An Animated Cryptographic Learning Object

Rachid Anane[1], Kevin Purohit[2] and Georgios Theodoropoulos[2]
[1]*Department of Computer Science, Coventry University, UK.*
r.anane@coventry.ac.uk
[2]*School of Computer Science, University of Birmingham, UK.*
{k.purohit, g.k.theodoropoulos}@cs.bham.ac.uk

Abstract

Algorithmic animation has been the focus of intense research in many disciplines and its impact on the educational process has been marked by increasing learner autonomy. Research in this field is driven by the belief that algorithm animation can be a more effective means of instruction than manual or verbal modes of delivery. Encryption algorithms in particular offer an interesting domain for the application of animation principles in learner/content interaction. The main challenge however has been how to design effective animations with a pedagogical value. This paper is concerned with the presentation of an animation of the DES algorithm that exhibits many of the features of useful instructional material. Its pedagogical value is expressed in terms of intrinsic qualities and, in particular, the degree of interactivity and the granularity of abstraction.

Keywords: Encryption, DES, algorithm animation, pedagogical value

1. Introduction

Computer science instruction is increasingly relying on animation as a way of addressing the complexity and the dynamic nature of many core topics. It is also providing a motivational impetus and offering students greater autonomy. The educational concerns that are driving the research into animations have also shaped the mode of interaction that their use entails and constrained their application; one major issue in the design of these systems is to ensure that the meta-tools maintain accuracy and facilitate the understanding of the subject matter in a transparent manner.

Algorithm animation has been the subject of much attention and various systems have been proposed and their impact evaluated. In this context the learning content goes beyond mere e-distribution: it allows learners to control and to follow the dynamic unfolding of the algorithm as the data is being transformed [2].

Research into this field is driven by the belief that algorithm animation can be a more effective means of instruction than mere static text-based resources. It is considered as a serious alternative to verbal modes of delivery such as lectures [1]. There is also a growing awareness that the emergence of what is termed the 'Nintendo generation' is influencing the delivery of courseware material. Modes of interaction that have greater affinity with the pervasive nature of multimedia environments are being promoted vigorously [3].

These modes of operation are deemed to have a greater motivational power than text-based methods and to facilitate student-centred learning. Among the benefits that accrue from the interactions with simulations in general and animations in particular is the validation and refinement of the conceptual models of the underlying objects of investigation [4]. Simulation in general is extremely important in engaging learners with the content and in motivating independent investigation [5]. The role of animation in enhancing the learning process has been confirmed by many studies [6], although in some cases the outcome of the experimental results was not conclusive [7]. One major criticism levelled at many animations is that the user is passive and has no control over the animation apart from the initial start [1]. The use and impact of animations on the learning process has brought to the fore the issue of how to evaluate their pedagogical value, namely their contribution to the understanding of the underlying concepts. These concerns are particularly relevant to encryption algorithms because of their abstraction and their inherent complexity.

An understanding of security in general and encryption in particular has become a requirement for students of computer science. With the increasing awareness of the risks associated with networks and distributed systems the deployment of sophisticated methods of encryption is also associated with the demand for suitable instructional material in encryption principles and models. Although encryption algorithms are usually complex and their behaviour is difficult to visualise, they follow sequential and discrete stages that are well suited to animation. The design of adequate animations has presented researchers with the challenge on how to unravel these stages in a clear and unambiguous manner. Various systems have been proposed with different emphasis on levels of interaction and the inner operations of the encryption process [8, 9].

This paper is concerned with the presentation of the animation of the DES encryption algorithm and a consideration of its pedagogical value. The DES algorithm is part of a suite of animated encryption algorithms such as the substitution algorithm; the animation of the exchange of keys in Kerberos is used as a motivational element. The animation is presented as a tutorial.

The remainder of the paper is organised as follows. Section 2 identifies the requirements for the design and deployment of animations. Section 3 gives a procedural description of the DES algorithm. Section 4 presents the salient features of the DES animation in terms of screenshots with a brief outline of its implementation. Section 5 gives an evaluation in terms of conformance to evaluation criteria and comparisons with some existing systems. Section 6 concludes the paper.

2 Design Considerations

The enhancement of the pedagogical value of the encryption algorithms was the main motivation behind the design approach used for the animations. The pedagogical value of the animation is formulated in terms of two requirements, a contextual requirement and an interaction requirement. These refer to the intrinsic qualities that an animation should possess.

2.1 Context

The contextual requirement is a prerequisite for e-learning and its fulfilment depends on the adequacy of the underlying distributed system. The DES animation should address the following issues:

***Access and autonomy*:** the animation should be designed as a Web-based application in order to facilitate its access from anywhere and at anytime. It should be self-contained and supported by contextual information; standard mathematical notations should be adhered to whenever possible. The animation, by definition, should be deterministic and repeatable. It should also foster autonomy in the learning process by decoupling instructor and student.

Motivation: a motivational framework should be provided through a specific topic such as key exchange in Kerberos and the intrinsic value of encryption should be highlighted. As an aid to the understanding of the structure of encryption algorithms and their dynamic unfolding animation should make them more accessible and less intimidating. Students should have the opportunity to investigate different levels of complexity, from transposition ciphers to DES as the archetypal cipher. Where applicable various cognitive activities should be combined in the instruction process.

2.2 Interaction

The learner/content interaction is considered as the main focus of the learning process. Its scope is formulated along various concepts [10]. This interaction is however more complex because it involves the animation of both the operational framework and the data. The interaction should take into account the following:

***Focus*:** a learning object should have well-defined objectives. The animation should emphasise what the algorithm is doing (through data transformation) and how it works (through a clear and annotated sequence of steps).

Abstraction: discrete levels of abstraction should be incorporated in the animation. The user should be able to go through the sequence of steps at various levels of abstraction and initiate transitions between stages. The learner should control the encryption process with the ability to focus on specific steps or skip them if required.

***Navigation*:** the algorithm unfolds dynamically in the transformation of the data. The student should be able to accommodate different types of learner. A structural view of algorithm should be explicit so that the learner can determine the status of the encryption process as it moves from one stage to the next.

Reduced cognitive load: the main tasks and operations of the encryption should be clearly identified, and the information manipulated or carried across the different stages by the learner should be minimal. The animation should make the obscure parts of the DES algorithm more accessible and understandable [11].

Correspondence: a one to one mapping between the steps of the algorithm and the structure of the animation should be evident and supported by a relevant integration of animation and text. The system should also accommodate different modes of representation if applicable.

3 Data Encryption Standard (DES)

The DES algorithm is based around Horst Feistel's Lucifer Cipher, which was the first among many Feistel ciphers. A Feistel cipher works by applying the same function to the plaintext for numerous rounds, with each round being parameterised by a round key (derived from the main key). DES is a block cipher, which operates on 64-bit plain text blocks and uses a 64-bit key of which only 56 bits are used to create cipher text. The remaining 8 bits are discarded or used as parity bits.

3.1 Encryption

The DES algorithm is made up of two main stages: the key schedule where sixteen sub-keys (round keys) are generated, and the plaintext block encoding, where the Feistel cipher is applied.

Key schedule

The key schedule involves the generation of sixteen 48-bit subkeys (the round keys) from the original 64-bit key, as follows:

1. The original 64-bit key is stripped of every eighth bit and permuted using the Permuted Choice 1 **(PC-1)** table.
2. The resulting 56 bits are split in two 28-bit blocks, C_0 and D_0.
3. For each round i, C_i and D_i are computed as follows:
   $C_i = C_{i-1} <<< p_i$
   $D_i = D_{i-1} <<< p_i$
   where $<<< p_i$ is 1 cyclic shift for i = 1, 2, 9 or 16, and 2 cyclic shifts for other values of i.
4. C_i and D_i are then concatenated $(C_i\ D_i)$ and permuted according to the **PC-2** table to give the subkey K_i.
5. All the subkeys are generated in a similar manner, for i = 1,..,16.

Feistel Cipher

Once the key schedule is complete, the algorithm then encrypts the plaintext, a 64-bit block, permuted following the initial permutation **(IP)** table to give a 64-bit cipher text. The output of the initial permutation is split into two 32-bit halves, L_0 and R_0, which are then subjected to the sixteen rounds of the Feistel cipher as follows:

For each round i = 0, …,15

$L_{i+1} = R_i$

$R_{i+1} = L_i$ **XOR** $F(R_i, K_i)$

The equations state that the right half R_i of the plaintext and key K_i are put through the Feistel function, denoted by F; its output is then XORed with the left half L_i. The original right half and the result of the XOR operation are then swapped and are used in the next round as the new L_i and R_i. The exception to this operatrion is in the final sixteenth round where there is no swap. For each round the DES Feistel function consists of the following stages:

1. Expansion: the 32-bit block, R_i, is expanded into 48 bits by using the Expansion Permutation table. This essentially involves duplicating 16 bits and permuting them.
2. Subkey addition: the 48-bit block is XORed with the K_i subkey for that round.
3. Substitution: the resulting 48 bits are then split into 8 blocks of 6 bits each, which are used as input to eight substitution boxes (S-Box). The S-Box takes the concatenation of the outer two bits of the 6 bits as the row, and the four inner bits as the column of a lookup table to return a 4-bit group. The output from each S-Box is concatenated sequentially to generate a 32-bit output. The first 4-bit output is generated by **S1** (S-Box 1) and the last group of 4 bits by **S8**. The S-Box substitutions are designed to make cryptanalysis difficult and a slight change to the tables can significantly reduce the security of DES.
4. Permutation: the 32 bits from the S-Boxes are permuted according to the Permutation Box **(P-Box)** to yield the final result of the Feistel function.

At the end of the sixteenth round the order of L_{16} and R_{16} is reversed to produce a block of 64 bits, $R_{16}L_{16}$, to which the final permutation is applied according to the inverse permutation **(IP^{-1})** table. This final permutation represents the cipher text of the 64-bit plain text.

3.2 Decryption and Modes of Operation

The decryption process works in the same manner as the encryption with one exception: the generation of the round keys occurs in reverse order, with a slight modification to the key schedule.

The DES algorithm transforms a 64-bit plain text into a 64-bit cipher text. The algorithm may however follow different modes of operation in the encryption of a whole text, which is split into 64-bit blocks. If the encryption of each 64-bit block is independent of the other blocks, the mode of operation is called Electronic Code Book (ECB); otherwise two other modes of encryption may be used, the Chain Block Coding (CBC) or the Cipher Feedback (CFB).

3.3 DES Characteristics

From the previous description of the DES algorithm it is evident that the algorithm is quite complex in the unfolding of its different phases and in the utilisation of a variety of complex mathematical transformations. This complexity is confirmed by some preliminary research, which reveals that many students found the use of the permutation and the substitution tables as major sources of confusion. DES has also the characteristic of introducing various operations while relying on carefully designed tables to direct their application. The combination of bit-shuffling (permutations or P-box), and simple non-linear functions (S-boxes) as well as simple mixing (XOR) has the merit of generating both diffusion and confusion, often considered as properties of secure ciphers. Diffusion is created by permutations or transpositions techniques while substitutions mechanisms are aimed at introducing confusion.

The DES algorithm is mature and has been very influential. Many recent encryption algorithms, such as the Advanced Encryption Standard (AES) and Blowfish, are based on design principles similar to DES. All these factors confer to the DES algorithm a special status in symmetric encryption, especially as it subsumes the architecture of transposition and substitution encryption schemes. In addition to a clear identification of the different stages, different levels of details and complexity are also manifest.

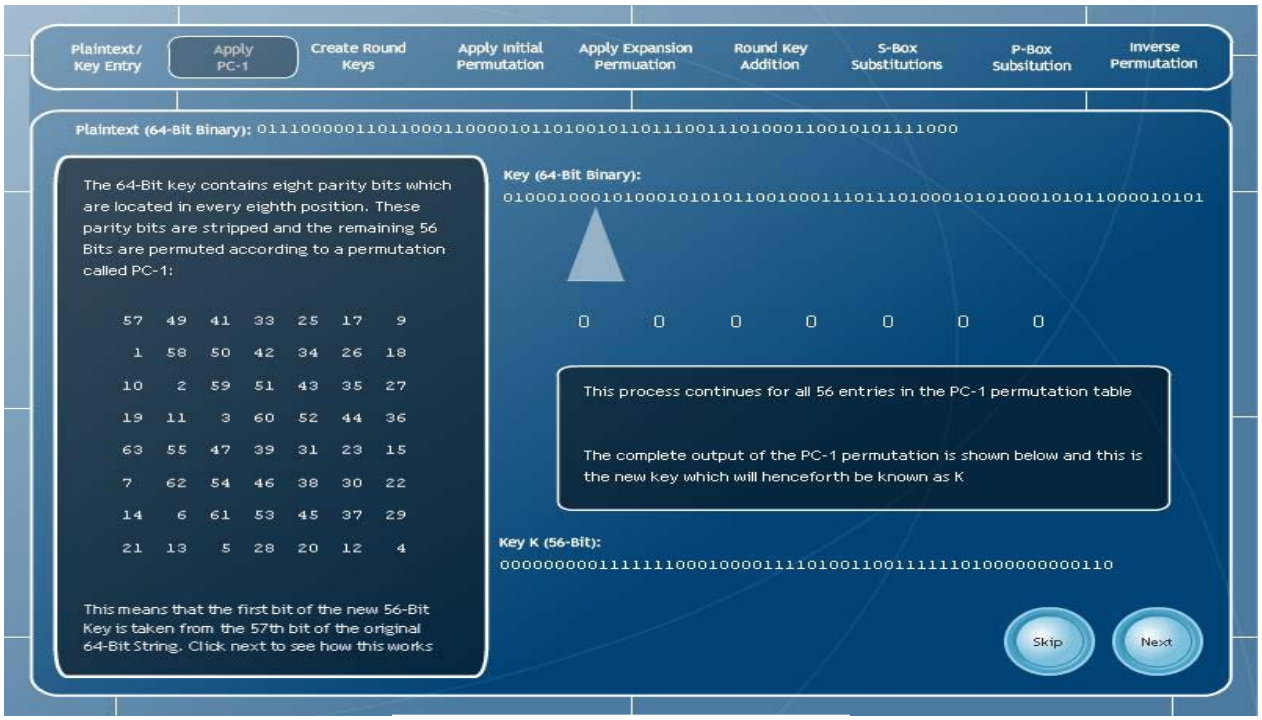

Figure 1. Permutation Table

4 DES Animation

The description of the DES animation follows the procedural nature of the DES algorithm. The focus is on the key stages of the algorithm with particular emphasis on the most obscure functions. Since various tables are focal points of activity their role and their operations should be made explicit. The user interface is consistent across the different animations. The DES animation is presented as a tutorial with nine stages.

For each animation a bar appears at the top of the screen. It indicates the progress of the animation and the current stage of the algorithm. A dialog box, providing information and instructions to the user, accompanies each stage of the algorithm; clearly visible navigation buttons are also included. The transitions between stages will be animated to indicate that the result of one stage of the algorithm is passed on to the next.

4.1 DES Introduction

The first frame of the DES animation offers a brief description of the algorithm and invites the user to start the encryption process. The user can enter 8 ASCII characters for plaintext and 16 Hexadecimal numbers for an encryption key. When the *encrypt* button is clicked, the cipher text is displayed in hexadecimal and ASCII notations so that the user can view the final cipher. The binary versions of the plaintext and key are computed by the server and displayed. The user can initiate the animation of the encryption process by clicking the *next* button.

When the next button is clicked, the fields for the binary plaintext and key are animated over 20 frames to appear at the top of the next screen. This enables the user to track the progress of the plaintext throughout the transformation process. Any text or image, which is carried forward onto the next stage is animated.

4.2 Interactive Permutation Tables

At various stages of the DES algorithm, permutation tables such the Initial Permutation, P-Box or Inverse Permutation, are applied to binary strings. These have proven to be a major source of confusion. A permutation animation consists of an interactive permutation table (clickable), a position arrow and the output of the permutation (Figure. 1). When the user clicks on one entry in the permutation table it is highlighted in bold; the position arrow moves accordingly to indicate the corresponding bit in the input string. The user is thus able to generate the output string bit by bit. All the permutation tables work in a similar manner and the user has the option of skipping the permutation phase and moving on to the next stage.

4.3 Key Schedule

The description of the DES algorithm in Section 3 has shown that the key schedule is a repetitive process, and as such the animation of all sixteen rounds would add very little to the educational value of the tutorial. It was

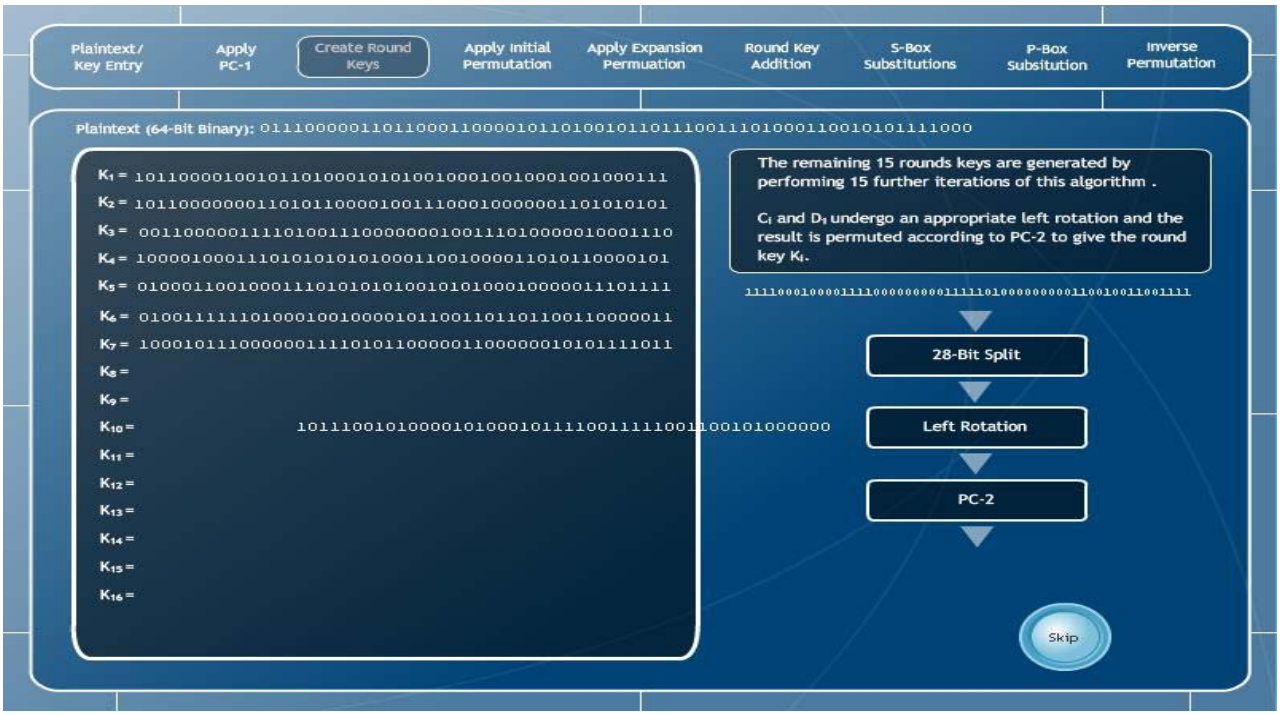

Figure 2. Key Generation

therefore decided that in the key schedule only the generation of the first key K_1 would be shown in detail. The key generation is shown in an animation where the user can generate the keys and view the process without going through all the stages (Figure. 2). The user has the option of viewing the generation of all sixteen round keys or to skip the animation.

4.4 Feistel Cipher

The initial permutation (IP) behaves in a similar manner to the interactive permutation table described above. The result of the initial permutation is unveiled by the different positions of the arrow. Since this forms the first stage of the Feistel cipher the output string is split into two sub-strings of the same size. The result of the round key addition then goes into the S-Boxes. Special care was taken in the design and implementation of the S-Box animation, as it is another a source of difficulty and confusion in the DES cipher. Various representations of the S-Box have been introduced sequentially in order to elucidate its behaviour. The S-Box can be considered as a black box without any explanation of the inner operations. The four bits generated by each S-Box are simply displayed as output

The user has also the option of opening any S-Box by clicking on it. This stage of the tutorial demonstrates in detail how the S-Box substitutions work. It shows an animation of the stripped outer bits, the inner bits and an animated table lookup (Figure 3). Although five distinct stages have been allocated to S-Boxes, the user can skip intermediate stages and investigate various levels of abstraction.

4.5 Decryption

The tutorial includes an animation and explanation of how the decryption process works. This animation presents a higher level of abstraction in the decryption process. It encapsulates all the stages of Feistel cipher in one single diagram.

4.6 System Implementation

The system architecture is shaped by the need to address the issue of access and availability. The underlying technology should also provide support for animation and user interface design, without weakening the semantics of the interaction.

The system architecture conforms to the client-server model and is implemented using technologies that will ensure the platform independence of the system. The integration of Java and Flash in the implementation was aimed at enhancing the interactive features of the teaching material and maintaining the semantics of the animation. The client requests are handled by JSP acting as middleware with the algorithms running on the server side. A clear separation between client side and server side was enforced in order to allow for enhancements to be introduced seamlessly. For example, it is possible to replace the implementations without any change to the animation or the interface.

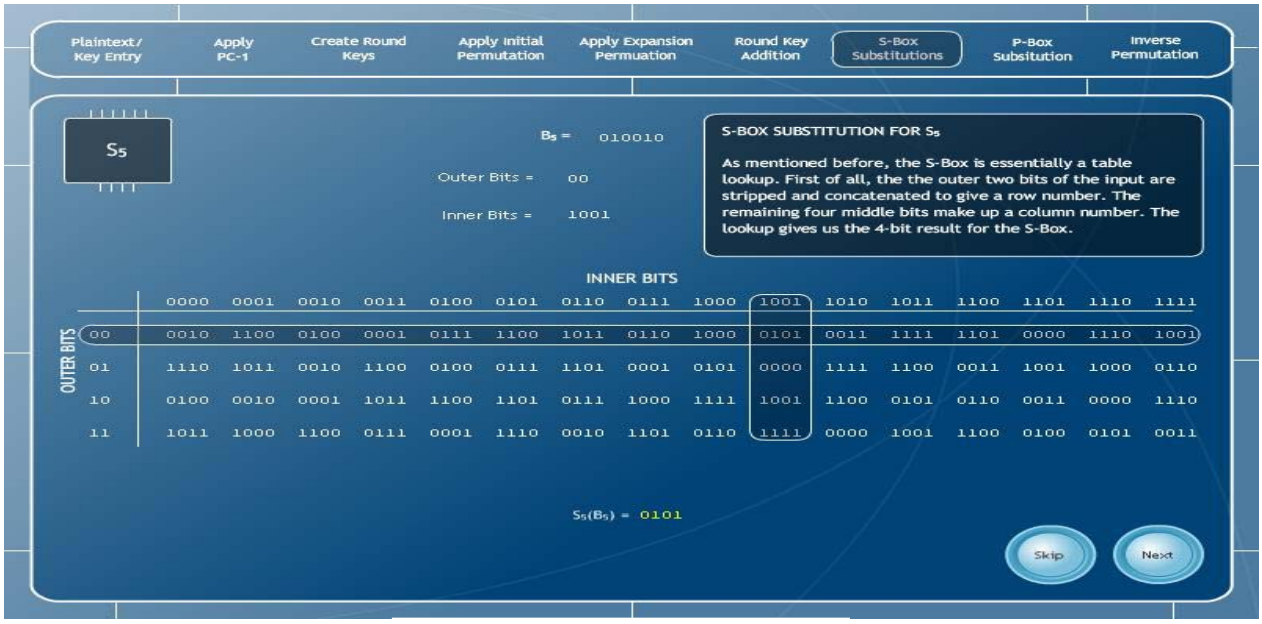

Figure 3. S-box Substitution

5 Evaluation

This section is concerned with the determination of the pedagogical value of the DES animation. Following the discussion in Section 2, the pedagogical value refers to the intrinsic qualities of the DES animation. The animation will also be put in perspective through a brief appraisal of some published DES animations. Although a preliminary user evaluation of the animation was conducted with a small group of users its lack of formality precludes its discussion here despite its positive feedback. The group played an important role in determining the scope of the animation and the final application conformed to their requirements. A formal evaluation with a control group will produce a more objective evaluation of the pedagogical value of the animation.

5.1 Context and Interaction

The evaluation of the context and interaction of the DES animation will be framed in terms of a model of evaluation of interactive education software proposed by Reichert and Hartmann [12]. They introduce some criteria for good interactive Education Software in their evaluation of the learning in e-learning, a concept that overlaps the notion of pedagogical value. The 'criteria for good interactive Education Software' involve the following:

- Promotion of fundamental ideas
- Incorporation of different cognitive levels
- Affordance and a high degree of interactivity
- Provision of feedback
- Support for effective visualization and usability

The application of these criteria to the DES animation yields the following points:

The content is based on fundamental ideas: security is a core concept in computer science and DES can be considered as an archetypal encryption algorithm with an intrinsic value. It has the advantage of subsuming the permutation and substitution schemes and demonstrates effectively the properties of confusion and diffusion. DES is also based on sound mathematical ideas.

Incorporation of different cognitive levels: the learner is engaged in the exploration of different levels of abstraction. The main cognitive activities involve comprehension of encryption processes and acquisition of knowledge. Should the animation be incorporated into a blended programme other cognitive activities such as problem solving can be also invoked. The DES animation can be the subject of various pedagogical activities.

High degree of interactivity: within the constraints of the encryption process the system is highly interactive. The level of interactivity depends on the nature of the algorithm and the dependence of intermediate values on the initial input. Within these constraints the learner is able to control the pacing and the sequencing of the animation. The taxonomy proposed by Schulmeister can be used as a benchmark for evaluating the level of interactivity of the application. In this classification level 1 stands for no interactivity and level 6 enables 'visualisation generated by students' [13]. The DES animation operates at level 2 since navigation through the animation is fully supported. Furthermore, as multiple representations of the same function are provided, level 3 is also partially covered. This is particularly relevant to the different representations and operations of the S-Box. Higher levels of interactivity namely level 4, 5 and 6 are not supported because of the procedural and the sequential nature of DES.

Abstraction levels: as the algorithm is highly iterative many intermediate operations are duplicated. The user has the option of controlling their visibility in order to improve readability and focus on the main steps of the algorithm. Users can also situate themselves in the encryption process as it unfolds through the different stages; an animated pointer sweeps the corresponding bit string. Control over the pacing of the animation is also afforded at different degrees of granularity: at the bit level in the gradual generation of the bits of the round keys, at the group level in the substitution, at block level in the application of the Feister function, at stage level in the unfolding of the algorithm and finally at cipher level in the direct encryption of plaintext into cipher text. Different levels of abstraction are clearly manifest in the screens dedicated to the S-Box transformation. The user is able to delve into the low level details of the substitution if required.

Feedback: this feature takes two forms. The first concerns the transformational impact and the visibility of the effect of selected operation on bits. The second deals with the navigational process in terms of the horizontal shifts across stages as well as the vertical transitions between levels. Within the serial constraints of the DES algorithm and the absence of intermediate input, the feedback is immediate, visible and animated as an indication of the transition to the next stage.

Visualization and usability: implementing a user interface that conforms to sound HCI principles is a fundamental requirement. There is an overlap between interactivity and usability [14]. For the animation these are expressed in terms of the following heuristics:

- ***Visibility of system status****:* users are informed about the progress through the different stages of the algorithm and within each stage.
- ***User control and freedom****:* clear navigation with the aid buttons; ability to skip the operations and to generate partial results progressively.
- ***Consistency and standards****: c*onsistent use of standard mathematical notation and consistent structure of the animation within the DES animation and across all the encryption animations; provision of invariants such as status bar.
- ***Recognition rather than recall*****:** information is introduced in stages with explicit instructions in order to minimise cognitive and memory load.
- ***Flexibility and efficiency of use:*** The ability to skip through operations and avoid repetitive tasks with no educational value. The system offers flexibility despite the inherent sequential nature of the DES algorithm and the high dependence of the generation of new data on the output of the previous stage.
- ***Aesthetic and minimalist design:*** only relevant information is included so as to maintain interest and focus on the underlying principles of the encryption.

5.2 Related DES Animations

DES encryption has been the subject of some animations, which are either available on line or described in the literature. Their evaluation was conducted mainly in terms of the scope of the animation, the level of abstraction and the degree of interactivity.

The implementation of DES by Eriko Nurvitadhi and Elias Khair [8], demonstrates DES encryption in a Java Applet. The user is allowed to enter a key and plaintext, which is then encrypted using ECB. This is a useful program for viewing intermediate values and the final result of the encryption; it does not however open the 'black box' of DES and there is also a lack of explanation of the different stages of the algorithm. The application assumes that the user is already familiar with the inner workings of DES. Apart from the initial input, there is no user interaction and the encryption runs from start to finish without user intervention. This animation has little pedagogical value since an understanding of DES requires access to additional sources; the level of abstraction is restricted since it operates at block level.

In contrast, an application that operates at bit level is presented in [9]. It demonstrates the diffusion property that results from the animation of the whole Feister cipher. The movement of bits provided is the only dynamic feature in this animation, as they traverse the labelled stages of the cipher. All the components of the Feistel cipher are presented in one single static diagram; this increases the cognitive load of the learner and its instructional content is weak. Users are passive viewers of an animation with no control over its pacing or its sequencing.

McNear and Petty describe a pedagogical tool where the stages of the encryption in the animation are clearly identified. Although some user control is provided over the pacing of the animation the operations take place at block level. Overall, the animation appears too compact to be used as a stand-alone tool and all the operations are presented in one screen with little explanation. Although the operational framework is made up of a few screenshots it is on the whole static [15].

CrypTool provides a more sophisticated animation of DES [16]. The behaviour of the algorithm is presented by a sequence of screen shots of the different stages. The stages are gradually introduced with adequate explanation, supported by an overview of the encryption process. An effort is made to explain the function of the different tables. This animation offers a procedural description of the algorithm and is very diagrammatic in appearance and essentially uniform in colour. The different levels of abstraction are predefined and illustrated accordingly. Although this animation has the merit of opening the 'black box' of DES there is a lack of explicit control by the user on the pacing of the animation.

5.3 DES Animations and Pedagogical Value

The appraisal of some DES animations has highlighted deficiencies in the degree of interaction, the granularity of abstraction and the heavy cognitive load imposed on the learner. With the relative exception of CrypTool the operation framework is static while the data is dynamically modified. In contrast, the proposed DES tool in this paper demonstrates the dynamic nature of both operational framework and data. The learner is able to interact with individual components such as permutation tables. It also offers a high degree of interactivity at different levels of abstraction. The animation enhances the dynamic unfolding of the algorithm and is appropriate in encryption. More static approaches to operational framework are however more appropriate in many applications [17].

The DES animation described in this paper satisfies most of the criteria required for a tool with an adequate pedagogical value. Its significance is confirmed by its relevance to formal models, its compliance with sound HCI principles and its agreement with good practice as elicited by empirical studies. This covers the ability of learners to interact directly with the animation outside the classroom, and to focus on the logical steps of the algorithm with minimum distraction by the meta-tools. It also implies a meaningful integration of text and animation and a clear indication of the status of the encryption process at any stage [18]. The mapping between algorithm and animation has also enhanced its relevance. All these features confer to the DES animation an intrinsic pedagogical value.

6 Conclusion

This paper has presented a DES animation, which possesses an intrinsic pedagogical value. The animation conforms to sound engineering and pedagogical principles, and its evaluation has highlighted the importance of designing animations that support a high level of interactivity and promote autonomy. Although this animation can be used effectively as a standalone unit of instruction, its impact is greater when used as a means of engaging students of computer security with the complex topic of encryption. The flexibility and availability of this animation makes it a useful contribution to the teaching of encryption, especially as a component of a blended programme.

References

[1] Shaffer C.A., Cooper M., and Edwards S.E., Algorithm Visualisation: A Report on the State of the Field, *ACM SIGSE'07,* Covington, Kentucky, USA, March 2007, pp150-154.

[2] Muldner T. and Shakshuki E., A New Approach to Learning Algorithms, *Proceedings of the IEEE International Conference on Information Technology: Coding and Computing IEEE (ITCC'04)*, Las Vegas, USA, 2004, pp141-145.

[3] Guzdial M. and Soloway E., Teaching the Nintendo Generation to Program, *CACM 45(4),* 2002*,* pp17-21.

[4] Waern Y., On the Dynamics of Mental Models, *In Mental Models and Human-Computer Interaction 1*, Eds. D. Ackermann and M.J. Tauber, Elsevier Science Publishing Company, New York, 1990, pp73-93.

[5]. Ursulet S. and Gillet D., Introducing Flexibility in Traditional Engineering Education by Providing Dedicated On-Line Experimentation and Tutoring Resources", *International Conference on Engineering Education*, Manchester, UK, August 2002, pp1-4.

[6] Hansen S.R., Narayanan N.H. and Schrimpsher D., Helping learners visualize and comprehend algorithms, *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning, 2(1*), 2000.

[7] Stasko J., Badre A. and Lewis C., Do Algorithm Animation Assist Learning? An Empirical Study and Analysis, *Proceedings INTERCHI'93 Conference on Human Factors in Computing Systems*, Amsterdam, Netherlands, April 1993, pp61-66.

[8] Nurvitadhi E. and Khair E. *DES Java Implementation.* Oregon State University, *http://islab.oregonstate.edu/koc/ece575/02Project/Nur+Kha/*

[9] Fikus E. and Crandall J., Diffusion in DE, *http://nsfsecurity.pr.erau.edu/crypto/sdesf.html*

[10] Moore M.G., Three Types of Interaction, *Journal of Distance Education, vol. 3, no.2,* 1989, pp1-6.

[11] Tudoreanu M.E, Wu R., Hamilton-Taylor A., and Kramer E. Empirical Evidence that Animation Promotes Understanding of Distributed Algorithms, *Proceedings of the IEEE Symposium on Human Centric Computing Languages and Environments (HCC'02),* 2002, pp236-243.

[12] Reichert R. and Hartmann W. On the Learning of E-Learning, *Proceedings of the EDMEDIA 2004 – World Conference on Education Multimedia, Hypermedia and Telecommunications*, June 2004, Lugano, Switzerland, pp1590-1595.

[13] Schulmeister R., Taxonomy of multimedia Component interactivity. A Contribution to the Current Metadata Debate. *Studies in Communication Sciences. 3(1),* 2003, pp61-80;also in *http://www.izhd.uni-hamburg.de/pdfs/Interactivity.pdf.*

[14] Nielsen, J., Ten usability heuristics, 2005 *http://www.useit.com/papers/heuristic/heuristic_list.htm.*

[15] McNear C. and Petty C., A Free, Readily Upgradeable, Interactive Tool for Teaching Encryption Algorithms, *43rd ACM Southeast Conference*, Kennesaw, GA, USA, March 2005, pp1:280-285.

[16] CrypTool DES animation, *http://www.animal.ahrgr.de/showAnimationDetails.php3?lang=en&anim=214*.

[17] Anane R., Crowther S., Beadle J. and Theodoropoulos G. eLearning Content Provision. *Proceedings of the 15th IEEE International Workshop on Databases and Expert Systems (DEXA04),* Zaragoza, Spain, August 2004, pp420-425.

[18] Saraiya P., Shaffer C., McCrickard D. and North C., Effective features of algorithm visualization, *Proceedings of the 35th SGSE Technical Symposium on Computer Science Education (SGSE'04),* Norfolk MA, March 2204, pp382-386.