
- Python knowledge engine
- Employs logic programming to develop expert systems in Python

3.2 Potential Resources (Java)

I• Algernon
- Efficient and concise knowledge base traversal and retrieval
- Straightforward access to ontology classes and instances
- Supports both forward and backward chaining

• Blaze Advisor
- Used by Sun Microsystems
- Can verify the rules and procedural logic

• Drools
- Open source Java-based framework
- Employs the Rete, ReteOO, and PHREAK algorithms
- Allows for fast and reliable evaluation of rules and complex event processing
- Object-Oriented

- Declarative logic programming
- Flexible
- High performance execution

• DTRules
- Rules engine in Java using decision tables

• Esper
- Enables rapid development that process large volumes of incoming messages or events
- Supports event stream processing: time-based, interval-based, length-based and sorted windows; grouping, aggregation, sorting, filtering and merging of event streams; SQL-like query language using insert into, select, from, where, group-by, having and order-by clauses; Inner-joins and outer joins (left, right, full)
- Filters and analyzes events in various ways, and responds to conditions of interest in real-time

• Euler
- A backward-chaining reasoner enhanced with Euler path detection

• Hammurapi Rules
- Simplifies complex problems into smaller steps
- No need to learn specialized rules language
- Rules and facts are written in Java
- Same rules and rulesets can be applied or forward or backward chaining

• InfoSapient
- Semantics of business rules expressed

• Jamocha
- Rule engine and expert system shell environment
- Comes with an FIPA-compliant agent
- The agent is based on the Multiagent System JADE that supports speech-acts and the FIPA agent interaction protocol.

• JCHR
- An embedding of Constraint Handling Rules (CHR) in Java
- The multi-paradigmatic integration of declarative, forward chaining CHR rules and constraint (logic) programming within the imperative, OO host language
- High performance is achieved through an optimized compilation to Java code
- Suited for the high-level development of expert systems, incremental constraint solvers and constraint-based algorithms

• jColibri
- Implements case-based reasoning

• Jena
- Includes a rule-based inference engine, an ontology application programming interface, and a query engine

• Jena 2
- Java framework for writing semantic web applications
- It has a reasoner subsystem which includes a generic rule based inference engine together with configured rule sets
- The subsystem is designed to be extensible so that it should be possible to plug a range of external reasoners into Jena

• JEOPS
- A forward chaining rule engine
- Can define rules in Java application servers, client apps and servlets
- Declarative programming
- Employs forward-chaining, first-order production rules

• JESS
- Fastest
- Scripting environment written in Java
- Can manipulate and reason Java objects
- Alternative rules engine and scripting environment to CLIPS

• JLisa
- CLIPS-like rule engine with a common LISP interface in Java
- Powerful framework for building business rules accessible to Java
- JLisa is more powerful than CLIPS because it has the expanded benefit of having all the features from common LISP available.

• JLog
- An implementation of a Prolog interpreter, written in Java
- BSF-compatible language
- It includes built-in source editor, query panels, online help, animation primitives, and a GUI debugger

• JRuleEngine
- Open source
- Based on Java specification
- Rules can be loaded by an XML file or JRuleEngine APIs, so rules can be stored externally into a database
- Its distribution consists of a JAR library, source, examples and javadoc

• JShop2
- Simple Hierarchical Ordered Planner (SHOP) written in Java
- Domain-independent automated-planning systems

- Based on ordered task decomposition, which is a type of Hierarchical Task Network (HTN) planning
- Uses a new "planner compilation" technique to achieve faster execution speed

• JSL (Java Search Library)
- Library written in Java
- Provides a framework for general searching on graphs
- Standard search algorithms depth-first, breadth-first and A* are provided

• JTP (Java Theorem Prover)
- Based on a very simple and general reasoning architecture
- Easy to extend the system by adding new reasoning modules (reasoners), or by customizing or rearranging existing ones

• JxBRE
- Light-weight BRE (Business Rules Engine) for controlling the process flow for an application
- Uses XML to control the application

• Mandarax
- Based on backward reasoning
- Easy integration of all kinds of data sources
E.g., database records can be easily integrated as sets of facts

• MINS Reasoner
- Mins is not Silri
- A reasoner for Datalog programs with negation and function symbols
- supports the Well-Founded Semantics

• mProlog
- Sub-product of the 3APL-M project
- It delivers a reduced Prolog engine, optimized for J2ME applications
- The 3APL-M project is a platform for building applications using Artificial Autonomous Agents Programming Language (3APL) as the enabling logic for the deliberation cycles and internal knowledge representation

• OFBiz Rule Engine
- Business rules engine
- Backward chaining is supported

• OpenLexicon
- Business rules and business process management tool
- Rapidly develops applications for transaction and process-based applications
- 2 main components of Lexicon: the metadata repository and the business rules engine

- Can evaluate small, in-line Java expressions

• OpenL Tablets
- Create decision tables in Excel and use them in Java application in a convenient type-safe manner
- Use data tables in Excel for data setup and testing
- Eclipse plugin controls validity of Excel tables

• Open Rules
- Less expensive and Easier to develop
- Full-scale, open source business rules management framework
- Create, deploy, execute, and maintain decision services
- Can meet the increasing demand for complex rule bases and high-transaction volumes
- User-friendly Rules Administrator that utilizes the power of MS Excel and Eclipse

• Pellet OWL Reasoner
- Open-source Java based OWL Description Logics reasoner
- It can be used in conjunction with either Jena or OWL API libraries
- provides functionalities to see the species validation, check consistency of ontologies, classify the taxonomy, check entailments and answer a subset of RDQL queries
- Based on the tableaux algorithms developed for expressive Description Logics

• Prova
- It extends Mandarax by providing a proper language syntax, native syntax integration with Java, and agent messaging and reaction rules
- The language is used as a rules-based backbone for distributed web applications in biomedical data integration

• RDFExpert
- Uses the Jena API and parser
- A simple expert system shell that uses RDF for all of its input: knowledge base, inference rules and elements of the resolution strategy employed
- Supports forward and backward chaining

• SweetRules
- Integrated toolkit for semantic web rules
- Based on RuleML (Rule Markup Language)
- Supports logic programs extension of RuleML
- Can prioritize conflict handling and procedural attachments for actions and tests

• Termware

- Term Processing System
- Better cost effective
- Comes with Java debug interface
- Includes computational algebra systems, formal models analysis and transformation

• TyRuBa
- Supports higher order logic programming: variables and compound terms are allowed everywhere in queries and rules
- Speeds up execution by making specialized copies of the rule-base
- Builds an index for fast access to rules and facts in the rule base
- The indexing techniques works also for higher-order logic
- TyRuBa does 'tabling' of query results

• Zilonis
- An extremely efficient, multithreaded rules engine based on a variation of the forward chaining Rete algorithm
- Can define a scope for a user or group of users with inheritance of rules between them
- The rules language is similar to CLIPS

SECTION 4: DETERMINATION OF EFFECTIVENESS MEASURES

4.1 Mission Tasks

Mission tasks include the representation of JTCCC guidelines as knowledge, that is, knowledge base development, and the implementation of the knowledge-based framework.

4.2 Characteristics of Resources

An ideal candidate resource should have the following strengths: it should be easily accessible, have extensive documentation, possess speed, employ known algorithms, allow for forward/backward chaining of rules, be easy to understand, and be easy to implement on a personal computer and on a mobile platform.

SECTION 5: METHODOLOGY

5.1 Search Engines

This analysis relied on current online search engines and tools for reviewing potential resources on knowledge-based expert systems.

5.2 Matrix Analysis

This analysis compared potential resources on knowledge-based expert systems through a table/matrix of characteristics.

SECTION 6: ANALYSIS RESULTS

Please see the table in the Appendix

SECTION 7: RECOMMENDED ALTERNATIVES AND RATIONALE

This analysis showed that there were many potential candidate resources for developing a knowledge-based expert system. However, due to government restrictions on certain Internet sites, this study was not able to examine all characteristics of potential resources.

An ideal candidate resource should have the following strengths: it should be easily accessible, have extensive documentation, possess speed, employ known algorithms, allow for forward/backward chaining of rules, be easy to understand, and be easy to implement on a personal computer and on a mobile platform.

Many of the Java-based and Python-based resources met these criteria.

Recommended resources include Drools, Jess, and Hammurapi Rules for Java and Pyke for Python.

APPENDIX

(A) Table of Characteristics

(B) References

Appendix

Table of Characteristics

Resource	Language	Syntax	Forward Chaining	Backward Chaining	Algorithm	Use on Mobile?	Speed	Documentation
Algernon	Java		x	x		Yes		
ARDEN	MLM	Hard	x					Good
Blaze Advisor	Java					Yes		
CLIPS	C	Hard	x		Rete		Fast	Excellent
d3web	XML	Medium						Good
DEX	GUI							
Drools	Java	Medium	x	x	Rete+	Yes	Fast	Excellent
DTRules	Java				Tables	Yes		
Esper	Java					Yes		
Euler	Java			x		Yes		
Hammurapi	Java	Medium	x	x	Rete+	Yes	Fast	Excellent
InfoSapient	Java					Yes		
Jamocha	Java				JADE	Yes		
JCHR	Java				CHR	Yes		
jColibri	Java		x		Case-based	Yes		
Jena	Java	Hard	x	x	Rete+	Yes	Fast	Good
Jena 2	Java	Hard	x	x	Rete+	Yes	Fast	
JEOPS	Java		x			Yes		
JESS	Java	Hard	x	x	Rete+	Yes	Very Fast	Excellent
JLisa	Java	Hard	x	x	Rete+	Yes	Fast	Good
JLog	Java					Yes		
JRuleEngine	Java, XML					Yes		
JShop2	Java				SHOP, HTN	Yes		
JSL	Java				Tree	Yes	Fast	Fast
JTP	Java					Yes		
JxBRE	Java, XML					Yes		
Mandarax	Java			x		Yes		Good
MINS Reasoner	Java					Yes		
mProlog	Java				3APL	Yes		
OFBiz	Java			x		Yes		Excellent

Open Rules	Java					Yes		
OpenCyc	Cyc							
OpenExpert	PHP							
OpenLexicon	Java					Yes		
OpenL Tablets	Java, Excel				Tables			
OpenRules	Java, Ecel							
Pellet OWL	Java				OWL DL	Yes		Good
Protégé	GUI							Good
Prova	Java			x		Yes		
Pyke	Python	Hard	x	x		Yes		Excellent
RDFExpert	Java		x	x		Yes		
SweetRules	Java				RuleML	Yes		
Termware	Java					Yes	Simple	
TyRuBa	Java					Yes		Good
Zilonis	Java		x		Rete	Yes	Efficient	Good

References

ARDEN

Hripcsak G. Writing Arden Syntax medical logic modules. Computers in Biology and Medicine 1994;24(5):331-363.

Hripcsak G, Ludemann P, Pryor TA, Wigertz OB, Clayton PD. Rationale for the Arden Syntax. Computers and Biomedical Research 1994;27:291-324.

Jenders RA, Hripcsak G, Sideli RV, DuMouchel W, Zhang H, Cimino JJ, Johnson SB, Sherman EH, Clayton PD. Medical decision support: experience with implementing the Arden Syntax at the Columbia-Presbyterian Medical Center. Proc AMIA Symp 1995;:169-73.

Johansson B, Shahsavar N, Åhlfeldt H. Database and knowledge base integration - a query mapping method for Arden Syntax knowledge modules. Meth Inform Medicine 1996;35:302-308.

Karlsson D, Ekdahl C, Wigertz O, Shahsavar N, Gill H, Forsum U. Extended telemedical consultation using Arden Syntax based decision support, hypertext and WWW technique. Meth Inform Med 1997;36:108-114.

Wigertz O, Hripcsak G, Shahsavar N, Bågenholm P, Åhlfeldt H, Gill H. Data-driven medical knowledge-based systems based on Arden Syntax. In Knowledge and decisions in health telematics. Barahona P and Christensen JP eds. Amsterdam: IOS Press, 1994;126-131.

Clinical Decision Support Systems

Bates DW, Cohen M, Leape LL, Overhage JM, Shabot MM, Sheridan T. Reducing the frequency of errors in medicine using information technology. J Am Med Inform Assoc 2001;8(4):299-308.

Bates DW, Gawande AA. Improving safety with information technology. N Engl J Med 2003;348(25):2526-2534.

Evans RS, Pestotnik SL, Classen DC et al. A computer-assisted management program for antibiotics and other antiinfective agents. N Engl J Med1998;338(4):232-238.

Johnston ME, Langton KB, Haynes RB, Mathieu A. Effects of computer-based clinical decision support systems on clinician performance and patient outcome. A critical appraisal of research. Ann Intern Med 1994;120:135-142.

McDonald CJ, Overhage JM, Tierney WM, Abernathy GR, Dexter PR. The promise of computerized feedback systems for diabetes care. Ann Intern Med 1996;124(1 pt 2):170-174.

Pestonik SL, Classen DC, Evans RS, Burke JP. Implementing antibiotic practice guidelines through computer-assisted decision support: clinical and financial outcomes. Ann Intern Med 1996;124:884-890.

Rind DM, Safran C, Phillips RS, Wang Q, Calkins DR, Delbanco TL, Bleich HL, Slack WV. Effect of computer-based alerts on the treatment and outcomes of hospitalized patients. Arch Intern Med 1994;154:1511-1517.

Shea S, DuMouchel W, Bahamonde L. A meta-analysis of 16 randomized controlled trials to evaluate computer-based clinical reminder systems for preventive care in the ambulatory setting. J Am Med Inform Assoc 1996;3(6):399-409.

Shea S, Sideli RV, DuMouchel W, Pulver G, Arons RR, Clayton PD. Computer-generated informational messages directed to physicians: effect on length of hospital stay. J Am Med Informatics Assoc 1995;2:58-64.

Somkin CP, Hiatt RA, Hurley LB, Gruskin E, Ackerson L, Larson P. The effect of patient and provider reminders on mammography and Papanicolaou smear screening in a large health maintenance organization. Arch Intern Med 1997;157:1658-1664.

Weingarten SR, Riedinger MS, Conner L, Lee TH, Hoffman I, Johnson B, Ellrodt AG. Practice guidelines and reminders to reduce duration of hospital stay for patients with chest pain. An interventional trial. Ann Intern Med 1994;120:257-263.

CLIPS

CLIPS: A Tool for Building Expert Systems. NASA Johnson Space Center. http://www.clipsrules.net/Documentation.html

d3web

Baumeister J. Agile Development of Diagnostic Knowledge Systems. Dissertation. Universit¨at W¨urzburg, 2004.

DEX

Bohanec M, Žnidaršič M, Rajkovič V, Bratko I, Zupan B. DEX Methodology: Three Decades of Qualitative Multi-Attribute ModelingDEX Methodology: Three Decades of Qualitative Multi-Attribute Modeling. Informatica 2013;37:49-54.

Drools

The JBoss Drools Team. Drools Documentation. https://docs.jboss.org/drools/release/7.7.0.Final/drools-docs/html_single/index.html

JESS

Ernest Friedman-Hill. Jess, the Rule Engine for the Java Platform. Sandia National Laboratories. https://www.jessrules.com/jess/

Rete

Forgy CL. Rete: A Fast Algorithm for the Many Pattern/ Many Object Pattern Match Problem. Artificial Intelligence 1982;19:17-37