The Communications Requirements Evaluation & Assessment Prototype (CREAP)

Joseph E. Kasser DSc CEng CM
Stephen C. Cook PhD FIEE FIEAust
Systems Engineering and Evaluation Centre
University of South Australia
School of Electrical and Information Engineering F2-37
Mawson Lakes Campus
South Australia 5095
Telephone: +61 (08) 830 23941, Fax: +61 (08) 830 24723
Email: joseph.kasser@unisa.edu.au

Abstract. Today's requirements engineering processes do not have very effective checks to ensure that system requirements are feasible. This paper describes extending the use of Requirements Engineering tools to assist the writers of requirements to only accept requirements that are feasible using the FBRET approach (Cook et al. 2001). The approach is illustrated by the description of the Communications Requirements Evaluation & Assessment Prototype (CREAP). The CREAP is a prototype software package that is able to integrate military communications domain expertise and requirements engineering practice in order to ensure the feasibility of equipment selected to meet the requirements imposed on communications systems for a rapid force deployment in a design to inventory scenario.

BACKGROUND

Today's requirements engineering processes do not have very effective checks to ensure that system requirements are feasible (Cook et al. 2001). In the main, this is because the feasibility of requirements is governed by domain knowledge (application or technology) which is lacking in the writers of the requirements.

The purpose of the CREAP was to determine if Requirements Engineering (RE) tools could be extended to assist the writers of requirements to only accept requirements that are feasible using the FBRET approach. The Communications Requirements Evaluation & Assessment Prototype (CREAP) project constructed a prototype software package. The CREAP is a tool that is able to demonstrate the integration of military communications domain expertise and requirements engineering practice in order to ensure the feasibility of the equipment selected to meet the requirements imposed on communications systems.

The need for FBRET based tools like the CREAP arose from Defence. In contrast to most businesses, the military spends most of its time in a mode of operations that differs from its prime reason for existence. For example, a manufacturing company concentrates on designing and making products and that is its reason for existence, whereas during peacetime, military personnel are tasked with building the force for the infrequent times that it is used for military operations (its prime purpose). Another reality with military systems is that they need to be adaptable to a wide range of scenarios and potentially many missions within each scenario.

Elaborate, lengthy acquisition processes have been adopted around the world to address these issues and the generally high levels of complexity of military systems. These processes, however, are somewhat at odds with contemporary commercial information systems practice in two major ways.

1. It is customary to evolve civil information systems in conjunction with the enterprise they serve. While information systems development is frequently problematical (Standish 1995), established processes exist that can be employed to tackle the task, and should the resulting information system be less than ideal (a common occurrence), the issues will be immediately apparent and problem rectification commenced without delay.
2. Most enterprises can be characterized as having stable, enduring activities against which an information architecture can be designed. In contrast, for each military operation, a purpose-matched task force is assembled from an inventory of available assets. This task force is

This work was partially funded from the DSTO SEEC Centre of Expertise Contract.

then often integrated with other task forces to form an international coalition force. Over the last decade or so, there have been several examples of problems with information system integration that have delayed the initiation of peacekeeping operations (refs). The FBRET research effort, of which CREAP is only a part, seeks to ameliorate this problem.

THE CLASS OF PROBLEM THAT CREAP IS ADDRESSING

Unlike a traditional development where equipment is designed to fulfil a need, the task force information system problem can be characterized as an unprecedented, real-time component integration problem. (The term unprecedented refers to the fact that many of the parts will never have been integrated before.) The unprecedented nature of the problem means that there would be a dearth of experience from which to draw to avoid integration problems.

Like any systems design problem, task force information system design starts with a user needs statement. The needs of the user can be obtained from planning documents but many will be mission-specific for missions that may or may not resemble the proposed rapid deployment about to take place.

The role of CREAP-like tools is to assist a knowledgeable user to translate these high-level user needs into a system design that can be implemented from available components (inventory). To do this task, the tool needs to be aware of the contents of the inventory and the interfacing potential of each piece of equipment and be able to make decisions as to the suitability of the combination of components for the desired mission.

The CREAP is an instance of such a tool for the purpose of formulating the communications for a rapidly deployed task force using communications components in inventory. It also needs to reason using a communications (and information systems) architecture model that can estimate the performance of the proposed information system.

THE FBRET APPROACH

The FBRET implementation of the CREAP is an extension of expert system programming techniques based on combining work by (Cook 1990, 1991, 1993), and (Kasser 1992).

Expert systems take the form of software packages residing on a hardware platform (computer). An expert system consists of three elements

- **An User Interface** such as a standard Windows platform graphical user interface (GUI). The user is presented with menu choices and enters information either with a mouse click or by typing some text into a data entry box.
- **A Knowledge Base** that contains the information about the capabilities of the elements of the system. It also contains the rules governing their interaction.
- **A Inference Engine** that computes using the information in the Knowledge Base to provide the desired output.

A common use of an expert system is via a man-machine dialogue. The user types something at the keyboard and the system replies. The user interface accepts the input. The input is parsed in some manner. The inference and knowledge engines process the user input in a predetermined manner and an output appears. The parsing function may be a simple pattern matching method commonly called keyword analysis, or a more complicated function using syntactic analysis. Keyword analysis is a logical function in which the presence of various keywords are detected. When a keyword is found, the system responds in the manner in which it has been programmed. In syntactic analysis, a sentence is analyzed according rules which allow the system to respond differently to keywords which appear in different sequences.

The semantic network is the most general and oldest artificial intelligence scheme for representing knowledge. A semantic network is a collection of objects called nodes. The nodes are connected together by links. Ordinarily, both the links and the nodes are labeled. A drawing of a semantic network contains bubbles to represent the nodes, and lines connecting the nodes to represent the links. Both nodes and links are labeled. The drawing looks just like a PERT or CP/M chart. It is also the drawing used to represent a state machine.

A state machine is a system that contains a number of states yet exists in only one of them at a time. The system makes a transition from one state to another as a result of the receipt of a stimulus. All states and transitions are defined at the time the system is designed. The links in the semantic network are the transitions in the state diagram.

From the user's perspective the system appears to be in a rest state waiting for user input. It then receives an input from the user and enters a transition to a state in which it processes the data it has just received. After processing the input, it generates an output and returns to what seems to be the same rest state. In reality, the two rest states (initial and final) are different. For example, the system maybe in resting in State 26 waiting for a user input. When the user enters some data, the interpreter moves the system to State 27 and processes the user input, advaces to State 28 to generate an outpout and then to State 29 to await the next user input.

PRIOR WORK

Specriter. (Cook 1990, 1991, 1993), discusses a structured document generation and processing shell, which represents the earliest reported knowledge-based system for computer-aided generation of equipment specifications. Specriter elicited user input through a series of context-sensitive screens and tailored the list of questions to suit the problem at hand. Specriter contained domain knowledge from specification practices, measurement science, and equipment design and testing. It was able to use this to provide feedback on the quality of a response and had some limited consistency checking. On completion of the specification elicitation, checking, and editing functions, the tool was able to process the model of the specification held in the frame structure in Prolog to produce a written document in MIL-STD-490A format.

ELMER. Kasser (1992) introduced ELMER, a simple expert system based on a state machine written as a student term paper in a postgraduate class in system engineering at The George Washington University. ELMER had ways of reacting to a word match. ELMER's frame contained six entries

1. **Current State**, the state that the string match is performed in.
2. **Next State,** the state that the ELMER will advance to if a string match is found.
3. **Repeat Flag,** a flag to allow or disallow repeats.
4. **Command Flag**, a number which determines how a data file is treated. It could
   - Do nothing,
   - Output a text file,
   - Execute a program, or
   - Overlay a new state table.
5. **Keyword,** the text string to match in the syntactic analysis of the input text.
6. **Data File**, the file to be processed in the current state.

THE CREAP IMPLEMENTATION OF FBRET

The CREAP implementation of FBRET extends the work begun in Specriter and ELMER as follows. Communications links are made up of a number of elements in series. For example, terminal, modem, transmitter, transmitting antenna, free space path loss, desired link margin, receiving antenna, receiver, modem, and terminal. Each element can be considered as having properties (some amount of either gain or loss). Thus for a given communications category of service (audio, video, or data), the computed received signal strength needed is the sum of the properties (gains and losses) of the specific elements chosen for the link. This value can be compared with the computed value and a decision made to determine if the properties of the selected components would provide a link. This is a design-to-inventory situation complicated by the large number of possible combinations of inventory items each having different properties. For example, several combinations of transmitters and antennas could provide the required uplink radiated signal power, while other combinations would not.

CREAP Use Cases. The following Use Cases demonstrate the capabilities of the CREAP.

- **Operational Scenario OS1:** The user chooses a category of communications service (audio, data, video) then progressively selects the equipment to build the communications link. As the user selects the equipment, suitable items that can provide a link within the Bit Error Rate (BER) limits for the category of service are progressively enabled.
- **Operational Scenario OS2:** The user chooses a category of communications service then progressively selects the equipment to build the link. As the user selects the equipment, suitable items that cannot provide a link within the BER for the category of service are progressively disabled.
- **Training Scenario TS1:** The User progressively selects one item from a number of options for each element of the link. After each selection CREAP enables the choices for the next link element. After all of the link elements have been selected, the user clicks on the *Evaluation* button to see the results of the choices. If the componets selected to esablish the link will provide service then CREAP notifies the user of the fact. If not, CREAP notifies the user that the selection will not meet the user needs.
- **Training Scenario TS2:** The user progressively selects one item from a number of options for each element of the link. If the element selection will provide service then CREAP enables the next link element option, else CREAP notifies the user of the error.

The CREAP Architecture. Unlike traditional expert system implementations that separate the knowledge base and the inference engines, FBRETs as illustrated by the CREAP, combine the knowledge bases and inference engine in frame structures. The concept of frames as a knowledge representation technique was initiated by Minsky (1975), although he gives credit for many of the concepts to Bartlett (1932). CREAP contains the following elements

- **The Frames** are designed to be general purpose so that the contents of the frames can be replaced by a set of frames for a different scenario allowing the tool to be used in another scenario potentially with no additional programming.
- **The user interface template**, which provides the

template for the information to be displayed on the GUI.

- **The frame interpreter,** which is a state machine, which interprets the contents of the frames.

There is also a meta-inference engine that is employed when populating the frames. The domain (subject matter) expert currently performs this function.

THE CONTENTS OF THE FRAMES

The information contained in the CREAP frames depends on the frame. The frames are organized in a hierarchy to provide the flexibility for a general-purpose tool. Thus each frame may contain some or all of the following elements

1. Frame identification number for linking purposes in the state sequence.
2. The predefined type of frame.
3. The current state in the computation path
4. The next state in the sequence.
5. The next frame to move to (parent and child frames).
6. Frame type dependent information such as
   - User interface text and graphics (or pointers) to fill the GUI template.
   - Expected user responses.
   - Link elements, and their properties.
   - Fixed or parametric relationships between the contents of the frame and the next in the computation sequence.
   - The type of relationship (equation).
   - Reasoning Rule number to be applied.

Rules may be implemented in a sequence of frames to simplify the design. For example, the sequence of displaying visual components in the GUI may be implemented in a series of frames. Assuming State 73 is the state in which a specific set of visual components is to be displayed on the GUI. The states may change in a manner represented by the following sequence

State	Operation	Next State
73	Clear GUI	74
74	Display component 1	75
75	Display component 2	76
76	Display component 3	77
77	Display component 4	78
78	Display component 5	79
79	Wait for user input	80

Typical predefined frames are

1. Action frame similar to the ELMER frame.
2. GUI display frame.
3. Component element frame.
4. Data computation frame. The equation used to process component element entries is a function of a parameter in the frame. This concept is inherited from table-driven software concepts.

Frames can read and store (write) information to locations in their own or in other frames[1]. However, the convention has been established that while frames may read information from their own or other frames they may only store information in their own frame. This convention minimizes errors since the errors do not overwrite information in unknown locations thus ensuring that information is always current.

THE INTERNAL MODELS

Reiterating, the CREAP is not designed as a tool for computing communication link performance. CREAP was designed as a general-purpose, frame-based expert system to assist in the choice of various combinations of equipment that meets requirements from an inventory. To demonstrate the concept, a communications scenario was selected. A hypothetical rapid deployment force will need communications. Equipment is in inventory and must be selected. However not all combinations of transmitters and antennas will meet the uplink requirements. Two communications models are to be implemented; a satellite communications model, and a high frequency radio communications model. The two links are similar in concept and implementation. However, the parameters used to compute the availability of communications links are different (satellite communications uses free space path loss at specific frequencies, coupled with atmospheric absorption, while H.F. communications is subject to variations in the ionosphere and incorporates a propagation prediction model).

The choice of two communications models was made to provide a range of types of parametric equations that need to be interpreted. The wider the range, the more complete the tool is, and the greater the degree of flexibility without the need to modify the software engine. While CREAP is extendable by defining new frame templates at any time, each time a new frame template is developed the frame interpreter may need upgrading. The two models in the initial CREAP implementation should provide a wide range of generic frames. The only change to the CREAP to switch models is to change the set of frames. Thus CREAP could be extended to incorporate the choice of models simply by adding a new set of frames that would be used to establish the type of communications link desired.

[1] In actuality a frame only stores instructions to read or write information. The interpreter does the actual reading and writing of information.

The satellite communications model. The link elements of a digital communication link in the Satellite communication Model are

- Uplink Transmitting Modem
- Uplink Transmitter
- Uplink Transmitting Antenna
- Uplink free space path loss
- Communications satellite
- Downlink Free space path loss
- Downlink Receiving Antenna
- Downlink Receiver
- Downlink Receiving Modem

Communications are only possible when there is a sufficiently large value of signal at the receiving end of the link. Thus the link power budget for any specified BER at any specified frequency for any category of service may be calculated using a number of equations readily available in any textbook on the topic. For example, the input power to a receiver is given by (Miller 1993, p.44):

$$P_r = P_t + G_t + G_r - L - M \ (dBW) \quad (1)$$

Where: P_r : received power at receiver

P_t : transmit power of transmitter

G_t, G_r: gains of transmit and receive antennas

L: total losses between that transmit antenna and receiving antenna

M: Link margin

Thus for any given category of service, the category will have a specific minimum BER specification. Once the link characteristics are known it only remains to determine if a selected combination of components can provide the requisite minimum BER and received signal strength values. Thus, a digital link is possible only if all of the following conditions (rules) are satisfied

1. The maximum bit rate of both of the modems are greater than the required bit rate for the selected category of service.
2. Both of the modems have the same type of modulation.
3. The received signal power at the receiver has to be greater than the minimum computed value.
4. The BER of the link is less than maximum allowable BER for the selected category service.

The H.F. communications model

The link elements of a digital H.F. communications link in the H.F. communications model are

- Transmitting Modem
- Uplink Transmitter
- Uplink Transmitting Antenna
- Ionospheric path loss (depending on frequency, time, and solar conditions)
- Downlink Receiving Antenna
- Downlink Receiver
- Downlink Receiving Modem

For any given category of service over the hf link, the category will have a specific minimum BER and minimum bandwidth specification. Once the link characteristics are known it only remains to determine if a selected combination of elements can provide the requisite minimum BER, received signal strength, and bandwidth values. Thus, a digital link is possible only if all of the following conditions (rules) are satisfied

1. The maximum bit rate of both of the modems are greater than the required bit rate for the selected category of service.
2. Both of the modems have the same type of modulation.
3. The received signal power at the receiver has to be greater than the minimum computed value.
4. The BER of the link is less than maximum allowable BER for the selected category service.
5. The maximum link bandwidth is greater than the required link bandwidth for the selected category of service.

FROM MODELS TO CREAP

Implementation of the models is straightforward. The Use Cases provide the sequence of events for the frames. Thus each model is a true sequential program. The interpreter has to be built to interpret the contents of the frame. The interpreter can be built using Case statements to modularize the software needed for each command flag operation. While an individual frame may contain information similar to that in other frames, each frame in the CREAP is unique. A tool will need to be developed to expedite future frame construction and population. This tool and other applications are planned to be the subjects of further research.

CREAP IN USE

The CREAP prototype inhibits users from selecting inventory link components that will not provide the needed service. It is a useful tool in situations wherein decisions have to be made quickly by people with minimal domain expertise. Figure 1 provides a screen display from the prototype.

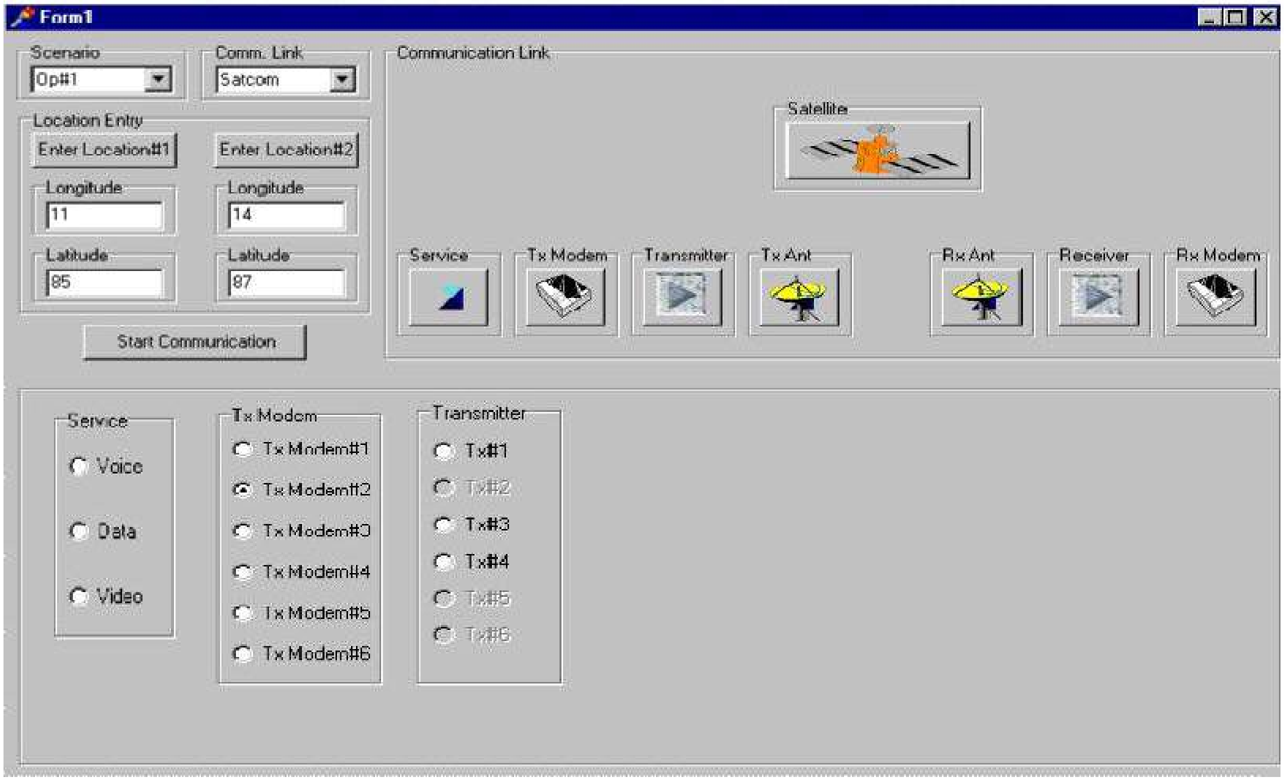

Figure 1 Typical Screen Display

SUMMARY

The CREAP implementation of FBRET provides a very flexible tool that extends early work on simple expert systems by the incorporation of parametric relationships. The design to inventory situation can be addressed at many levels of system engineering.

CONCLUSIONS

The CREAP implementation of a FBRET using interpreted frame based knowledge and inference provides a very flexible and readily testable tool that provides a capability that is new. Thus Requirements Engineering tools can be extended by the FBRET approach.

REFERENCES

Bartlett F.C. (1932). *Remembering*, Cambridge University Press, Cambridge.

Cook S.C. (1990), "Knowledge-Based Generation of Measuring Instrument Specifications", in *IMEKO International Symposium on Knowledge-Based Measurement*, 1st, Karlsruhe, FDR, pp145-152, 1990.

Cook S.C. (1991), *A Knowledge-Based System for Computer-Aided Generation of Measuring Instrument Specifications*, Ph.D. thesis, City University, London, UK.

Cook S.C. (1993), "A Knowledge-Based Specification Generation System for Computer-Aided Production of Measuring Instrument Specifications", *Measurement*, Vol 11, pp 235-255.

Cook S.C., Kasser J.E. (2001) Asenstorfer, J., "A Frame-Based Approach to Requirements Engineering", *11th International Symposium of the INCOSE*, Melbourne, Australia.

CREAP "A case study", submitted in parallel with this paper.

Kasser J.E (1992) "ELMER: An Expert System Based on a Finite State Machine", *AMSAT Symposium*, Washington DC.

Kasser J.E. (2000) "A Framework for Requirements Engineering in a Digital Integrated Environment", *SETE* 2000.

Miller M.J., Vucetic B.., Berry, L. (1993). *Satellite Communications Mobile and Fixed Services*, Kluwer Academic Publishers, Massachusetts.

Minsky M. (1975). "A framework for representing knowledge." The Psychology of Computer Vision, Ed. Winston P.H., McGraw-Hill, New York, p211-277.

Standish (1995), *Chaos*, The Standish Group, http://www.standishgroup.com/chaos.html, last accessed March 19, 1998.

AUTHOR'S BIOGRAPHY

Joseph Kasser has been a practising systems engineer for 30 years. He is the author of "*Applying Total Quality Management to Systems Engineering*" published by Artech House. Dr. Kasser is a DSTO Associate Research Professor at the University of South Australia (UniSA). He performs research into improving the acquisition process. Prior to taking up his position at UniSA, he was a Director of Information and Technical Studies at the Graduate School of Management and Technology at University of Maryland University College. There, he developed and was responsible for the Master of Software Engineering degree and the Software Development Management track of the Master of Science in Computer Systems Management (CSMN) degree. He is a recipient of NASA's Manned Space Flight Awareness Award for quality and technical excellence (Silver Snoopy), for performing and directing systems engineering. Dr. Kasser also teaches systems and software engineering in the classroom and via distance education.

Professor Stephen Cook. After graduating in Electronics Engineering from the South Australian Institute of Technology in 1977, Prof Cook commenced work as a telecommunications equipment design engineer in the UK where he also completed an MSc in Computer Science at the University of Kent. On his return to Australia, he worked in the defence electronics industry until 1988 when he joined the Defence Science and Technology Organisation (DSTO). He completed a PhD in 1990, in Measurement Science and Systems Engineering at the City University London. Prof Cook joined the University of South Australia as the Foundation Professor of Systems Engineering in 1997. He has contributed to three books and has published over fifty refereed journal and conference papers. Prof Cook is a Fellow of the Institution of Electrical Engineers (UK), a Fellow of the Institution of Engineers, Australia and Past President of the Systems Engineering Society of Australia.